Keysight 4080 Series Parametric Test System

Programming Reference for C Users

KEYSIGHT TECHNOLOGIES

Notices

Copyright Notice

Manual Part Number

N9100-90170

Edition

Edition 1, March 2020

Printed in:

Printed in Japan

Published by:

Keysight Technologies Japan K.K.
9-1, Takakura-cho, Hachioji-shi,
Tokyo 192-0033 Japan

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

U.S. Government Rights

The Software is “commercial computer software,” as defined by Federal Acquisition Regulation (“FAR”) 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement (“DFARS”) 227.7202, the U.S. government acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in its End User License Agreement (EULA), a copy of which can be found at http://www.keysight.com/find/sweula. The license set forth in the EULA represents the exclusive authority by which the U.S. government may use, modify, distribute, or disclose the Software. The EULA and the license set forth therein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software documentation. No additional government requirements beyond those set forth in the EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in the EULA. Keysight shall be under no obligation to update, revise or otherwise modify the Software. With respect to any technical data as defined by FAR 2.101, pursuant to FAR 12.211 and 27.404.2 and DFARS 227.7102, the U.S. government acquires no greater than Limited Rights as defined in FAR 27.401 or DFAR 227.7103-5 (c), as applicable in any technical data.

Warranty

THE MATERIAL CONTAINED IN THIS DOCUMENT IS PROVIDED “AS IS,” AND IS SUBJECT TO BEING CHANGED, WITHOUT NOTICE, IN FUTURE EDITIONS. FURTHER, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, KEYSIGHT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, WITH REGARD TO THIS MANUAL AND ANY INFORMATION CONTAINED HEREIN, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. KEYSIGHT SHALL NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR OF ANY INFORMATION CONTAINED HEREIN. SHOULD KEYSIGHT AND THE USER HAVE A SEPARATE WRITTEN AGREEMENT WITH WARRANTY TERMS COVERING THE MATERIAL IN THIS DOCUMENT THAT CONFLICT WITH THESE TERMS, THE WARRANTY TERMS IN THE SEPARATE AGREEMENT SHALL CONTROL.

Safety Notices

CAUTION

A CAUTION notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a CAUTION notice until the indicated conditions are fully understood and met.

WARNING

A WARNING notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a WARNING notice until the indicated conditions are fully understood and met.

Contents

1 Using TIS Reference for C Users

How to Use the Reference Pages 14
Function Name 14
Synopsis 14
Example 16
See Also 16

Assumptions 17
Abbreviations 17

Supported Library Files 19

2 TIS Function Reference

A 22
address3458 22
auto_cal 23

C 24
ch_sw_off_pg, ch_sw_off_pg2, ch_sw_off_pg3 24
ch_sw_on_pg, ch_sw_on_pg2, ch_sw_on_pg3 26
cmu_adjust 28
connect_chuck 29
connect_ex 31
connect_pin, connect_pin2, connect_pin3 32
connect_th 38
conv_mode 41
conv_ztr. 42
complx 43

D 44
disable_corr 44
disable_port, disable_port2, disable_port3 45
disable_port_all 49

disable_spa 50
disable_stand_by_port 51
disconnect_all 51
dispoff_spa 52
dispon_spa 53

E 54
error_info 54

F 56
fnaux 56
fncmh 57
fncml 58
fncmu84 59
fngcmu 60
fngnd 61
fngsmu 62
fnpg 63
fnport 65
fnsmu 67
fnunitsmu 69
fnvm 70
fnvs 71
force_i 72
force_meas, force_meas_t 77
force_pg 81
force_v 84
frequency 91
ftoim 93

G 94
get_freq 94
get_freq_step 95

get_modelname 96
gnd_open_guard 97
H 98
high_voltage_state 98
hp_init_system 99
htos 101
htoy 103
htoz 104
I 106
imabs 106
imadd 107
imaginary 108
imargument 109
imdiv 110
imexp 110
imln 111
imlog10 111
immultiply 112
imreal 113
imsqrt 113
impower 114
imsub 115
init_spa 116
init_system 117
is_on_line 121
ivt_measure 122
ivt_set_iv 125
ivt_set_sync 128
ivt_set_t 131
L 134

load_corr_data 134
load_hfportmap 135
load_pg 136
load_pg_conf 137
long_cal 138

M 139
measure_bdv 139
measure_cmu84 142
measure_cpg, measure_cpgt 147
measure_csrs, measure_csrst 150
measure_i, measure_it 153
measure_ileak 159
measure_m 162
measure_p 165
measure_spa 169
measure_v, measure_vt 172
measure_vth 176
measure_v3458 179
measure_ztr, measure_ztrt 180

O 183
open_corr84 183
open_corr_th 184

P. 186
p_ivt_measure 186
p_ivt_set_iv 189
p_ivt_status 192
p_measure_m 194
p_measure_vth 197
PORT 201
port_status, port_status_t 206

prep_pg 212
pround_cmu 213
p_rsearch_iv 214
p_rsearch_stop 218
p_rsweep_miv, p_rsweep_mivt 219
p_rsweep_stop 225
p_search_iv 226
p_search_iv2 229
p_set_bdv_srch 233
p_set_group 239
p_set_iv 241
p_set_lsearch 249
p_set_post_mode 255
p_set_skip 256
p_set_stop_mode 258
p_set_sync (for synchronous search measurements) 259
p_set_sync (for synchronous sweep measurements) 263
p_set_vth 268
p_status_miv 272
p_sweep_miv 274

Q 278
query_cmu 278
query_pgu 279

R 280
range_mode 280
report_event 282
reset_timestamp 283
rfs_calc_coef1 284
rfs_calc_coef2 286
rfs_clear_coef 287
rfs_cmu_cal 288

rfs_deembed_s1 290
rfs_deembed_s2 291
rfs_ft 292
rfs_init_cal 293
rfs_load_coef_r 294
rfs_measure_cmu 296
rfs_measure_s1, rfs_meas_s1_0 298
rfs_measure_s2, rfs_meas_s2_0 300
rfs_sel_calkit 302
rfs_set_adc 303
rfs_set_average 304
rfs_set_cmu 305
rfs_set_cv 307
rfs_set_s 309
rfs_sol_cal 311
rfs_store_coef 314
rfs_sweep_cf 317
rfs_sweep_cv 319
rfs_thru_cal 321
rfs_z0 323
rfs_z0_r 323
round_level_pg 324
rsearch_iv 325
rsearch_stop 328
rsweep_iv, rsweep_ivt 329
rsweep_miv, rsweep_mivt 335
rsweep_stop 340

S 341
save_pg 341
search_iv 342
search_iv2 346

set_adc 351
set_adc_i 354
set_adc_v 358
set_adc3458 362
set_bdv 363
set_bdv_search 366
set_bsearch 371
set_cmu 378
set_cmu84 381
set_cv 386
set_cv84 389
set_cvf 392
set_event_signal 396
set_freq 398
set_ileak 402
set_iv 405
set_level_pg 412
set_lsearch 415
set_pbias 420
set_piv 426
set_rangemode 433
set_skip 435
set_smu 437
set_smu_ch 439
set_spa 441
set_sr_control 443
set_sr_mode 444
set_sr_pg 446
set_sync (for synchronous search measurements) 449
set_sync (for synchronous sweep measurements) 453
set_time_pg, set_time_pg1 458
set_timestamp 462

set_type_pg 464
set_vth 468
set_wait_time 471
set_zf 472
short_cal 473
short_corr84 474
short_corr_th 475
spgu_alwg_force 477
spgu_alwg_load 478
spgu_alwg_r 479
spgu_measure_r 480
spgu_mode 482
spgu_mode_q. 483
stand_by_port, stand_by_port2, stand_by_port3 484
start_pg 486
status_corr 487
status_cvf 488
status_miv 490
stoh 491
stop_pg 492
store_corr_data 493
stoy. 494
stoy1. 496
stoz. 497
stoz1. 499
sweep_cv 500
sweep_cvf 502
sweep_cv84 505
sweep_iv 508
sweep_miv 511
sweep_spa 515
sweep_zf 516

switch_sr 518
switch_sr_wait 520
sync_th 522
T 523
tap_port 523
W 525
wait_event 525
wait_th 527
Y 528
ytoh 528
ytos 530
ytos1 532
ytoz 533
ytoz1 534
Z 535
ztoh 535
ztos 537
ztos1 539
ztoy 540
ztoy1 541

3 Error Messages

1-xxxxx 544
2-xxxxx 551
3-xxxxx 560
4-xxxxx 574
5-xxxxx 580
6-xxxxx 581

7-xxxxx 621
8-xxxxx 622
42-xxxxx 633

4 Unsupported 4062/4070 TIS Functions

1 Using TIS Reference for C Users

How to Use the Reference Pages 14
Assumptions 17
Supported Library Files 19

This manual describes the C functions provided for the Keysight 4080 series Parametric Test System.

C Library functions include the following:

- Test Instruction Set (TIS) functions, which are used for measurement programming.
- Functions, which are used for measurement programming of the SPA (Spectrum Analyzer or Signal Analyzer).

How to Use the Reference Pages

This manual describes the functions in alphabetical order.

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

- Function name
- Synopsis
- Example
- See Also

If a section does not apply to a function, the section is omitted.

Function Name

The function name is listed at the top of the description module with a brief description. The functions are listed in alphabetical order.

Synopsis

The synopsis of the function is printed as in following example:

```
int connect_ex (int ports_array[n], int pins_array[n], int n);
```

NOTE

Return value

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

0:	Successful completion
–1:	Error occurred

The include line must be at beginning of your program if you want to use the function.

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

This line specifies the name of file that contains various necessary information for the function, such as function declaration and macro definitions.

The arguments of a function specify values to pass to the function or where to return values to a function. See the following example:

Argument	Description
ports_array	Specify a pointer to an integer array that determines the measurement ports. This array should have *n* elements. The array should contain the port addresses that are connected to the pins specified in *pins_array*.
pins_array	Specify a pointer to an integer array that determines the measurement pins. This array should have *n* elements. The array should contain the pin numbers that are connected to the ports specified in *ports_array*.
n	Specify effective array size for connection. *n* must be less than or equal to the size of *ports_array* and *pins_array*.

The arguments are listed in a table and each argument is described briefly in the table. The details about each argument may be further described after the table. 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.

The description after the table provides details about the function, including information about offline mode, compliance values, and return values. Along with text, the description may include tables and figures.

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.

Assumptions

To simplify the function descriptions, certain information is assumed. This section lists the information that you should be familiar with to effectively use this manual.

This manual assumes that you are familiar with the 4080 system. Please see the other 4080 manuals if you are not familiar with the system.

This manual also assumes that you are familiar with the C programming language.

Abbreviations

Throughout this manual, the following abbreviations are used to identify subsystems and instruments of the 4080.

SMS:	**S**witching **M**atrix **S**ubsystem
SWM:	**SW**itching **M**atrix
DCS:	**DC** Measurement **S**ubsystem: consists of SMUs and DVM
CMS:	**C**apacitance **M**easurement **S**ubsystem
HSCMU:	**H**igh-**S**peed **C**apacitance **M**easurement **U**nit
CMU:	**C**apacitance **M**easurement **U**nit (LCR Meter)
SMU:	**S**ource **M**onitor **U**nit
HPSMU:	**H**igh **P**ower **S**ource **M**onitor **U**nit
HRSMU:	**H**igh **R**esolution **S**ource **M**onitor **U**nit
DVM:	**D**igital **V**olt **M**eter (Digital Voltage Multimeter)
PNA:	PNA Microwave Network Analyzer (Keysight E8362B/C)
RFU:	**R**adio **F**requency Measurement **U**nit
SPA:	**SP**ectrum **A**nalyzer (Signal Analyzer)

PGU:	**P**ulse **G**enerator **U**nit (Keysight 81150A)
SPGU:	**S**emiconductor **P**ulse **G**enerator **U**nit
GNDU:	**G**rou**ND U**nit
TIS:	**T**est **I**nstruction **S**et

NOTE

About 81150A

The 81150A is supported by the 4080 system software revision F.05.20 or later.

NOTE

PG and PGU

“PG” stands for Pulse Generator. “PGU” stands for Pulse Generator Unit.

In this manual, “PG” indicates a pulse output channel and “PGU” indicates the 81150A. “PGU” is also used to indicate the SPGU when describing functions common to the 81150A and the SPGU.

Supported Library Files

This following describes files of the supported C software:

Table 1-1 Keysight 4080 Library Files

File Name	Description
/opt/hp4070/lib/libhp4070.a /opt/hp4070/lib/libhp4070.so	These are the C Library files. libhp4070.a is an archived library and libhp4070.so is a shared object. Both libraries consist of TIS, PCL, and FCL functions, and some prober support functions.
/opt/hp4070/lib/libspa.a /opt/hp4070/lib/libspa.so	These are the C Library files. libspa.a is an archived library and libspa.so is a shared object. These files are for the SPA.
/opt/hp4070/include/tis.h	This include file contains the function declarations and the macro definitions for the TIS functions.
/opt/hp4070/include/spa.h	This include file contains the function declarations for the SPA control functions.

Using TIS Reference for C Users
Supported Library Files
Abbreviations

2 TIS Function Reference

This chapter describes each function in alphabetical order.

A

This section describes the C functions that begin with A.

address3458

Revision	B.01.10 or later
Control	DVM

This function returns a pointer to a character variable where the GPIB address of DVM is stored.

Synopsis

```
char *address3458 (void);
```

GPIB address is stored as "*ll*,*bb*".

Where, *ll*: interface select code *bb*: primary GPIB address

auto_cal

Revision	C.04.00 or later
Control	SMU, HSCMU, SPGU

This function sets the periodic execution of self-calibration of the SMU, HSCMU, and SPGU.

Synopsis

```
int auto_cal (double interval);
```

Argument	Description
interval	Specify the execution interval in minute. Note that the self-calibration will be executed only if the following conditions are met: · The specified interval passes after the previous self-calibration and the previous init_system. · No TIS functions have not been executed after the previous init_system. If 0 or a negative value is specified, the periodic execution is disabled.

C

This section describes the C functions that begin with C.

ch_sw_off_pg, ch_sw_off_pg2, ch_sw_off_pg3

Revision	B.01.10 or later
Control	PGU, SPGU

This function opens the output switch of the specified pulse generator (disables output).

Synopsis

```
int ch_sw_off_pg (int pg);
int ch_sw_off_pg2 (int pg1, int pg2);
int ch_sw_off_pg3 (int pg1, int pg2, int pg3);
```

Argument	Description	
pg1, *pg2*, *pg3*	Specify PG unit address to set the output switch off.	
	PG	Unit Address
	PG11	20281
	PG12	20282
	PG21	20283
	PG22	20284
	PG31	20285
	PG32	20286
	PG41	20287
	PG42	20288
	PG51	20289
	PG52	20290

For PGU (81150A), PG11 and PG12 are available.

For SPGU, the available units depend on the system configuration.

This function opens the output switch of the specified PGs (disables output).

This function is used to control PGs from TIS that are not defined in the present pulse generator configuration information initially loaded from the `/etc/opt/hp4070/config/pgconnection` file during the `hp4070` logon, or loaded using `load_pg_conf` function.

If 0 is specified as *pg1*, this function disables the output of all available PGs that defined in the pulse generator configuration information.

Example

```
int err;

err=ch_sw_off_pg(PORT(5,12));
```

See Also

- ch_sw_on_pg

ch_sw_on_pg, ch_sw_on_pg2, ch_sw_on_pg3

Revision	B.02.00 or later
Control	PGU, SPGU

This function closes the output switch of the specified pulse generator (enables output).

Synopsis

```
int ch_sw_on_pg (int pg);
int ch_sw_on_pg2 (int pg1, int pg2);
int ch_sw_on_pg3 (int pg1, int pg2, int pg3);
```

Argument	Description	
pg1, *pg2*, *pg3*	Specify PG unit address to set the output switch on.	
	PG	Unit Address
	PG11	20281
	PG12	20282
	PG21	20283
	PG22	20284
	PG31	20285
	PG32	20286
	PG41	20287
	PG42	20288
	PG51	20289
	PG52	20290

For PGU (81150A), PG11 and PG12 are available.

For SPGU, the available units depend on the system configuration.

This function closes the output switch of the specified PGs and the PGs starts to force the pulse base voltage.

This function is used to control PGs from TIS that are not defined in the present pulse generator configuration information initially loaded from the `/etc/opt/hp4070/config/pgconnection` file during the `hp4070` logon or loaded using `load_pg_conf` function.

If 0 is specified as *pg1*, this function closes the output switch of all available PGs defined in the pulse generator configuration information.

If an illegal voltage is set to the specified PG, the output switch does not close.

Example

```
int err;
err=ch_sw_on_pg(PORT(5,12));
```

See Also

- ch_sw_off_pg

cmu_adjust

Revision	C.04.00 or later
Control	HSCMU

This function executes the calibration of HSCMU and capacitance measurement path in the system.

Synopsis

```
int cmu_adjust (void);
```

When the pin board configuration or the probe card is changed, you should execute this function.

Before executing this function, connect CML port to any pins.

Example

```
#include "/opt/hp4070/include/tis.h"
connect_pin(CML,1);
cmu_adjust();
```

connect_chuck

Revision	A.01.00 or later
Control	SWM

This function connects the specified port to the Chuck Connection Output connector of the testhead.

Synopsis

```
int connect_chuck (int port);
```

Argument	Description
port	Specify SWM port address to connect to the Chuck Connection Output connector of the testhead.

You can specify 0 or one of following macros or values for *port*:

Macro	Value and Meaning
SMU1	20001
SMU2	20002
SMU3	20003
SMU4	20004
SMU5	20005
SMU6	20006
SMU7	20007
SMU8	20008
GNDU	20009 (Ground Unit)
none	20101 (AUX1 port)
none	20102 (AUX2 port)
none	20103 (AUX3 port)

none	20104 (AUX4 port)
DVMH	20105 (DVM high port: AUX5)
DVML	20106 (DVM low port: AUX6)
CMH	20107 (CMS high port: AUX7)
CML	20108 (CMS low port: AUX8)
INGRD1	20109 (Guard of SMU1)
INGRD2	20110 (Guard of SMU2)
EXGRD1	20111 (Guard of AUX1)
EXGRD2	20112 (Guard of AUX2)
CCOM	20113 (CMS guard terminal)

For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

If the *port* is 0, chuck connection is disconnected.

The connect_chuck function is functionally identical to `connect_pin(`*port*`, 49)`. Refer to the connect_pin function for further details.

You *cannot* specify port address 20114 (`EXIO` Extended Path).

Example

```
int err;
err=connect_chuck(PORT(0,1));
```

See Also

- connect_pin

connect_ex

Revision	B.02.10 or later
Control	SWM

This function connects the multiple DC measurement ports and pins as specified in the arrays. This function does not support the RF matrix.

Synopsis

```
int connect_ex (int ports_array[n], int pins_array[n], int n);
```

Argument	Range Restrictions/Description
ports_array	Specify a pointer to an integer array that determines the measurement ports. This array should have *n* elements. The array should contain the port addresses that are connected to the pins specified in *pins_array*.
pins_array	Specify a pointer to an integer array that determines the measurement pins. This array should have *n* elements. The array should contain the pin numbers that are connected to the ports specified in *ports_array*.
n	Specify effective array size for connection. *n* must be less than or equal to the size of *ports_array* and *pins_array*.

This function connects the ports specified in the *ports_array* to the pins specified in the corresponding elements of *pins_array* and disconnect the pins that are not specified in the *pins_array*.

The size of the *ports_array* and *pins_array* must be the same. The *n* should be less than or equal to the array size.

If the *n* is less than the size of the *ports_array* and the *pins_array*, this function covers the first *n* port-pin pairs.

If zero (0) is specified as an array element, the connection corresponding to the element is ignored. This function ignores the port-pin pair if either is 0.

If you select multiple ports for the same pin, the pin is connected to the port that has the largest port number.

connect_pin, connect_pin2, connect_pin3

Revision	A.01.00 or later
Control	SWM

These functions connect an SWM port to SWM pin or pins.

For RF direct output (without using RF matrix), it is not necessary to call these functions.

The connect_pin2 and connect_pin3 functions do not support the RF matrix.

Synopsis

```
int connect_pin (int port, int pin);
int connect_pin2 (int port, int pin1, int pin2);
int connect_pin3 (int port, int pin1, int pin2, int pin3);
```

Argument	Range Restrictions/Description
port	Specify SWM port address to connect to specified SWM *pin*. (Note 1)
pin, *pin1*, *pin2*, *pin3*	Specify SWM pin number to connect to specified SWM *port*. For DC matrix, 0 to 49. (Note 2) For RF matrix, 101 to 105, and 201 to 205. (Note 3)

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 HF1, HF2, or HF3 can connect to pin 1 through pin 24 only. HF4, HF5, or HF6 can connect to pin 25 through pin 48 only. THF1, THF2, THF3, or Extended Path can not connect to the chuck connection output connector of the testhead (pin number 49).

Note 3 RF11, RF21, RF31, RF41, and RFU11 can connect to pin 101 through 105 only, and RF12, RF22, RF32, RF42, and RFU12 can connect to pin 201 through 205 only.

You can specify 0 or one of following macros or values for *port*:

Macro	Port Address	Description
SMU1	20001	SMU1
SMU2	20002	SMU2
SMU3	20003	SMU3
SMU4	20004	SMU4
SMU5	20005	SMU5
SMU6	20006	SMU6
SMU7	20007	SMU7
SMU8	20008	SMU8
GNDU	20009	Ground Unit
AUX1	20101	AUX1 port
AUX2	20102	AUX2 port
AUX3	20103	AUX3 port
AUX4	20104	AUX4 port
AUX5	20105	AUX5 port
AUX6	20106	AUX6 port
AUX7	20107	AUX7 port
AUX8	20108	AUX8 port
DVMH	20105	DVM high port: AUX5
DVML	20106	DVM low port: AUX6
CMH	20107	CMS high port: AUX7
CML	20108	CMS low port: AUX8
INGRD1	20109	Guard of SMU1
INGRD2	20110	Guard of SMU2

Macro	Port Address	Description
EXGRD1	20111	Guard of AUX1
EXGRD2	20112	Guard of AUX2
CCOM	20113	CMS guard terminal
EXIO	20114	Extended Path
HF1	20201	HF1 port
HF2	20202	HF2 port
HF3	20203	HF3 port
HF4	20204	HF4 port
HF5	20205	HF5 port
HF6	20206	HF6 port
THF1	20211	THF1 port
THF2	20212	THF2 port
THF3	20213	THF3 port
RFU11	20301	RFU1 port1
RFU12	20302	RFU1 port2
RF11	20401	RF11 port
RF21	20402	RF21 port
RF31	20403	RF31 port
RF41	20404	RF41 port
RF12	20406	RF12 port
RF22	20407	RF22 port
RF32	20408	RF32 port
RF42	20409	RF42 port

Macro	Port Address	Description
PNA11	20509	PNA1 port1
PNA12	20510	PNA1 port2

The connect_pin function is for connecting one port to one pin, connect_pin2 is for connecting one port to two pins, and connect_pin3 is for connecting one port to three pins. connect_pin2 and connect_pin3 do not support RF matrix.

To connect more than three pins to one port, you can repeatedly execute one of these connection functions, or execute connect_th function to connect a range of pins.

Multiple ports *cannot* be connected to one SWM pin. So, if these functions specify the same *pin* for multiple ports, only the port specified in the most recent connection function is connected to the SWM pin.

If the *port* is 0, all pins specified by *pin* and the chuck pin are disconnected.

If the *pin* or *pin1* is 0, the specified port is disconnected from all pins.

If the *pin2* is 0, this function makes the connection of the *pin1* but does not change the connection of the *pin3*.

If the *pin3* is 0, this function makes the connections of the *pin1* and *pin2*.

If both *port* and *pin* are 0, all pins are disconnected from all ports.

connect_pin(*port*,49) is functionally identical to connect_chuck(*port*).

If the specified port connects to the PG, the PG closes its output switch and starts to force pulse base voltage when this function is executed.

If there is an Active unit when the connection function executes, all Active units except for the standby port are set to the zero output condition before switching. For the standby port, see “stand_by_port, stand_by_port2, stand_by_port3”.

After switching, SMU and CMU that were Active are returned to their previous conditions by the next force_i, force_v, force_meas, or force_meas_t function for any unit.

NOTE

Active Unit

Active unit means the measurement resource in the following state:

- SMU: Output or compliance setting is |V| > 2 V or |I| > 10 mA.
- HSCMU: Applies voltage other than 0 V.
- CMU: Applies voltage other than 0 V.
- PGU or SPGU: Applies pulse signal or applies pulse base voltage greater than 2 V.

NOTE

Zero Output

Zero Output means the following state:

- SMU

  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 μA

- HSCMU

  Output Voltage: 0 V

- CMU

  Output Voltage: 0 V

- PGU or SPGU

  Output Voltage (Pulse Base): 0 V

Example

```
int err;

err=connect_pin(PORT(0,1),32);
err=connect_pin2(PORT(0,1),12,16);
err=connect_pin3(PORT(0,9),28,32,36);
```

See Also

- connect_th, disconnect_all

connect_th

Revision	A.01.00 or later
Control	SWM

This function connects a specified DC measurement port to a range of SWM DC measurement pins. This function does not support the RF matrix.

Synopsis

```
int connect_th (int port, int first_pin, int last_pin);
```

Argument	Range Restrictions/Description
port	Specify an SWM port address to be connected to the specified *pin*. (Note 1)
first_pin, *last_pin*	Specify a range of pin numbers: 1 through 49 (Note 2)

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 HF1, HF2, or HF3 can connect to pin 1 through pin 24 only. HF4, HF5, or HF6 can connect to pin 25 through pin 48 only. THF1, THF2, THF3, or Extended Path can not connect to the chuck connection output connector of the testhead (pin number 49).

This function connects a range of DC measurement pins to one port. All available pins specified from *first_pin* to *last_pin* are connected to the specified *port*.

You can specify one of the following macros or values for *port*:

Macro	Port Address	Description
SMU1	20001	SMU1
SMU2	20002	SMU2
SMU3	20003	SMU3
SMU4	20004	SMU4
SMU5	20005	SMU5
SMU6	20006	SMU6
SMU7	20007	SMU7
SMU8	20008	SMU8
GNDU	20009	Ground Unit
AUX1	20101	AUX1 port
AUX2	20102	AUX2 port
AUX3	20103	AUX3 port
AUX4	20104	AUX4 port
AUX5	20105	AUX5 port
AUX6	20106	AUX6 port
AUX7	20107	AUX7 port
AUX8	20108	AUX8 port
DVMH	20105	DVM high port: AUX5
DVML	20106	DVM low port: AUX6
CMH	20107	CMS high port: AUX7
CML	20108	CMS low port: AUX8
INGRD1	20109	Guard of SMU1
INGRD2	20110	Guard of SMU2
EXGRD1	20111	Guard of AUX1

Macro	Port Address	Description
EXGRD2	20112	Guard of AUX2
CCOM	20113	CMS guard terminal
EXIO	20114	Extended Path
HF1	20201	HF1 port
HF2	20202	HF2 port
HF3	20203	HF3 port
HF4	20204	HF4 port
HF5	20205	HF5 port
HF6	20206	HF6 port
THF1	20211	THF1 port
THF2	20212	THF2 port
THF3	20213	THF3 port

Except for the number of pins connected, connect_th is functionally identical to the connect_pin function. Refer to the description of the connect_pin function for further details.

Example

```
int err;

err=connect_th(SMU3, 20, 48);
```

See Also

- connect_pin
- disconnect_all

conv_mode

Revision	A.01.00 or later
Control	n.a.

This function converts the parallel capacitance and conductance measurement values obtained by measure_cpg or measure_cmu84 to the capacitance and resistance values of a series equivalent circuit, and returns the values.

Synopsis

```
int conv_mode (double cp, double gp, double *cs, double *rs, double
               frequency);
```

Argument	Range Restrictions/Description
cp	Cp value. Must be greater than 0.
gp	Gp value. Must be greater than 0.
cs	Pointer to double variable in which to store the returned Cs value.
rs	Pointer to double variable in which to store the returned Rs value.
frequency	Measurement frequency. Must be greater than 0.

The measure_cpg or measure_cmu84 function measures the capacitance Cp and conductance Gp of a parallel equivalent circuit. This function converts these measurement values to the capacitance Cs and resistance Rs of a series equivalent circuit. The capacitance Cs and resistance Rs of a series circuit are calculated by the following equations:

$$Cs = (1 + D^2) \times Cp$$

$$Rs = \frac{D^2}{1 + D^2} \times \frac{1}{Gp}$$

where D=Gp/(ωCp), and ω=2πf, and f is frequency (measurement frequency).

conv_ztr

Revision	C.04.00 or later
Control	n.a.

This function converts the capacitance and conductance measurement values obtained by measure_cpg or measure_cmu84 to the impedance and phase values and returns the values.

Synopsis

```
int conv_ztr (double cp, double g, double *z, double *theta, double freq);
```

Argument	Range Restrictions
cp	Cp value [F]. Must be greater than 0.
g	Gp value [S]. Must be greater than 0.
z	Pointer to double variable in which to store the returned impedance value [Ω].
thera	Pointer to double variable in which to store the returned phase value [rad].
freq	Measurement frequency [Hz]. Must be greater than 0.

The measure_cpg or measure_cmu84 functions measure the capacitance cp and conductance g of a parallel equivalent circuit. This function converts these measurement values to the impedance z [Ω]and phase theta [rad].

complx

Revision	D.05.10 or later
Control	n.a.

This function returns a complex value consisting of real_value and imaginary_value which are of the double data type.

Synopsis

```
COMPLEX complx (double re_value, double im_value);
```

Argument	Range Restrictions/Description
re_value	Real part of a complex value
im_value	Imaginary part of a complex value

D

This section describes the C functions that begin with D.

disable_corr

Revision	A.01.00 or later
Control	HSCMU, CMU

This function clears the compensation data beyond the switching matrix pins on specified two capacitance measurement pins.

Synopsis

```
int disable_corr (int h_pin, int l_pin);
```

Argument	Range Restrictions/Description
h_pin	Specify the high capacitance measurement pin, which is connected to CMH. 1 though 48.
l_pin	Specify the low capacitance measurement pin, which is connected to CML. 1 through 48.

disable_corr function clears the compensation data beyond the switching matrix pins on specified two capacitance measurement pins.

To clear compensation data for all measurement pins, execute load_corr_data(0).

Example

```
disable_corr(1,2);
```

See Also

- open_corr_th
- short_corr_th
- load_corr_data

disable_port, disable_port2, disable_port3

Revision	A.01.00 or later
Control	SMU, HSCMU, CMU, PGU, SPGU, RFU

These functions set the specified units to the Zero Output state, even if specified unit is SMU and set to Standby mode.

To disable two or more pins or ports, use disable_port2, disable_port3, or disable_port_all.

Synopsis

```
int disable_port (int port);
int disable_port2 (int port1, int port2);
int disable_port3 (int port1, int port2, int port3);
```

Argument	Range Restrictions/Description
port, *port1*, *port2*, *port3*	Specify SWM port address of SMU, HSCMU,CMU, PGU, SPGU, or RFU, or a pin number that is currently connected to port that you want to disable.

You can specify 0 or one of following macros or values for *port*, or you specify a pin number that is currently connected to the port:

Macro	Port Address	Description
SMU1	20001	SMU1
SMU2	20002	SMU2
SMU3	20003	SMU3
SMU4	20004	SMU4
SMU5	20005	SMU5
SMU6	20006	SMU6

Macro	Port Address	Description
SMU7	20007	SMU7
SMU8	20008	SMU8
GNDU	20009	Ground Unit
AUX1	20101	AUX1 port
AUX2	20102	AUX2 port
AUX3	20103	AUX3 port
AUX4	20104	AUX4 port
AUX5	20105	AUX5 port
AUX6	20106	AUX6 port
AUX7	20107	AUX7 port
AUX8	20108	AUX8 port
DVMH	20105	DVM high port: AUX5
DVML	20106	DVM low port: AUX6
CMH	20107	CMS high port: AUX7
CML	20108	CMS low port: AUX8
HF1	20201	HF1 port
HF2	20202	HF2 port
HF3	20203	HF3 port
HF4	20204	HF4 port
HF5	20205	HF5 port
HF6	20206	HF6 port
THF1	20211	THF1 port
THF2	20212	THF2 port
THF3	20213	THF3 port

Macro	Port Address	Description
RFU11	20301	RFU1 port1
RFU12	20302	RFU1 port2
RF11	20401	RF11 port
RF21	20402	RF21 port
RF31	20403	RF31 port
RF41	20404	RF41 port
RF12	20406	RF12 port
RF22	20407	RF22 port
RF32	20408	RF32 port
RF42	20409	RF42 port
PNA11	20509	PNA1 port1
PNA12	20510	PNA1 port2

NOTE

For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

If 0 is specified, all ports are set to Zero Output when this function executes.

For SMU, Zero Output means as follows:

SMU mode	V source
Output range	No change or 20 V if SMU was in I source mode
Output voltage	0 V
Current compliance	100 μA

For HSCMU, CMU, PGU, and SPGU, Zero Output means 0 V.

For RFU, Zero Output disables RF signal output and sets DC bias source to the following state:

Output mode	V source
Output voltage range	No change
Output voltage	0 V
Current range	No change or maximum 100 μA
Current compliance	Range value or maximum 100 μA

Example

```
if (disable_port(0)==-1) error_rep();
```

See Also

- disable_port_all

disable_port_all

Revision	A.01.00 or later
Control	SMU, HSCMU, CMU, PGU, SPGU, RFU

This function sets all SMUs, HSCMU,CMU, PGU, SPGU, and RFUs to the Zero Output state.

Synopsis

```
int disable_port_all (void);
```

All units are set to Zero Output State when this function executes.

For SMU

SMU mode	V source
Output range	No change or 20 V if SMU was in I source mode
Output voltage	0 V
Current compliance	100 μA

For HSCMU, CMU, PGU, and SPGU Voltage output is set to 0 V.

For RFU Zero Output disables RF signal output and sets DC bias source to the following state:

Output mode	V source
Output voltage range	No change
Output voltage	0 V
Current range	No change or maximum 100 μA
Current compliance	Range value or maximum 100 μA

If you want to disable one, two, or three units, use disable_port, disable_port2, or disable_port_3.

Example

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

See Also

- disable_port
- disable_port2
- disable_port3

disable_spa

Revision	D.05.16 or later
Control	(SPA)

This is a dummy function that performs nothing for the spectrum analyzer. This function is provided so that exiting ring oscillator measurement programs that use 4070 can run on 4080 without modification. Do not use this function to make new measurement programs for 4080.

To initialize the SPA, use the init_system function.

Synopsis

```
#include "/opt/hp4070/include/spa.h"
   int disable_spa (void);
```

NOTE

The SPA does not generate any output voltage or current. Hence, there is no need to initialize the SPA before disconnection.

See Also

- init_system
- init_spa
- disable_port

disable_stand_by_port

Revision	A.01.00 or later
Control	SMU, RFU

This function cancels Standby mode for all SMU ports and bias source of RFUs.

Synopsis

```
int disable_stand_by_port (void);
```

Example

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

See Also

- stand_by_port
- stand_by_port2
- stand_by_port3

disconnect_all

Revision	A.01.00 or later
Control	SWM

This function disconnects all SWM pins from all SWM ports.

Synopsis

```
int disconnect_all (void);
```

This function disconnects all SWM pins from all SWM ports, then sets the SWM to the initialized status. The function is equivalent to the connect_pin(0,0) function, and is recommended for better program readability.

Example

```
int err;
err=disconnect_all();
```

See Also

- connect_pin

dispoff_spa

Revision	D.05.16 or later
Control	SPA

This function disables the display on the SPA, as well as the auto calibration function. Once this function is called, subsequent command executions will run faster.

This function stores a value of 1 as an internal variable (dispmode), and calls "Auto calibration OFF". When auto calibration is ON, measurements are slowed down by execution of calibrations.

This function has a timeout function. If the SPA does not respond within 30 seconds, the function sends an error message to the standard out, and returns –1.

Synopsis

```
#include "/opt/hp4070/include/spa.h"
   int dispoff_spa (void);
```

See Also

- dispon_spa

dispon_spa

Revision	D.05.16 or later
Control	SPA

This function enables the display on the SPA, as well as the auto calibration function. When this function is called, subsequent command executions will run slower.

This function stores a value of 0 as an internal variable (dispmode), and calls "Auto calibration ON". When auto calibration is ON, measurements are slowed down by execution of calibrations.

This function has a timeout function. If the SPA does not respond in 30 seconds, the function sends an error message to the standard out, and returns –1.

Synopsis

```
#include "/opt/hp4070/include/spa.h"
   int dispon_spa (void);
```

See Also

- dispoff_spa

E

This section describes the C function that begins with E.

error_info

Revision	A.01.00 or later
Control	n.a.

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

Synopsis

```
int error_info (int errn[2], char errm[1024]);
```

Argument	Range Restrictions/Description
errn	Pointer to an integer array in which to return the TIS error number and HSCMU, CMU, DVM, PGU, or SPGU error number (if available). errn[0]: TIS error number errn[1]: HSCMU or CMU error number (if errn[0] = 85) DVM error number (if errn[0] = 86) PGU error number (if errn[0] = 87) SPGU error number (if errn[0] = 90). 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.

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. The first element of the *errn* array is the TIS error number. The second element is the HSCMU or CMU error number if the TIS error number is

85, the DVM error number if the TIS error number is 86, the PGU error number if the TIS error number is 87, or the SPGU error number if the TIS error number is 90.

If the init_system function executes, 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);
```

F

This section describes the C functions that begin with F.

fnaux

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 port address of a specified 4062 AUX (auxiliary) port.

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fnaux (int numb);
```

Argument	Range Restrictions/Description
numb	An integer value specifying a 4062 AUX port number.

The following table shows the port address conversion by default port mapping table at factory shipment.

	4062		4080	
numb	Port	Port Address	Port Address	Port
1	AUX1	32705	20006	SMU6
2	AUX2	32706	20006	SMU6
3	AUX3	32708	20107	CMH
4	AUX4	32709	20108	CML

fncmh

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 CMH port address (32708) and is functionally identical to fnport(8).

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fncmh (void);
```

NOTE

Default Port Address Conversion

By default port mapping table at factory shipment, the 4080 TIS functions convert the 4062 CMH port address (32708) to the 4080 CMH port address (20107).

fncml

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 CML port address (32709) and is functionally identical to fnport(9).

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fncml (void);
```

NOTE

Default Port Address Conversion

By default port mapping table at factory shipment, the 4080 TIS functions convert the 4062 CML port address (32709) to the 4080 CML port address (20108).

fncmu84

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062's CMH84 or CML84 unit address of the 4062's CMU84.

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fncmu84 (int numb);
```

Argument	Range Restrictions/Description
numb	An integer value specifying a 4062 CMU84 unit number.

The following table shows the port address conversion by default port mapping table at factory shipment.

	4062		4080	
numb	Port	Port Address	Port Address	Port
1	CMH84	32676	20107	CMH
2	CML84	32677	20108	CML

fngcmu

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 GCMU port (Guard terminal of CMU) address (32711) and is functionally identical to fnport(11).

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fngcmu (void);
```

NOTE

Default Port Address Conversion

By default port mapping table at factory shipment, the 4080 TIS functions convert the 4062 GCMU port address (32711) to the 4080 GCMU port address (20113).

fngnd

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 GNDU port address and is functionally identical to fnport(7).

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fngnd (void);
```

NOTE

Default Port Address Conversion

By default port mapping table at factory shipment, the 4080 TIS functions convert the 4062 GNDU port address (32707) to the 4080 GNDU port address (20009).

fngsmu

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 GSMU port (guard of SMU1) address (32710) and is functionally identical to fnport(10).

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fngsmu (void);
```

NOTE

Default Port Address Conversion

By default port mapping table at factory shipment, the 4080 TIS functions convert the 4062 GSMU port (guard of SMU1) address (32710) to the 4080 GSMU2 port address (20110).

fnpg

Revision	B.02.00 or later
Control	n.a.

NOTE

4062F TIS function

This function is provided to execute the 4062F measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the unit address of the specified 4062F PGU.

TIS functions automatically convert the unit address of the 4062F PGU to the 4080 unit address. Table 2-1 shows the PGU unit address conversion.

Synopsis

```
int fnpg (int numb);
```

Argument	Range Restrictions/Description
numb	An integer value specifying PGU unit number.

Table 2-1 PGU Unit Address Conversion Table

	4062F		4080
numb	Unit	Unit Address	Unit Address
11	PG11	32581	20281
12	PG12	32582	20282
21	PG21	32583	20283
22	PG22	32584	20284
31	PG31	32585	20285

	4062F		4080
numb	Unit	Unit Address	Unit Address
32	PG32	32586	20286
41	PG41	32587	20287
42	PG42	32588	20288
51	PG51	32589	20289
52	PG52	32590	20290

For PGU (81150A), PG11 and PG12 are available.

For SPGU, the available units depend on the system configuration.

fnport

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062's port address of the specified port.

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fnport (int numb);
```

Argument	Range Restrictions/Description
numb	An integer value specifying a 4062's port number.

The following table shows the port address conversion by default port mapping table at factory shipment.

	4062		4080	
numb	Port	Port Address	Port Address	Port
1	SMU1	32701	20002	SMU2
2	SMU2	32702	20003	SMU3
3	SMU3	32703	20004	SMU4
4	SMU4	32704	20005	SMU5
5	AUX1	32705	20006	SMU6
6	AUX2	32706	20006	SMU6
7	GNDU	32707	20009	GNDU
8	AUX3/CMH	32708	20107	AUX7/CMH
9	AUX4/CML	32709	20108	AUX8/CML
10, –1	GSMU	32710	20110	GSMU2
11, –8, –9	GCMU	32711	20113	GCMU (Circuit Common)

fnsmu

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 port address of a specified the 4062 SMU port.

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fnsmu (int numb);
```

Argument	Range Restrictions/Description
numb	An integer value specifying a 4062 SWM SMU port position.

The following table shows the port address conversion by default port mapping table at factory shipment.

	4062		4080	
numb	Port	Port Address	Port Address	Port
1	SMU1	32701	20002	SMU2
2	SMU2	32702	20003	SMU3
3	SMU3	32703	20004	SMU4
4	SMU4	32704	20005	SMU5
5	AUX1	32705	20006	SMU6
6	AUX2	32706	20006	SMU6
–1	GSMU	32710	20109	GSMU1

fnunitsmu

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 unit address of the specified the 4062's SMU.

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fnunitsmu (int numb);
```

Argument	Range Restrictions/Description
numb	An integer value specifying SMU unit number.

The following table shows the SMU unit address conversion by default port mapping table at factory shipment.

	4062		4080	
numb	Unit	Unit Address	Port Address	Port
1	MPSMU	32621	20002	SMU2
3	HPSMU	32629	20003	SMU3
4	MPSMU	32622	20004	SMU4
5	MPSMU	32623	20005	SMU5

fnvm

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 unit address of the specified 4062's VM.

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fnvm (int numb);
```

Argument	Range Restrictions/Description
numb	An integer value specifying a 4062 VM unit number.

The following table shows the VM unit address conversion by default port mapping table at factory shipment.

	4062		4080	
numb	Unit	Unit Address	Port Address	Port
1	VM1	32634	20006	SMU6

fnvs

Revision	A.01.00 or later
Control	n.a.

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function returns the 4062 unit address of the specified 4062's VS.

TIS functions of 4080 convert the 4062's port address to the 4080 port address by using port mapping table.

Synopsis

```
int fnvs (int numb);
```

Argument	Range Restrictions/Description
numb	An integer value specifying a VS unit number.

The following table shows the VS unit address conversion by default port mapping table at factory shipment.

	4062		4080	
numb	Unit	Unit Address	Port Address	Port
1	VS1	32633	20006	SMU6

force_i

Revision	A.01.00 or later
Control	SMU, RFU

This function sets the output current of a specified SMU or the DC bias source of specified RFU.

Synopsis

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

Argument	Range Restriction/Description	
port	Specify the port from which to force current. You can specify the port address of SMU port or RFU that has a bias source or a pin number that is currently connected to the port.	
	port address:	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008), RFU11(20301), RFU12(20302).
	pin number:	1 to 49 (for SMU), 101 to 105 (for RFU11), or 201 to 205 (for RFU12)
current (Note 2)	Value of current to force:	
	Numeric expression [A]:	–0.1 to 0.1 (MPSMU, HRSMU) –1 to 1 (HPSMU) –0.14 to 0.14 (RFU with RF matrix) –0.2 to 0.2 (RFU direct output)
range (Note 2)	Output range of current:	
	0.0 for Auto range or Numeric expression [A]:	–0.1 to 0.1 (MPSMU, HRSMU) –1 to 1 (HPSMU) Ignored and fixed to auto range (RFU)

Argument	Range Restriction/Description	
compliance	Voltage compliance value:	
	REMAIN macro (Note 3) or Numeric expression [V]:	–100 to 100 (Note 4) (MPSMU, HRSMU) –200 to 200 (Note 4) (HPSMU) –7 to 7 (RFU with RF matrix) –40 to 40 (RFUdirect output)

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 For details about *current* and *range*, refer to description after this table.

Note 3 REMAIN can be specified only if optimization level is 0 or 1. REMAIN means the previous compliance setting if mode was I_SOURCE mode, or 20 V if mode was V_SOURCE mode.

Note 4 The maximum *compliance* depends on the specified *current* as described later.

To specify the port, use either the *port address*, or the *pin number* of any SWM pin connected to the port.

The *range* argument sets the current range of the SMU.

If you specify 0 for *range*, the SMU is set to Auto range mode. In auto-ranging mode, the SMU forces the current at the lowest range that can force the *current*. Refer to following table.

If you specify a value that is not an exact output range listed in following table, the range is used according to the following table. For example, if you set *current* and *range* to 11 nA, the 10 nA range is used.

If you specify a range value, the SMU is set to limited auto range mode. In limited auto-ranging mode, the SMU forces the 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*. Refer to following table.

Table 2-2 Resolution of Current Force (MPSMU and HPSMU)

Output Range	*current*	Setting Resolution (Note 1)
1 nA	0 A ≤ |*output current*| ≤ 1.15 nA	50 fA
10 nA	1.15 nA < |*output current*| ≤ 11.5 nA	500 fA
100 nA	11.5 nA < |*output current*| ≤ 115 nA	5 pA
1 μA	115 nA < |*output current*| ≤ 1.15 μA	50 pA
10 μA	1.15 μA < |*output current*| ≤ 11.5 μA	500 pA
100 μA	11.5 μA < |*output current*| ≤ 115 μA	5 nA
1 mA	115 μA < |*output current*| ≤ 1.15 mA	50 nA
10 mA	1.15 mA < |*output current*| ≤ 11.5 mA	500 nA
100 mA (MPSMU)	11.5 mA < |*output current*| ≤ 100 mA	5 μA
100 mA (HPSMU)	11.5 mA < |*output current*| ≤ 115 mA	5 μA
1 A (HPSMU)	115 mA < |*output current*| ≤ 1 A	50 μA

Note 1 Determined by actual range used to force the *output current*. See below.

Table 2-3 **Resolution of Current Force (HRSMU)**

Output Range	*current*	Setting Resolution (Note 1)
10 pA	0 A ≤ |*output current*| ≤ 11.5 pA	1 fA
100 pA	11.5 pA < |*output current*| ≤ 115 pA	5 fA
1 nA	115 pA < |*output current*| ≤ 1.15 nA	50 fA
10 nA	1.15 nA < |*output current*| ≤ 11.5 nA	500 fA
100 nA	11.5 nA < |*output current*| ≤ 115 nA	5 pA
1 μA	115 nA < |*output current*| ≤ 1.15 μA	50 pA
10 μA	1.15 μA < |*output current*| ≤ 11.5 μA	500 pA
100μ A	11.5 μA < |*output current*| ≤ 115 μA	5 nA
1 mA	115 μA < |*output current*| ≤ 1.15 mA	50 nA
10 mA	1.15 mA < |*output current*| ≤ 11.5 mA	500 nA
100 mA	11.5 mA < |*output current*| ≤ 100 mA	5 μA

Note 1 Determined by actual range used to force the *output current*. See below.

The output range actually used to force the *current* determines the setting resolution of the *current*. As shown in the above table, lower output range has better setting resolution. But a lower output range may increase settling time.

The *compliance* sets the maximum output voltage of an SMU operating as a current source (IS mode). The maximum *compliance* that you can set depends on the specified *current* as follows:

Table 2-4 **Output Current and Maximum Compliance (MPSMU and HRSMU)**

current Value	Maximum *compliance*
0 ≤ |*current*| ≤ 20 mA	100 V
20 mA < |*current*| ≤ 50 mA	40 V
50 mA < |*current*| ≤ 100 mA	20 V

Table 2-5 **Output Current and Maximum Compliance (HPSMU)**

current Value	Maximum *compliance*
0 ≤ |*output current*| ≤ 50 mA	200 V
50 mA < |*output current*| ≤ 125 mA	100 V
125 mA < |*output current*| ≤ 350 mA	40 V
350 mA < |*output current*| ≤ 700 mA	20 V
700 mA < |*output current*| ≤ 1 A	14 V

Example

```
if (force_i(drain, i_ds1, i_ds2, 2.0) == -1) error_rep();
```

See Also

- force_meas
- force_v

force_meas, force_meas_t

Revision	A.01.00 or later
Control	SMU

These functions force current or voltage from the specified SMU, measure the current or voltage using the specified SMU, then return the measurement value.

Synopsis

```
int force_meas (int source_port, int measure_port, int source_mode, double
            source, double *measure, double source_range, double
            measure_range, double compliance, double hold_time);

int force_meas_t (int source_port, int measure_port, int source_mode, double
            source, double *measure, double source_range, double
            measure_range, double compliance, double hold_time, double
            *time_stamp);
```

Argument	Range Restrictions/Description	
source_port	Specify the *source* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address:	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number:	1 to 49
measure_port	Specify the *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address:	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU6 (20007), SMU8 (20008).
	pin number:	1 to 49

Argument	Range Restrictions/Description	
source_mode	Specify voltage source (VS) or current source (IS) mode for *source_port*. You may choose from one of two macros: V_SOURCE (= 1) I_SOURCE (= 2)	
source	Specify the source output value.	
	When using SMU in V_SOURCE mode [V]:	–100 to 100 –200 to 200 (HPSMU)
	When using SMU in I_SOURCE mode [A]:	–0.1 to 0.1 (MPSMU, HRSMU) –1 to 1 (HPSMU) –1.4 to 1.4 (RFU)
measure	Pointer to double variable in which to store the measurement value.	
source_range	Specify the output range. 0.0 for Auto range mode or	
	When using SMU in V_SOURCE mode [V]:	–100 to 100 –200 to 200 (HPSMU)
	When using SMU in I_SOURCE mode [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)
measure_range	Specify the measurement range. 0.0 for Auto range mode or	
	When using SMU in V_SOURCE mode [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)
	When using SMU in I_SOURCE mode [V]:	–100 to 100 –200 to 200 (HPSMU)
compliance	Specify the compliance value. REMAIN macro (Note 2) or	
	If the SMU is in V_SOURCE mode [A]:	–0.1 to 0.1 (MPSMU, HRSMU) –1 to 1 (HPSMU) –1.4 to 1.4 (RFU)
	If the SMU is in I_SOURCE mode [V]:	–100 to 100 –200 to 200 (HPSMU)

Argument	Range Restrictions/Description
hold_time	Specify the wait time since the source SMU starts output until the measurement SMU starts measurement. 0 to 655.3500 Resolution: 0.0001
time_stamp (Note 3)	Pointer to double variable in which to return the time stamp value.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1. REMAIN means the previous compliance setting if source mode of search SMU is same as the previous mode. If was V_SOURCE mode, and is now I_SOURCE mode, compliance is set to 20 V. If was I_SOURCE mode, and is now V_SOURCE mode, compliance is set to 100 μA.

Note 3 Only for force_meas_t function.

For the available *source*, *source_range* and *compliance* values, see "force_v" for the voltage source or see "force_i" for the current source.

If the measurement SMU is different from the source SMU, execute the following function before this function for setting the measurement mode.

- force_v: Sets the SMU to current measurement mode.
- force_i: Sets the SMU to voltage measurement mode.

For details about the *measure_range*, refer to measure_v or measure_i.

If you execute force_meas_t function, the accumulated time from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command to measurement start is returned in seconds to *time_stamp*.

If reset_timestamp function has been executed after last init_system or /opt/hp4070/bin/hp4070 -start, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode This function returns simulated measurement values to the *measure* argument as follows. Returned values are determined by the *measure_range* as follows.

Table 2-6 **For current measurements:**

Unit	Meas. Range	Returned Value	Polarity
SMU	= 0	I compliance	Same as *source*
	≠ 0	Range Value	

Table 2-7 **For voltage measurements:**

Unit	Meas.Range	Returned Value	Polarity
SMU	n.a.	V compliance	Same as *source*

See Also

- force_i
- force_v
- measure_i
- measure_v

force_pg

Revision	B.02.00 or later
Control	PGU, SPGU

This function forces pulse from all pulse generators (PG) that are ready.

If SPGU is used in the ALWG mode, executing this function will cause an error.

Synopsis

```
int force_pg (double count, double period);
```

Argument	Range Restrictions/Description
count	Specify the number of repeat of pulse period. 0 to 1000000000
period	Specify the pulse period. Numeric expression [s]. If SPGU, 0 or T to 10.0 If PGU (81150A), 0 or T to 999.0 Resolution is 1E–8. 0 automatically sets the minimum period T (refer to the following description).

This function actually forces the pulses from the following PGs:

- All PGs that connect to the HF, THF, PS, or AUX ports that are connected to the measurement pins using connect_pin, connect_pin2, connect_pin3, connect_th, or connect_chuck functions.
- All PGs that do not connect to the HF, THF, PS, of AUX ports, and are output-enabled by ch_sw_on_pg function.

You must setup these PGs using the set_level_pg and set_time_pg functions before executing this function.

- *count*

  The fractional part of the specified value is truncated. For example, if you specify *count* to be 13.7, the pulse will be forced 13 times.

  If the specified *count* is a non-zero value, this function instructs the PG to force the pulses for the specified number of periods and then returns program execution to the program that called this function. You cannot abort the pulse output when the pulses are being forced from the measurement program.

  If the specified *count* is 0, this function makes the PG start to force pulses continuously, and immediately returns the program execution to the program. The pulses will continue to be forced until the next stop_pg, connect_pin, connect_pin2, connect_pin3, connect_th, init_system, or tap_port function execution.

- *period*

  The *period* must be greater than or equal to T given by the following formula.

  $T = MAX(T_{PGxx\text{-}2L}, T_{PGxx\text{-}3L}, T_{PG\text{-}CTRL})$

  Where,

  - Minimum pulse period T_{PGxx2L} for 2-level pulse output PG:

    $T_{PGxx\text{-}2L} = width1 + (trailing\_edge1) \times 1.25 + delay1 + 25$ ns

    This value is 0 if the 2-level pulse output PG is not used.

  - Minimum pulse period $T_{PGxx\text{-}3L}$ for 3-level pulse output PG:

    $T_{PGxx\text{-}3L} = MAX(A, B)$

    - $A = width1 + (trailing\_edge1) \times 1.25 + delay1 + 25$ ns
    - $B = width2 + (trailing\_edge2) \times 1.25 + delay2 + 25$ ns

    This value is 0 if the 3-level pulse output PG is not used.

  - Minimum pulse period $T_{PG\text{-}CTRL}$ for switch control PG:

    $T_{PG\text{-}CTRL} = width + delay + 0.5$ ms

    This value is 0 if the switch control PG is not used.

If *period* = 0 is specified, the pulse period is automatically set to T.

For the pulse setup parameters, see “set_time_pg, set_time_pg1” and “set_sr_pg”.

See Also start_pg, stop_pg, set_level_pg, set_time_pg

force_v

Revision	A.01.00 or later
Control	SMU, HSCMU, CMU, RFU

This function forces the specified voltage from the specified SMU, or forces the specified DC bias from the HSCMU, CMU, or RFU.

Synopsis

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

Arguments (for SMUs or RFU)

For HSCMU or CMU, refer to "Arguments (for HSCMU)" or “Arguments (for CMU)” later in this section.

Argument	Range Restrictions/Description	
port	Specify the port from which to force current. You can specify the SMU port address or a pin number that is currently connected to the port.	
	port address:	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008). RFU11(20301), RFU12(20302).
	pin number:	1 to 49 for SMU 101 to 105 for RFU11 201 to 205 for RFU12
voltage	Value of voltage to force:	
	Numeric expression [V]:	–100 to 100 (Note 2) –200 to 200 (Note 2) (HPSMU) –7 to 7 (RFU with RF matrix) –40 to 40 (RFU direct output)

Argument	Range Restrictions/Description	
range	Output range of voltage: For RFU, dummy parameter and auto-ranging is used. For SMU, 0.0 for Auto range mode or	
	Numeric expression [V]:	–100 to 100 (Note 2) –200 to 200 (Note 2) (HPSMU)
compliance	Current compliance value: REMAIN macro (for SMU) (Note 3) or	
	Numeric expression [A]:	–0.1 to 0.1 (Note 4) (MPSMU, HRSMU) –1 to 1 (Note 5) (HPSMU) –0.14 to 0.14 (RFU with RF matrix) –0.2 to 0.2 (RFU direct output)

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 For details for *voltage* and *range*, refer to description after this table.

Note 3 REMAIN can be specified only if optimization level is 0 or 1. REMAIN means the previous compliance setting if mode was V_SOURCE mode, or 100 μA if mode was I_SOURCE mode.

Note 4 The maximum *compliance* depends on the specified *voltage* as described later.

Note 5 The maximum *compliance* depends on the specified *range* as described later.

- *port*

  To specify the SMU, use either the *port address* of SMU or RFU, or the *pin number* of any SWM pin connected to the SMU or RFU port.

- *voltage*, *range*, and *compliance*

  The voltage range of the SMU is set by the *range* argument.

  If you specify an *range*, the SMU is fixed to specified range.

If you specify 0 as *range*, the SMU is set to Auto-ranging mode. In auto-ranging mode, the SMU forces the voltage at the lowest range that can force the *voltage*. Refer to following table.

Table 2-8 Resolution of Voltage Force

Output Range	*voltage*	Setting Resolution (Note 1)
2 V	0 V ≤ |*output voltage*| ≤ 2 V	100 μV
20 V	2 V < |*output voltage*| ≤ 20 V	1 mV
40 V	20 V < |*output voltage*| ≤ 40 V	2 mV
100 V	40 V < |*output voltage*| ≤ 100 V	5 mV
200 V (for HPSMU)	100 V < |*output voltage*| ≤ 200 V	

Note 1 Determined by actual range used to force the *voltage*. See below.

The output range actually used to force the *voltage* determines the setting resolution of the *voltage*. As shown in the above table, lower output range has better setting resolution.

The *compliance* sets the maximum output current of an SMU operating as a voltage source (V_SOURCE mode).

- If the SMU is MPSMU or HRSMU, the maximum *compliance* that you can set depends on the specified *voltage* as follows:

voltage Value	Maximum *compliance*
0 V ≤ |*voltage*| ≤ 20 V	100 mA
20 V < |*voltage*| ≤ 40 V	50 mA
40 V < |*voltage*| ≤ 100 V	20 mA

- If the SMU is HPSMU, the maximum *compliance* that you can set depends on the specified *range* and *voltage* as follows:

range Value	*voltage* Value	Maximum *compliance*
0 V ≤ |*range*| ≤ 20 V	0 V ≤ |*voltage*| ≤ 14 V	1 A
	14 V < |*voltage*| ≤ 20 V	700 mA
20 V < |*range*| ≤ 40 V	0 V ≤ |*voltage*| ≤ 40 V	350 mA
40 V < |*range*| ≤ 100 V	0 V ≤ |*voltage*| ≤ 100 V	125 mA
100 V < |*range*| ≤ 200 V	0 V ≤ |*voltage*| ≤ 200 V	50 mA

Do not set *compliance* too low.

An undesired current, in series with an SMU's output, may be generated as a result of dirty contacts caused by an electro-chemical reaction or a temperature difference between contacts.

If *compliance* is too low, this undesired current may cause SMU to reach I compliance, which would cause a voltage greater than the specified *voltage* to be applied to the test device.

Arguments (for HSCMU) For SMUs or RFU, refer to "Arguments (for SMUs or RFU)" earlier in this section.

Argument	Range Restrictions/Description	
port	Specify the CMH port. You can specify the CMH port address or a pin number that is currently connected to the CMH port.	
	port address	CMH or 20107
	pin number	1 to 49
voltage	Value of voltage to force. Numeric expression [V]: –10 to 10 (1 mV resolution)	
range	Any value. (Note 1)	
compliance	Numeric expression: any value (Note 1)	

Note 1 This is a dummy parameter, and is ignored.

To specify the CMH, use either the CMH *port address*, or the *pin number* of any SWM pin connected to the CMH port.

Output *voltage* defines the DC bias voltage for the spot capacitance measurement using HSCMU.

Arguments (for CMU) For SMUs or RFU, refer to "Arguments (for SMUs or RFU)" earlier in this section.

Argument	Range Restrictions/Description	
port	Specify the CMH port. You can specify the CMH port address or a pin number that is currently connected to the CMH port.	
	port address	CMH or 20107
	pin number	1 to 49
voltage	Value of voltage to force. Numeric expression [V]: –40 to 40 (Note 1)	
range	Any value. (Note 2)	
compliance	Numeric expression: any value (Note 2)	

Note 1 If the CMU is not equipped with a power amplifier (Option 321/531), the voltages available are 0, 1.5, and 2.0 V.

Note 2 This is a dummy parameter, and is ignored.

To specify the CMH, use either the CMH *port address*, or the *pin number* of any SWM pin connected to the CMH port.

Output *voltage* defines the dc bias voltage.

The allowable dc bias voltage is 0 through ±40.0 V, as shown in the following table:

Output *voltage* Value	Actual Dc Bias Voltage	Resolution
$0 \le \lvert voltage \rvert \le 4.000$	*voltage*	1 mV
$4.000 < \lvert voltage \rvert \le 8.000$	*voltage*	2 mV
$8.000 < \lvert voltage \rvert \le 20.000$	*voltage*	5 mV
$20.000 < \lvert voltage \rvert \le 40.00$	*voltage*	10 mV
$40.00 < \lvert voltage \rvert < 40.01$	±40.00 V	n.a.
$40.01 \le \lvert voltage \rvert$	error	n.a.

If CMU is 4284A and option 001 is *not* installed, the dc bias voltage can be set to 0 V, 1.5 V, or 2.0 V, as shown in the following table:

Output *voltage* Value	Actual Dc Bias Voltage
$0 \le \lvert voltage \rvert < 1.5$	0 V
$1.5 \le \lvert voltage \rvert < 2.0$	±1.5 V
$\lvert voltage \rvert = 2.0$	±2.0 V
$2.0 < \lvert voltage \rvert$	error

See Also

- force_i
- force_meas

frequency

Revision	C.04.00 or later
Control	HSCMU, CMU

This function sets the measurement frequency of the HSCMU, or CMU.

Synopsis

```
int frequency (double freq);
```

Argument (for HSCMU)

Argument	Range Restrictions
freq	1 kHz to 2 MHz (Note 1)

Note 1 See the following description.

The measurement frequency can be set to
1.0, 1.2, 1.5, 2.0, 2.5, 3.0, 4.0, 5.0, 6.0, or 8.0 kHz,
10, 12, 15, 20, 25, 30, 40, 50, 60, or 80 kHz,
100, 120, 150, 200, 250, 300, 400, 500, 600, or 800 kHz,
or 1.0, 1.2, 1.5, or 2.0 MHz.

If *on_off* parameter of pround_cmu is set to 1, you can specify any frequency between 0 Hz and 2.0 MHz and the approximate valid frequency is set as follows:

freq	Set Frequency	freq	Set Frequency
freq ≤ 0.0 Hz	error	40 kHz ≤ freq ≤ 50 kHz	50 kHz
0.0 Hz < freq ≤ 1.0 kHz	1.0 kHz	50 kHz < freq ≤ 60 kHz	60 kHz
1.0 kHz < freq ≤ 1.2 kHz	1.2 kHz	60 kHz < freq ≤ 80 kHz	80 kHz
1.2 kHz < freq ≤ 1.5 kHz	1.5 kHz	80 kHz < freq ≤ 100 kHz	100 kHz
1.5 kHz < freq ≤ 2.0 kHz	2.0 kHz	100 kHz < freq ≤ 120 kHz	120 kHz

freq	Set Frequency	freq	Set Frequency
2.0 kHz < freq ≤ 2.5 kHz	2.5 kHz	120 kHz < freq ≤ 150 kHz	150 kHz
2.5 kHz < freq ≤ 3.0 kHz	3.0 kHz	150 kHz < freq ≤ 200 kHz	200 kHz
3.0 kHz < freq ≤ 4.0 kHz	4.0 kHz	200 kHz < freq ≤ 250 kHz	250 kHz
4.0 kHz < freq ≤ 5.0 kHz	5.0 kHz	250 kHz < freq ≤ 300 kHz	300 kHz
5.0 kHz < freq ≤ 6.0 kHz	6.0 kHz	300 kHz < freq ≤ 400 kHz	400 kHz
6.0 kHz < freq ≤ 8.0 kHz	8.0 kHz	400 kHz < freq ≤ 500 kHz	500 kHz
8.0 kHz < freq ≤ 10 kHz	10 kHz	500 kHz < freq ≤ 600 kHz	600 kHz
10 kHz < freq ≤ 12 kHz	12 kHz	600 kHz < freq ≤ 800 kHz	800 kHz
12 kHz < freq ≤ 15 kHz	15 kHz	800 kHz < freq ≤ 1.0 MHz	1.0 MHz
15 kHz < freq ≤ 20 kHz	20 kHz	1.0 MHz < freq ≤ 1.2 MHz	1.2 MHz
20 kHz < freq ≤ 25 kHz	25 kHz	1.2 MHz < freq ≤ 1.5 MHz	1.5 MHz
25 kHz < freq ≤ 30 kHz	30 kHz	1.5 MHz < freq ≤ 2.0 MHz	2.0 MHz
30 kHz < freq ≤ 40 kHz	40 kHz	2.0 MHz < freq	error

NOTE

Measurement Accuracy Specifications

Actually, you can set *freq* to any valid frequency from 1 kHz through 2 MHz without error. But the 4080 guarantees the measurement accuracy only for measurement frequencies at 1 kHz, 10 kHz, 100 kHz, and 1 MHz, and test signal level at 30 mVrms.

The `frequency(`*freq*`)` function is equivalent to the `set_freq(CMH,`*freq*`)` function.

Argument (for CMU)

Argument	Range Restrictions
freq	Measurement frequency of the CMU. You can specify the following measurement frequencies: 1E3, 1E4, 1E5, or 1E6 Hz.

You can specify 1 kHz, 10 kHz, 100 kHz, or 1 MHz.

NOTE

Measurement Accuracy Specifications

Actually, you can set *frequency* to any frequency from 20 Hz through 1 MHz without error.

But the 4080 guarantees the measurement accuracy only for frequencies 1 kHz, 10 kHz, 100 kHz, and 1 MHz, and output level 30 mVrms.

The `frequency(`*freq*`)` function is equivalent to the `set_freq(CMH,`*freq*`)` function.

ftoim

Revision	D.05.10 or later
Control	n.a.

This function returns a complex value that has the specified real part.

Synopsis

```
COMPLEX ftoim (double real_value);
```

Argument	Range Restrictions/Description
real_value	Real value converted to a complex value

G

This section describes the C functions that begin with G.

get_freq

Revision	C.04.00 or later
Control	HSCMU

This function gets the measurement frequency of capacitance measurement set by the set_freq or frequency function previously executed.

Synopsis

```
int get_freq (double *freq);
```

Argument	Range Restrictions/Description
freq	Pointer to double variable in which to return the measurement frequency set by set_freq function previously executed

If the *on_off* parameter of pround_cmu is set to 1, the actual measurement frequency may differ from the value specified by set_freq or frequency function.

The actually-set measurement frequency set by the previously-executed set_freq or frequency function is return to the *freq*.

See Also

- frequency
- set_freq
- pround_cmu

get_freq_step

Revision	C.04.00 or later
Control	HSCMU

This function gets the number of frequency steps of Z/θ-f sweep measurement set by the set_zf function previously executed.

Synopsis

```
int get_freq_step (int *freq_steps);
```

Argument	Range Restrictions/Description
freq_steps	Pointer to int variable in which to return the number of frequency steps of Z/θ-f sweep measurement set by the previously-executed set_zf function

See Also

- set_zf
- sweep_zf

get_modelname

Revision	D.05.10 or later
Control	–

This function returns the test system's model name.

This function may be useful when making a universal measurement program for Keysight 4070/4080 series test systems (including 4072B/4073B).

Synopsis

```
int get_modelname (int *model);
```

Argument	Range Restrictions/Description
model	Specify the pointer which is used to return the test system's model name. You can use the associated macros in your program to determine which model name was returned: MODEL_4071A (=1) MODEL_4072A (=2) MODEL_4073A (=3) MODEL_4072B (=4) MODEL_4073B (=5) MODEL_4075 (=6) MODEL_4076 (=7) MODEL_4082A (=8) MODEL_4082F (=9) MODEL_4083A (=10)

Example

```
int model;
 :
 :
get_modelname(&model);
if (model==MODEL_4083A) {
 :
 :
 }
```

gnd_open_guard

Revision	A.01.00 or later
Control	SWM

This function connects guard terminals of *unused* SWM pins to circuit common.

The guard terminals of used SWM pins (pins that are currently connected to a SWM port) are not affected by this function. But after a pin is disconnected from the SWM port, the guard terminal of pin will be connected or disconnected to circuit common as was specified by this function.

When init_system function is executed, all SWM pins are disconnected from SWM ports. Then, guard terminals of the pins will be connected or disconnected to circuit common as was specified by this function.

Synopsis

```
int gnd_open_guard (int on);
```

Argument	Range Restriction/Description
on	0 : disconnect
	≠0 : connect

H

This section describes the C functions that begin with H.

high_voltage_state

Revision	A.01.00 or later
Control	n.a.

This function specifies a value to indicate whether any *unsupported* instruments are in high voltage state (40 V or more).

If this function specifies 1 (high voltage state), a high voltage event is generated that can be detected by wait_event or report_event function.

Synopsis

```
int high_voltage_state (int flag);
```

Argument	Description/Default	Range Restrictions
flag	integer expression	0 : Not high voltage state.
		1 : high voltage state.

hp_init_system

Revision	A.01.00 or later
Control	SMU, HSCMU, CMU, DVM, PGU, SPGU, RFU, SPA, SWM

This function is equivalent to init_system function, and you can also use this function if you want to initialize only the HSCMU or CMU.

Synopsis

```
int hp_init_system (int device);
```

Argument	Range Restriction/Description
device	3 : HSCMU or CMU
	≠3 : equivalent to init_system.

hp_init_system(3) clears all previous settings, and sets the HSCMU or CMU as follows:

Table 2-9 HSCMU Settings

Measurement range	AUTO
Signal level	30 mV_{rms}
DC bias	0 V
Frequency	1 MHz

Table 2-10 **CMU Settings**

Measurement range	AUTO
Signal level	30 mV_{rms}
DC bias	0 V
Frequency	1 MHz
Open correction	Enabled
Short correction	Enabled

hp_init_system(3) also clears the measurement parameters set by the following functions:

- set_cmu, set_cmu84
- set_cv, set_cvf, set_zf, set_cv84
- set_freq, frequency

If a value other than 3 is specified for *device*, hp_init_system is equivalent to init_system.

Example

```
if (hp_init_system(3) == -1) error_rep();
```

See Also init_system

htos

Revision	D.05.10 or later
Control	n.a.

This function transforms H parameters into S parameters.

The following equation is used for the H-to-S transformation:

$$\begin{bmatrix} h11 & h12 \\ h21 & h22 \end{bmatrix} = \begin{bmatrix} \frac{H11}{Z0} & H12 \\ H21 & Z0 \times H22 \end{bmatrix}$$

$$\begin{bmatrix} S11 & S12 \\ S21 & S22 \end{bmatrix} = \begin{bmatrix} \frac{(h11-1)(h22+1)-h12\times h21}{(h11+1)(h22+1)-h12\times h21} & \frac{2\times h12}{(h11+1)(h22+1)-h12\times h21} \\ \frac{-2\times h21}{(h11+1)(h22+1)-h12\times h21} & \frac{(1+h11)(1-h22)+h12\times h21}{(h11+1)(h22+1)-h12\times h21} \end{bmatrix}$$

Synopsis

```
void htos (int num, COMPLEX h11[], COMPLEX h21[], COMPLEX h12[], COMPLEX
           h22[], COMPLEX s11[], COMPLEX s21[], COMPLEX s12[], COMPLEX
           s22[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
h11	Array containing the H11 parameters.
h21	Array containing the H21 parameters.
h12	Array containing the H12 parameters.
h22	Array containing the H22 parameters.
s11	Return array for S11 parameters transformed from H parameters.
s21	Return array for S21 parameters transformed from H parameters.

Argument	Range Restrictions/Description
s12	Return array for S12 parameters transformed from H parameters.
s22	Return array for S22 parameters transformed from H parameters.
z0	Characteristic impedance Z0.

If the absolute value of the denominator in the above equation [(h11+1)×(h22+1)–h12×h21] is equal to 0 (zero) or if Z0 is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to S parameters.

See Also rfs_z0_r

htoy

Revision	D.05.10 or later
Control	n.a.

This function transforms H parameters into Y parameters.

The following equation is used for the H-to-Y transformation:

$$\begin{bmatrix} Y11 & Y12 \\ Y21 & Y22 \end{bmatrix} = \begin{bmatrix} \frac{1}{H11} & \frac{-H12}{H11} \\ \frac{H21}{H11} & \frac{H11 \times H22 - H12 \times H21}{H11} \end{bmatrix}$$

Synopsis

```
void htoy (int num, COMPLEX h11[], COMPLEX h21[], COMPLEX h12[], COMPLEX
           h22[], COMPLEX y11[], COMPLEX y21[], COMPLEX y12[], COMPLEX
           y22[]);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
h11	Array containing the H11 parameters.
h21	Array containing the H21 parameters.
h12	Array containing the H12 parameters.
h22	Array containing the H22 parameters.
y11	Return array for Y11 parameters transformed from H parameters.
y21	Return array for Y21 parameters transformed from H parameters.
y12	Return array for Y12 parameters transformed from H parameters.
y22	Return array for Y22 parameters transformed from H parameters.

If the absolute value of the denominator in the above equation [H11] is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the Y parameters.

htoz

Revision	D.05.10 or later
Control	n.a.

This function transforms H parameters into Z parameters.

The following equation is used for the H-to-Z transformation:

$$\begin{bmatrix} Z11 & Z12 \\ Z21 & Z22 \end{bmatrix} = \begin{bmatrix} \frac{H11\times H22 - H12\times H21}{H22} & \frac{H12}{H22} \\ \frac{-H21}{H22} & \frac{1}{H22} \end{bmatrix}$$

Synopsis

```
void htoz (int num, COMPLEX h11[], COMPLEX h21[], COMPLEX h12[], COMPLEX
           h22[], COMPLEX z11[], COMPLEX z21[], COMPLEX z12[], COMPLEX
           z22[]);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
h11	Array containing the H11 parameters.
h21	Array containing the H21 parameters.
h12	Array containing the H12 parameters.
h22	Array containing the H22 parameters.
z11	Return array for Z11 parameters transformed from H parameters.
z21	Return array for Z21 parameters transformed from H parameters.

Argument	Range Restrictions/Description
z12	Return array for Z12 parameters transformed from H parameters.
z22	Return array for Z22 parameters transformed from H parameters.

If the absolute value of the denominator in the above equation [H22] is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the Z parameters.

I

This section describes the C functions that begin with I.

imabs

Revision	D.05.10 or later
Control	n.a.

This function returns the absolute value of the specified complex parameter.

Synopsis

```
double imabs (COMPLEX comp_value);
```

Argument	Range Restrictions/Description
comp_value	Complex value to be calculated

imadd

Revision	D.05.10 or later
Control	n.a.

This function returns a complex value derived by adding the complex parameters.

Synopsis

```
COMPLEX imdd (COMPLEX complex1, COMPLEX complex2);
COMPLEX imsum (COMPLEX complex1, COMPLEX complex2);
```

NOTE

`imsum` is a macro defined as follows:

```
#define imsum( x, y ) imadd( x, y )
```

Argument	Range Restrictions/Description
complex1	Complex value to be added
complex2	Complex value to be added

imaginary

Revision	D.05.10 or later
Control	n.a.

This function returns the imaginary part of the specified parameter.

Synopsis

```
double imaginary (COMPLEX comp_value);
double imag (COMPLEX comp_value);
```

NOTE

`imag` is a macro defined as follows:

```
#define imag( x ) imaginary( x )
```

Argument	Range Restrictions/Description
comp_value	Complex value to be calculated

imargument

Revision	D.05.10 or later
Control	n.a.

This function returns the theta for when the specified parameter is shown in the polar chart. The unit of measure is radian.

Synopsis

```
double imargument (COMPLEX comp_value);
double imarg (COMPLEX comp_value);
```

NOTE

`imarg` is a macro defined as follows:

```
#define imarg( x ) imargument( x )
```

Argument	Range Restrictions/Description
comp_value	Complex value to be calculated

When 0+0j is specified for the comp_value, 0 (zero) is returned.

imdiv

Revision	D.05.10 or later
Control	n.a.

This function returns a complex value derived by dividing *complex1* by *complex2*.

Synopsis

```
COMPLEX imdiv (COMPLEX complex1,COMPLEX complex2);
```

Argument	Range Restrictions/Description
complex1	Complex value used as the numerator
complex2	Complex value used as the denominator

The absolute value of the denominator *complex2* must not be zero.

imexp

Revision	D.05.10 or later
Control	n.a.

This function raises e to the power of the specified complex parameter.

Synopsis

```
COMPLEX imexp (COMPLEX comp_value);
```

Argument	Range Restrictions/Description
comp_value	Complex value to be calculated

imln

Revision	D.05.10 or later
Control	n.a.

This function returns the natural logarithm (base e) of the specified complex parameter.

Synopsis

```
COMPLEX imln (COMPLEX comp_value);
```

Argument	Range Restrictions/Description
comp_value	Complex value to be calculated

imlog10

Revision	D.05.10 or later
Control	n.a.

This function returns the common logarithm (base 10) of the specified complex parameter.

Synopsis

```
COMPLEX imlog10 (COMPLEX comp_value);
COMPLEX imlog (COMPLEX comp_value);
```

NOTE

`imlog` is a macro defined as follows:

```
#define imlog( x ) imlog10( x )
```

Argument	Range Restrictions/Description
comp_value	Complex value to be calculated

immultiply

Revision	D.05.10 or later
Control	n.a.

This function returns a complex value derived by multiplying the specified parameters.

Synopsis

```
COMPLEX immultiply (COMPLEX complex1,COMPLEX complex2);
COMPLEX improduct (COMPLEX complex1,COMPLEX complex2);
```

NOTE

`improduct` is a macro defined as follows:

```
#define improduct( x, y ) immultiply( x, y )
```

Argument	Range Restrictions/Description
complex1	Complex value to be multiplied
complex2	Complex value to be multiplied

When complex values a ($=x_1+jy_1$) and b ($=x_2+jy_2$) are multiplied,

$$a \times b = (x_1+jy_1) \times (x_2+jy_2)$$
$$= x_1 \times x_2 - y_1 \times y_2 + j(x_1 \times y_2 + x_2 \times y_1)$$

imreal

Revision	D.05.10 or later
Control	n.a.

This function returns the real part of the specified parameter.

Synopsis

```
double imreal (COMPLEX comp_value);
double real (COMPLEX comp_value);
```

NOTE

`real` is a macro defined as follows:

```
#define real( x ) imreal( x )
```

Argument	Range Restrictions/Description
comp_value	Complex value to be calculated

imsqrt

Revision	D.05.10 or later
Control	n.a.

This function returns the square root of the specified complex parameter.

Synopsis

```
COMPLEX imsqrt (COMPLEX comp_value);
```

Argument	Range Restrictions/Description
comp_value	Complex value to be calculated

impower

Revision	D.05.10 or later
Control	n.a.

This function returns the *number*-th power of a complex value *complex1*.

Synopsis

```
COMPLEX impower (COMPLEX complex1, double number);
COMPLEX impow (COMPLEX complex1, double number);
```

NOTE

`impow` is a macro defined as follows:

```
#define impow( x, n ) impower( x, n )
```

Argument	Range Restrictions/Description
complex1	Base complex value
number	power factor of *complex1*

imsub

Revision	D.05.10 or later
Control	n.a.

This function returns a complex value derived by subtracting *complex2* from *complex1*.

Synopsis

```
COMPLEX imsub (COMPLEX complex1,COMPLEX complex2);
```

Argument	Range Restrictions/Description
complex1	Complex value to be subtracted from
complex2	Complex value to subtract

init_spa

Revision	D.05.16 or later
Control	(SPA)

This is a dummy function that performs nothing for the spectrum analyzer. This function is provided so that exiting ring oscillator measurement programs that use 4070 can run on 4080 without modification. Do not use this function to create new measurement programs for 4080.

When a test session is started, the GPIB address of the spectrum analyzer is set by the system software, and the initialization is performed by the init_system function.

Synopsis

```
#include "/opt/hp4070/include/spa.h"
   int init_spa (int address);
```

Argument	Range Restrictions/Description
address	dummy parameter

See Also

- init_system

init_system

Revision	A.01.00 or later
Control	SMU, HSCMU, CMU, DVM, PGU, SPGU, RFU, SPA, SWM

This function initializes the system (excluding the wafer prober).

Synopsis

```
int init_system (void);
```

Execute init_system before executing any other TIS function. The init_system function clears all previous settings and sets the main functions of the system (excluding the wafer prober) as follows:

Table 2-11 SWM Settings

	Force	Sense	Guard
Pin board	Open	Open	Open or circuit common (Note 1)
Chuck connection	Open	Open	Open or circuit common (Note 1)
SMU 1 or SMU2 port	Open		Open
SMU 3 to SMU8 port	Open	Open	Open
GNDU port	Open	Open	
AUX 1 or AUX2 port	Open		Open
AUX 3 to AUX8 port	Open	Open	Open

Note 1 If optimization level is 0 or 1: Open.
If optimization level is 2 or 3: depends on setting of previously executed gnd_open_guard function.

Table 2-12 DVM Settings

Measurement range	AUTO
Integration time	1 PLC
Auto zero	Enabled

Table 2-13 SMU Settings

Mode	VS (Voltage Source/Current Monitor)
V output	0 V at 20 V range
I compliance	100 μA at 100 μA range
A/D converter selection	Depends on the optimization level setting (Note 1)
Integration time	SHORT
Auto zero	Depends on the optimization level setting (Note 2)
SMU filter	Depends on the optimization level setting (Note 3)

Note 1 If optimization level is 0 or 1, high-resolution ADC is selected.
If optimization level is 2 or 3, high-speed ADC is selected.

Note 2 If optimization level is 0 or 1, auto zero function is enabled.
If optimization level is 2 or 3, auto zero function is disabled.

Note 3 If optimization level is 0 or 1, SMU filter is enabled.
If optimization level is 2 or 3, SMU filter is disabled.

Table 2-14 HSCMU Settings

Measurement range	AUTO
Signal level	30 mV_{rms}
DC bias	0 V
Frequency	1 MHz

Table 2-15 **CMU Settings**

Measurement range	AUTO
Signal level	30 mV_{rms}
DC bias	0 V
Frequency	1 MHz
Open correction	Enabled
Short correction	Enabled

Table 2-16 **PGU Settings (81150A)**

Output relay	Open
Output mode	2-level Pulse Output
Pulse base	0 V (Note 1)
Pulse amplitude	0.2 V (Note 2)
Pulse width	100 ns
Delay time	0 s
Leading-edge transition time	20 ns
Trailing-edge transition time	20 ns
Load Impedance	1 MΩ

Note 1 If connected to PSC, 5.0 V.

Note 2 If connected to PSC, –5.0 V.

Table 2-17 **SPGU Settings**

Operation mode	PG mode
Number of period (Note 1)	1
Period	1 μs
Output relay	Open
Output mode	2-level Pulse Output
Pulse base	0 V (Note 2)
Pulse amplitude	0.2 V (Note 3)
Pulse width	100 ns
Delay time	0 s
Leading-edge transition time	20 ns
Trailing-edge transition time	20 ns
Load Impedance	1 MΩ

Note 1 Refer to force_pg, start_pg, and spgu_alwg_force.

Note 2 If connected to PSC, 5.0 V.

Note 3 If connected to PSC, –5.0 V.

Table 2-18 **RFU Settings**

RF signal output	Disabled
DC bias source mode	VS (Voltage Source/Current Monitor)
DC bias V output	0 V at 20 V range
DC bias I compliance	100 mA at 100 mA range
DC bias SMU filter	Disabled

Table 2-19 **SPA Settings**

Continuous trigger system	OFF
Numeric data format	Binary Real 64
Byte order for data transfer	Normal

The init_system function clears the following: measurement parameters set by p_set_post_mode, p_set_stop_mode, set_bsearch, set_cmu, set_cmu84, set_cv, set_cv84, frequency, set_freq, set_iv, p_set_iv, set_lsearch, p_set_lsearch, set_pbias, set_piv, set_sync, p_set_sync, set_bdv, set_ileak, set_vth, p_set_vth, stand_by_port, set_type_pg, set_level_pg, set_time_pg, set_mode_pg, spgu_mode, rfs_set_s, rfs_set_cmu, rfs_set_average, rfs_set_cv, and rfs_z0.

Example

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

See Also

- hp_init_system

is_on_line

Revision	A.01.00 or later
Control	n.a.

This function returns status of whether or not the test session is online.

Synopsis

```
int is_on_line (void);
```

If test session is online, this function returns 1.
If not online, this function returns 0.

ivt_measure

Revision	D.05.13 or later
Control	SMU

This function performs a sampling measurement according to the condition set by the ivt_set_t and ivt_set_iv functions, and returns the measurement result data.

Synopsis

```
int ivt_measure (int port,int mode,double range,double measure[],int
                 status[],double tstamp[]);
```

Argument	Range Restrictions/Description	
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008)
	pin number	1 to 49
mode	Specify whether to perform a voltage or current measurement for sampling. Use one of the following macros: V_MEAS (=1): Voltage measurement I_MEAS (=2): Current measurement	
range	Specify the measurement ranging mode. 0.0 for Auto-ranging mode or	
	For voltage measurements [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current measurement [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)
measure	Pointer to double array where to return the measured values.	

Argument	Range Restrictions/Description
status	Pointer to integer array where to return the measurement statuses for measured values.
tstamp	Pointer to double array where to return the time stamp values for measured values.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port address* or *pin number*

  To specify the SMU, use either the *port address* of SMU, or the *pin number* of any measurement pin connected to the SMU port.

- *mode*

  The *mode* determines the current sampling measurement or voltage sampling measurement.

- *range*

  The *range* determines the measurement ranging mode when the sampling interval (*interval* set by ivt_set_t) ≥ 2 ms as follows:

  ***range* = 0.0** auto-ranging is used, which performs measurement in range that gives the best resolution.

  ***range* ≠ 0.0** limited auto-ranging is used. The specified range is used if possible, but changes to a higher range if not possible.

  If sampling interval (*interval* set by ivt_set_t) < 2 ms, *range* is ignored and the measurement range is fixed to the compliance setting set by ivt_set_iv.

NOTE

Type of ADC and integration time to be used

- If sampling interval (*interval* set by ivt_set_t) ≥ 2 ms:

  according to the setting by set_adc, set_adc_i, or set_adc_v, and set_smu_ch functions.

- If sampling interval (*interval* set by ivt_set_t) < 2 ms:

  The high-speed ADC is used and the integration time may be changed from the setting by set_adc, set_adc_i, or set_adc_v.

- *measure*, *status*, *tstamp*

  The measured values are returned to *measure* array, the measurement status values are returned to *status* array, and the time stamp values are returned to *tstamp* array.

  The number of elements of these arrays should be equal to the number of sampling.

In Offline mode 0.0 is returned to *measure* array as the measured values.

The timestamp values calculated from the sampling interval and sampling mode specified by ivt_set_t are returned to *tstamp* array.

See Also

- ivt_set_t
- ivt_set_iv
- ivt_set_sync
- set_smu_ch
- set_adc, set_adc_v, or set_adc_i

ivt_set_iv

Revision	D.05.13 or later
Control	SMU

This function sets the output parameters of the synchronous bias source used for the sampling measurement performed by the ivt_measure function.

Synopsis

```
int ivt_set_iv (int port,int mode,double range,double bias,double
                base,double compliance,int stop_mode,int post_mode);
```

Argument	Range Restrictions/Description	
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008)
	pin number	1 to 49
mode	Specify whether to force a voltage bias or current bias for sampling. Use one of the following macros: V_SOURCE (=1): Voltage force I_SOURCE (=2): Current force	
range	Specify the voltage or current force range. 0.0 for Auto-ranging mode or	
	For voltage force [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current force [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)

Argument	Range Restrictions/Description	
bias, *base*	Specify the bias and base values of voltage or current being forced.	
	For voltage force [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current force [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)
compliance	Specify the voltage or current compliance of the *port*. REMAIN macro (Note 2) or	
	For voltage force [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current force [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)
stop_mode	Specify whether to continue or stop if abnormal condition of SMU is detected. Choose from the following macros: COMP_CONT (=1) COMP_STOP (=2)	
post_mode (Note 3)	Specify the bias output after the sampling measurement. 1: Set to the measurement start state. 3: Set to the bias voltage or current.	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1.
REMAIN means the previous compliance setting if source mode of SMU is same as the previous mode. Otherwise:
If source mode was VS mode and is now IS mode, compliance is set to 20 V.
If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

Note 3 Effective on software revision E.05.14 or later.

- *port address*, *pin number*

  To specify the SMU, use either the *port address* of SMU, or the *pin number* of any measurement pin connected to the SMU port.

- *bias*, *base*

  The *bias* determines the bias voltage/current forced from SMU.

  The *base* determines the base voltage/current that forced before bias voltage/current.

- *compliance*

  The *compliance* determines the compliance current/voltage setting of SMU.

- *stop_mode*

  If *stop_mode* is set to 1, the sampling measurement continues, even if an abnormal status of SMU is detected.

  If *stop_mode* is set to 2, the sampling measurement aborts and returns the dummy data if an abnormal status of SMU is detected.

- *post mode*

  If *post mode* is set to 1, the channels that have finished measurements are set to the initial measurement start state.

  If *post mode* is set to 3, the channels keep the bias output after the sampling measurement.

See Also

- ivt_set_t
- ivt_set_sync
- ivt_measure
- set_smu_ch
- set_adc, set_adc_v, or set_adc_i

ivt_set_sync

Revision	E.05.14 or later
Control	SMU

This function sets the output parameters of the bias source synchronized with the synchronous bias source set by the ivt_set_iv function.

The ivt_set_iv function must be executed *before* this function. This function clears the previous ivt_set_sync setting.

Synopsis

```
int ivt_set_sync (int port,int mode,double range,double bias,double
                  base,double compliance);
```

Argument	Range Restrictions/Description	
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008)
	pin number	1 to 49
mode	Specify whether to force a voltage bias or current bias for sampling. Use one of the following macros: V_SOURCE (=1): Voltage force I_SOURCE (=2): Current force	
range	Specify the voltage or current force range. 0.0 for Auto-ranging mode or	
	For voltage force [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current force [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)

Argument	Range Restrictions/Description	
bias, *base*	Specify the bias and base values of voltage or current being forced.	
	For voltage force [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current force [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)
compliance	Specify the voltage or current compliance of the *port*. REMAIN macro (Note 2) or	
	For voltage force [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current force [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1.
REMAIN means the previous compliance setting if source mode of SMU is same as the previous mode. Otherwise:
If source mode was VS mode and is now IS mode, compliance is set to 20 V.
If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

- *port address*, *pin number*

  To specify the SMU, use either the *port address* of SMU, or the *pin number* of any measurement pin connected to the SMU port.

- *bias*, *base*

  The *bias* determines the bias voltage/current forced from SMU.

  The *base* determines the base voltage/current that forced before bias voltage/current.

- *compliance*

  The *compliance* determines the compliance current/voltage setting of SMU.

See Also

- ivt_set_t, ivt_set_iv, ivt_measure
- set_smu_ch
- set_adc, set_adc_v, or set_adc_i

ivt_set_t

Revision	D.05.13 or later
Control	SMU

This function sets the timing parameters for the sampling measurement.

Synopsis

```
int ivt_set_t (int mode,double h_bias,double interval,int number,double
               h_base);
```

Argument	Range Restrictions/Description	
mode	Specify the sampling mode.	
		1: linear 2: log, 10 data/decade 3: log, 25 data/decade 4: log, 50 data/decade 5: log, 100 data/decade 6: log, 250 data/decade 7: log, 500 data/decade
h_bias	Specify the hold time before forcing bias voltage or current. unit: [s]	
		0 to 655.350000 (resolution: 10E–3) –90E-3 to –100E–6 (Note 1) (resolution: 100E-6)
interval	Specify the sampling interval. [s] (resolution: 10E-6)	
		100E-6 + 50E-6 × (number of measurement channels – 1) to 65.535
number	Specify the total number of samples.	
	if *mode* = 1 (linear sampling):	1 to 100001
	if *mode* ≠ 1 (log sampling):	1 to 11 × (number of data per decade) + 1

Argument	Range Restrictions/Description
h_base	Specify the hold time before forcing base voltage or current. unit: [s]
	0 to 655.350000 (resolution: 10E–3)

Note 1 *interval* must be less than 2 ms.

- *mode*

  The *smode* determines whether the sampling is linear or logarithmic, and number of sampling per decade if logarithmic sampling.

- *h_bias*, *h_base*

  The *h_bias* determines the hold time between the start of forcing bias voltage/current and the start of sampling.

  The *h_base* determines the hold time between the start of forcing base voltage/current and the start of forcing bias voltage/current.

  If *interval* is less than 2 ms, the sampling can be started before start of forcing bias voltage/current by specifying the negative value for the *h_bias* parameter.

- *interval*

  The interval of sampling points in second.

- *number*

  Specifies the total number of sampling points.

See Also

- ivt_set_iv
- ivt_set_sync
- ivt_measure
- set_smu_ch
- set_adc, set_adc_v, or set_adc_i

L

This section describes the C functions that begin with L.

load_corr_data

Revision	A.01.00 or later
Control	HSCMU, CMU

This function loads compensation data for capacitance measurement, which is measured by open_corr_th and short_corr_th, from the specified file.

Synopsis

```
int load_corr_data (char *file_specifier);
```

Argument	Range Restrictions/Description
file_specifier	Pointer to a character string that specifies the name of a file where the compensation data is stored and may include an absolute path name. If 0 is specified, all current compensation data is cleared. Maximum length: 1023 characters.

This function loads OPEN and SHORT compensation data, which are stored by store_corr_data function, beyond switching matrix measurement pins, for example up to the probe needles, from the specified file.

After loading compensation data, the 4080 automatically performs error compensation for capacitance measurement.

The loaded compensation data is effective until next execution of this function.

Example

```
load_corr_data("c-corr");
```

See Also

- store_corr_data
- open_corr_th

- short_corr_th
- disable_corr

load_hfportmap

Revision	B.02.00 or later
Control	SMU, HSCMU, CMU, DVM, PGU, SPGU, RFU, SPA, SWM

This function loads the port map file for the HF port and initializes the system.

Synopsis

```
int load_hfportmap (char *file_specifier);
```

Argument	Range Restriction/Description
file_specifier	Pointer to a character string that specifies the name of a file where the HF port map data is stored and may include an absolute path name. If NULL is specified, default file (/etc/opt/hp4070/hfportmap) is loaded. Maximum length: 1023 characters.

This function loads the port mapping information for the HF ports and then initializes the system.

load_pg

Revision	B.02.00 or later
Control	PGU, SPGU

This function loads the pulse generator setting file that was saved using the save_pg function in the measurement program, or that was interactively saved from the Interactive Debugging Panel window.

Synopsis

```
int load_pg (char *file_specifier);
```

Argument	Range Restriction/Description
file_specifier	Pointer to a character string that specifies the name of a file where the pulse generator setting data is stored and may include an absolute path name. Maximum length: 1023 characters.

Using save_pg and load_pg functions may reduce the setup time for pulse generators.

If the pulse generator configuration settings are different from those saved with the save_pg function, an error will occur.

For SPGU, only PG mode is supported.

In the offline mode, do not execute this function. Executing this function causes an error.

See Also

- save_pg
- set_time_pg
- set_level_pg
- prep_pg

load_pg_conf

Revision	B.02.00 or later
Control	SMU, HSCMU, CMU, DVM, PGU, SPGU, RFU, SPA, SWM

This function loads the pulse generator configuration file and initializes the system.

Synopsis

```
int load_pg_conf (char *file_specifier);
```

Argument	Range Restriction/Description
file_specifier	Pointer to a character string that specifies the name of a file where the pulse generator configuration data is stored and may include an absolute path name. If NULL is specified, the default file (/etc/opt/hp4070/pgconnection) is loaded. Maximum length: 1023 characters.

The pulse generator connection information can be interactively saved to a specified file in the Pulse Generator Configuration window.

This function loads the pulse generator configuration information from the specified file and then initializes the system.

long_cal

Revision	A.01.00 or later
Control	SMU, HSCMU, SPGU

This function sets all system hardware resources to their initial status. It also starts the self-calibration function of the SMUs, HSCMU, and SPGUs.

Synopsis

```
int long_cal (void);
```

This function is exactly the same as the short_cal function.

You should perform calibration after your system has fully warmed up to correct for offsets caused by temperature drifts.

For best measurement results, perform calibration when the ambient temperature changes more than 3 °C (6 °F).

Example

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

See Also

- short_cal

M

This section describes the C functions that begin with M.

measure_bdv

Revision	A.01.00 or later
Control	SMU

This function triggers quasi-pulsed measurements to measure breakdown voltage, then returns the breakdown voltage.

The conditions of the quasi-pulsed measurements are determined by the set_bdv function.

Synopsis

```
int measure_bdv (double *voltage, int *status, int interval);
```

Argument	Range Restrictions/Description
voltage	Specify a pointer to double variable in which to return the measured voltage.
status	Specify a pointer to integer variable in which to return the measurement status code. One of following is returned to *status*: BDV_NORMAL (= 0) BDV_COMP_OTHER (= 1) BDV_CURRENT_LOW (= 2) BDV_OSC (= 3) BDV_OVERFLOW (= 4) BDV_TIMEOUT (= 6) BDV_TOO_SLOW (= 7)
interval	Measurement interval. Specify one of the following macros: BDV_INTVL_SHORT (= 0) BDV_INTVL_LONG (= 1)

If this function finishes successfully, a 0 is returned. If not, –1 is returned.

- *voltage*

  The measured value (breakdown voltage) is returned.

- *status*

  One of the following statuses is returned:

Status Code	Condition
BDV_NORMAL	The quasi-pulsed measurement ended normally.
BDV_COMP_OTHER	Another unit reached compliance.
BDV_CURRENT_LOW	Breakdown voltage measurement failed because current did not reach breakdown *current* within the stop voltage specified by set_bdv function.
BDV_OSC	This unit is oscillating.
BDV_OVERFLOW	Measurement overflow occurred while monitoring output voltage or measuring breakdown voltage.
BDV_TIMEOUT	The quasi-pulsed measurement did not reach breakdown *current* within timeout. See below.
BDV_TOO_SLOW	The monitored slew rate of the output voltage is too slow. See below.

- *interval*

  As described for the set_bdv function, the gradient of voltage increase is monitored to determine when breakdown occurs.

  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 (300 μs interval). If the *interval* is set to BDV_INTVL_LONG, the gradient is calculated after every 8 voltage measurements.

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

The timeout is set to 3 seconds for BDV_INTVL_SHORT mode and 12 seconds for BDV_INTVL_LONG mode.

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_cmu84

Revision	A.01.00 or later
Control	CMU, (HSCMU)

NOTE

CMU Control TIS function

This function can be used to execute 4080 measurement program for using CMU on the 4080 with HSCMU.

Do not use this function to make new measurement program for using HSCMU. (Use measure_cpg.)

This function measures capacitance and conductance by using the HSCMU and returns error compensated values.

Synopsis

```
int measure_cmu84 (double *capacitance, double *conductance, double range);
```

Argument	Range Restrictions/Description
capacitance	Pointer to double variable in which to return the measured capacitance.
conductance	Pointer to double variable in which to return the measured conductance. To discard the measured value, specify NULL.
range	Specify the capacitance measurement range. The measure_cmu84 function translates the capacitance range to the impedance range. Specify 0.0 for Auto range mode or a Numeric expression [F]: real value

- *capacitance* and *conductance*

  This function performs a capacitance measurement and returns the error compensated capacitance and conductance values to *capacitance* and *conductance*, respectively.

  If measurement failed, 9.99999E+9 is returned to each variable.

- *range for HSCMU*

  The HSCMU actually performs the impedance measurement and then converts the measured impedance to the capacitance and conductance values.

  If the *range* (capacitance range) other than 0 is specified, the impedance range value (Ir) is calculated from the specified range (Cr) and measurement frequency (fm) as follows:

  $$Ir = 1/(2 \times \pi \times |fm| \times |Cr|)$$

  And an actual impedance range for impedance measurement is determined from the calculated impedance range value (Ir), test signal level (Vosc) and measurement frequency (fm) as shown in following table:

Calculated Ir value	Measurement Frequencies (fm), Test Signal level (Vosc), and Corresponding Impedance Ranges			
	Vosc = 30, 50, 100 mV_{rms}		Vosc = 10 mV_{rms}	
	fm ≤ 100 kHz	120 kHz ≤ fm	fm ≤ 100 kHz	120 kHz ≤ fm
0 ≤ *Ir* ≤ 300 Ω	100 Ω	100 Ω	100 Ω	100 Ω
300 Ω < *Ir* ≤ 1 kΩ	300 Ω	300 Ω	100 Ω	100 Ω
1 kΩ < *Ir* ≤ 3 kΩ	1 kΩ	1 kΩ	1 kΩ	1 kΩ
3 kΩ < *Ir* ≤ 10 kΩ	3 kΩ	3 kΩ	1 kΩ	1 kΩ
10 kΩ < *Ir* ≤ 30 kΩ	10 kΩ	10 kΩ	10 kΩ	10 kΩ
30 kΩ < *Ir* ≤ 100 kΩ	30 kΩ	30 kΩ	10 kΩ	10 kΩ
100 kΩ < *Ir* ≤ 300 kΩ	100 kΩ	30 kΩ	100 kΩ	10 kΩ
300 kΩ < *Ir*	300 kΩ	30 kΩ	100 kΩ	10 kΩ

If *range* parameter is 0, the HSCMU uses the Auto ranging mode, so HSCMU automatically changes to range that gives best resolution.

- *range for CMU*

  The specified *range* (capacitance range) and corresponding impedance ranges are shown in following table:

Allowed *range* in F	Measurement Frequencies and Corresponding Impedance Ranges			
	1 kHz	10 kHz	100 kHz	1 MHz
0 < |*range*| ≤ 1.592E-12	100 kΩ	100 kΩ	100 kΩ	100 kΩ (Note 1)
1.592E-12 < |*range*| ≤ 5.305E-12	100 kΩ	100 kΩ	100 kΩ	30 kΩ (Note 1)
5.305E-12 < |*range*| ≤ 1.592E-11	100 kΩ	100 kΩ	100 kΩ	10 kΩ
1.592E-11 < |*range*| ≤ 5.305E-11	100 kΩ	100 kΩ	30 kΩ	3 kΩ
5.305E-11 < |*range*| ≤ 1.592E-10	100 kΩ	100 kΩ	10 kΩ	1 kΩ
1.592E-10 < |*range*| ≤ 5.305E-10	100 kΩ	30 kΩ	3 kΩ	300 Ω
5.305E-10 < |*range*| ≤ 1.592E-9	100 kΩ	10 kΩ	1 kΩ	100 Ω
1.592E-9 < |*range*| ≤ 5.305E-9	30 kΩ	30 kΩ	300 Ω	100 Ω
5.305E-9 < |*range*| ≤ 1.592E-8	10 kΩ	1 kΩ	100 Ω	100 Ω
1.592E-8 < |*range*| ≤ 5.305E-8	3 kΩ	300 Ω	100 Ω	10 Ω (Note 2)
5.305E-8 < |*range*| ≤ 1.592E-7	1 kΩ	100 Ω	100 Ω	10 Ω (Note 2)
1.592E-7 < |*range*| ≤ 5.305E-7	300 Ω	100 Ω	10 Ω (Note 2)	1 Ω (Note 3)
5.305E-7 < |*range*| ≤ 1.592E-6	100 Ω	100 Ω	10 Ω (Note 2)	1 Ω (Note 3)
1.592E-6 < |*range*| ≤ 1.592E-5	100 Ω	10 Ω (Note 2)	1 Ω (Note 3)	1 Ω (Note 3)

Allowed *range* in F	Measurement Frequencies and Corresponding Impedance Ranges			
	1 kHz	10 kHz	100 kHz	1 MHz
1.592E-5 < |*range*| ≤ 1.592E-4	10 Ω (Note 2)	1 Ω (Note 3)	1 Ω (Note 3)	1 Ω (Note 3)
1.592E-4 < |*range*|	1 Ω (Note 3)	1 Ω (Note 3)	1 Ω (Note 3)	1 Ω (Note 3)

Note 1 If test signal level is 2.0 V or less, impedance range is set to 10 kΩ.

Note 2 If test signal level is 0.1 V or less, impedance range is set to 100 Ω.

Note 3 If test signal level is 2.0 V or less, impedance range is set to 10 Ω. If test signal level is 0.1 V or less, impedance range is set to 100 Ω.

The *range* argument sets the capacitance measurement range (C range) of the CMU. The conductance measurement range (G range) is automatically set to *range* × 1.0×10^7.

If *range* parameter is 0, the CMU uses the Auto ranging mode, so CMU automatically changes to range that gives best resolution.

If non-zero value is specified for the *range* argument, limited auto ranging is used. For limited auto ranging, the specified range is used if possible, but if specified range it too small, the CMU changes up to range that can perform the measurement.

The non-zero value should be a capacitance range value. However, the system converts to an impedance range according to the following equation, and CMU performs an impedance measurement:

$$\text{impedance range} = \frac{1}{2\pi f|range|}$$

Where, *f* is the measurement frequency *range* is specified capacitance range.

Refer to the *LCR Meter Operation Manual* for more information on the impedance range.

Offline Mode In offline mode execution, this function returns data as follows:

range	Returned Capacitance Value	Returned Conductance Value
AUTO	previous C range value default = 10 pF	previous G range value default = 100 μS
other than AUTO	C range value	G range value

Example

```
int err;
double capacitance, conductance;

……

err = force_v(CMH, 5.0, 20.0, 1E-3);
err = measure_cmu84(&capacitance, &conductance, 0.0);
```

See Also

- set_cmu84
- frequency
- force_v

measure_cpg, measure_cpgt

Revision	C.04.00 or later
Control	HSCMU

This function measures capacitance and conductance by using the HSCMU and returns error compensated values.

Synopsis

```
int measure_cpg (double *capacitance, double *conductance, double range);
int measure_cpgt (double *capacitance, double *conductance, double range,
            double *time_stamp);
```

Argument	Range Restrictions/Description
capacitance	Pointer to double variable in which to return the measured capacitance.
conductance	Pointer to double variable in which to return the measured conductance. To discard the measured value, specify NULL.
range	Specify the capacitance measurement range. The measure_cpg or measure_cpgt function translates the capacitance range to the impedance range. Specify 0.0 for Auto range mode or a Numeric expression [F]: real value
time_stamp (Note 1)	Specify a pointer to a double array in which to return the time stamps for measured values.

Note 1 Only for measure_cpgt function

- *capacitance* and *conductance*

  This function performs a capacitance measurement and returns the error compensated capacitance and conductance values to *capacitance* and *conductance*, respectively.

  If measurement failed, 9.99999E+9 is returned to each variable.

- *range*

  The HSCMU actually performs the impedance measurement and then converts the measured impedance to the capacitance and conductance values.

  If the *range* (capacitance range) other than 0 is specified, the impedance range value (Ir) is calculated from the specified range (Cr) and measurement frequency (fm) as follows:

  $$Ir = 1/(2 \times \pi \times |fm| \times |Cr|)$$

  And an actual impedance range for impedance measurement is determined from the calculated impedance range value (Ir), test signal level (Vosc) and measurement frequency (fm) as shown in following table:

Calculated Ir value	Measurement Frequencies (fm), Test Signal level (Vosc), and Corresponding Impedance Ranges			
	Vosc = 30, 50, 100 mV_{rms}		Vosc = 10 mV_{rms}	
	fm ≤ 100 kHz	120 kHz ≤ fm	fm ≤ 100 kHz	120 kHz ≤ fm
0 ≤ *Ir* ≤ 300 Ω	100 Ω	100 Ω	100 Ω	100 Ω
300 Ω < *Ir* ≤ 1 kΩ	300 Ω	300 Ω	100 Ω	100 Ω
1 kΩ < *Ir* ≤ 3 kΩ	1 kΩ	1 kΩ	1 kΩ	1 kΩ
3 kΩ < *Ir* ≤ 10 kΩ	3 kΩ	3 kΩ	1 kΩ	1 kΩ
10 kΩ < *Ir* ≤ 30 kΩ	10 kΩ	10 kΩ	10 kΩ	10 kΩ
30 kΩ < *Ir* ≤ 100 kΩ	30 kΩ	30 kΩ	10 kΩ	10 kΩ
100 kΩ < *Ir* ≤ 300 kΩ	100 kΩ	30 kΩ	100 kΩ	10 kΩ
300 kΩ < *Ir*	300 kΩ	30 kΩ	100 kΩ	10 kΩ

If *range* parameter is 0, the HSCMU uses the Auto ranging mode, so HSCMU automatically changes to range that gives best resolution.

- *time_stamp*

  If you execute measure_cpgt function, the accumulated time from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command to measurement start is returned in seconds to *time_stamp*.

  If reset_timestamp function has been executed after last init_system or /opt/hp4070/bin/hp4070 -start, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode In offline mode execution, this function returns data as follows:

range	Returned Capacitance Value	Returned Conductance Value
AUTO	previous C range value default = 10 pF	previous G range value default = 100 μS
other than AUTO	C range value	G range value

Example

```
int err;
double capacitance, conductance;

……

err = force_v(CMH, 5.0, 20.0, 1E-3);
err = measure_cpg(&capacitance, &conductance, 0.0);
```

See Also

- port_status

measure_csrs, measure_csrst

Revision	C.04.00 or later
Control	HSCMU

This function measures capacitance (Cs) and resistance (Rs) of a series equivalent circuit by using the HSCMU and returns error compensated values.

Synopsis

```
int measure_csrs (double *cs, double *rs, double range);
int measure_csrst (double *cs, double *rs, double range, double
            *time_stamp);
```

Argument	Range Restrictions/Description
cs	Pointer to double variable in which to return the measured capacitance.
rs	Pointer to double variable in which to return the measured resistance. To discard the measured value, specify NULL.
range	Specify the capacitance measurement range. The measure_csrs function translates the capacitance range to the impedance range. Specify 0.0 for Auto range mode or a Numeric expression [F]: real value
time_stamp (Note 1)	Specify a pointer to a double array in which to return the time stamps for measured values.

Note 1 Only for measure_csrst function

- *cs* and *rs*

  This function performs a capacitance measurement and returns the error compensated capacitance and resistance values to *cs* and *rs*, respectively.

  If measurement failed, 9.99999E+9 is returned to each variable.

- *range*

  The HSCMU actually performs the impedance measurement and then converts the measured impedance to the capacitance and conductance values.

  If the *range* (capacitance range) other than 0 is specified, the impedance range value (Ir) is calculated from the specified range (Cr) and measurement frequency (fm) as follows:

  $$Ir = 1/(2 \times \pi \times |fm| \times |Cr|)$$

  And an actual impedance range for impedance measurement is determined from the calculated impedance range value (Ir), test signal level (Vosc) and measurement frequency (fm) as shown in following table:

Calculated Ir value	Measurement Frequencies (fm), Test Signal level (Vosc), and Corresponding Impedance Ranges			
	Vosc = 30, 50, 100 mV_{rms}		Vosc = 10 mV_{rms}	
	fm ≤ 100 kHz	120 kHz ≤ fm	fm ≤ 100 kHz	120 kHz ≤ fm
0 ≤ *Ir* ≤ 300 Ω	100 Ω	100 Ω	100 Ω	100 Ω
300 Ω < *Ir* ≤ 1 kΩ	300 Ω	300 Ω	100 Ω	100 Ω
1 kΩ < *Ir* ≤ 3 kΩ	1 kΩ	1 kΩ	1 kΩ	1 kΩ
3 kΩ < *Ir* ≤ 10 kΩ	3 kΩ	3 kΩ	1 kΩ	1 kΩ
10 kΩ < *Ir* ≤ 30 kΩ	10 kΩ	10 kΩ	10 kΩ	10 kΩ
30 kΩ < *Ir* ≤ 100 kΩ	30 kΩ	30 kΩ	10 kΩ	10 kΩ
100 kΩ < *Ir* ≤ 300 kΩ	100 kΩ	30 kΩ	100 kΩ	10 kΩ
300 kΩ < *Ir*	300 kΩ	30 kΩ	100 kΩ	10 kΩ

If *range* parameter is 0, the HSCMU uses the Auto ranging mode, so HSCMU automatically changes to range that gives best resolution.

- *time_stamp*

  If you execute measure_csrst function, the accumulated time from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command to measurement start is returned in seconds to *time_stamp*.

  If reset_timestamp function has been executed after last init_system or /opt/hp4070/bin/hp4070 -start, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode In offline mode execution, this function returns data as follows:

range	Returned Capacitance Value	Returned Conductance Value
AUTO	previous C range value default = 10 pF	previous G range value default = 100 μS
other than AUTO	C range value	G range value

Example

```
int err;
double capacitance, resistance;

……

err = force_v(CMH, 5.0, 20.0, 1E-3);
err = measure_csrs(&capacitance, &resistance, 0.0);
```

See Also

- port_status

measure_i, measure_it

Revision	A.01.00 or later
Control	SMU, RFU

These functions measure DC current using a specified SMU or DC bias source (SMU) of RFU and returns the measurement value in amperes.

Synopsis

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

Argument		Range Restrictions/Description
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008), RFU11(20301), RFU12(20302).
	pin number	1 to 49 for SMU 101 to 105 for RFU11 201 to 205 for RFU12
current	Pointer to double variable in which to return the measurement value.	

Argument	Range Restrictions/Description	
range	Measurement range. 0.0 for Auto range mode or	
	When using SMU in VS (V_SOURCE) mode or RFU:	–0.1 to 0.1 (MPSMU and HRSMU) –1 to 1 (HPSMU) –0.2 to 0.2 (RFU)
	When using SMU in IS (I_SOURCE) mode:	any value (Note 2)
time_stamp (Note 3)	Pointer to double variable in which to return the time stamp value.	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 This is a dummy parameter. Measurement range is actually set to same as output current range. Refer to force_i function for details about output range.

Note 3 Only for measure_it function. If DC bias source of RFU, measurement timestamp value is not returned.

- *port*

  To specify the SMU, use either the *port address* of SMU or RFU, or the *pin number* of any SWM pin connected to the SMU or RFU port.

- *current*

  The measurement value is returned to *current*.

- *range*

  Specifies the range value as follows:

Table 2-20 Current Measurement Range and Resolution of Returned Measurement Value (MPSMU and HPSMU)

range	Range Value	Resolution of Returned Measurement Value					
		High-speed ADC (Note 1)	High-resolution ADC (Note 1)				
			Integration time *t* (μs)				
			$0 \le t < 320$	$320 \le t < 640$	$640 \le t < 2880$	$2880 \le t < 5680$	$5680 \le t$
0 A < |*range*| ≤ 1.15 nA	1 nA	50 fA	100 fA	20 fA	10 fA	2 fA	1 fA
1.15 nA < |*range*| ≤ 11.5 nA	10 nA	500 fA	1 pA	200 fA	100 fA	20 fA	10 fA
11.5 nA < |*range*| ≤ 115 nA	100 nA	5 pA	10 pA	2 pA	1 pA	200 fA	100 fA
115 nA < |*range*| ≤ 1.15 μA	1 μA	50 pA	100 pA	20 pA	10 pA	2 pA	1 pA
1.15 μA < |*range*| ≤ 11.5 μA	10 μA	500 pA	1 nA	200 pA	100 pA	20 pA	10 pA
11.5 μA < |*range*| ≤ 115 μA	100 μA	5 nA	10 nA	2 nA	1 nA	200 pA	100 pA
115 μA < |*range*| ≤ 1.15 mA	1 mA	50 nA	100 nA	20 nA	10 nA	2 nA	1 nA
1.15 mA < |*range*| ≤ 11.5 mA	10 mA	500 nA	1 μA	200 nA	100 nA	20 nA	10 nA
11.5 mA < |*range*| ≤ 100 mA (for MPSMU)	100 mA	5 μA	10 μA	2 μA	1 μA	200 nA	100 nA
11.5 mA < |*range*| ≤ 115 mA (for HPSMU)	100 mA	5 μA	10 μA	2 μA	1 μA	200 nA	100 nA
115 mA < |*range*| ≤ 1 A (for HPSMU)	1 A	50 μA	100 μA	20 μA	10 μA	2 μA	1 μA

Note 1 You can select ADC by using set_smu_ch function.

Table 2-21 Current Measurement Range and Resolution of Returned Measurement Value (HRSMU)

range	Range Value	Resolution of Returned Measurement Value					
		High-speed ADC (Note 1)	High-resolution ADC (Note 1)				
			Integration time *t* (μs)				
			$0 \le t < 320$	$320 \le t < 640$	$640 \le t < 2880$	$2880 \le t < 5680$	$5680 \le t$
0 A < |*range*| ≤ 10.5 pA	10 pA	500 aA	1 fA	200 aA	100 aA	20 aA	10 aA
10.5 pA < |*range*| ≤ 105 pA	100 pA	5 fA	10 fA	2 fA	1 fA	200 aA	100 aA
105 pA < |*range*| ≤ 1.15 nA	1 nA	50 fA	100 fA	20 fA	10 fA	2 fA	1 fA
1.15 nA < |*range*| ≤ 11.5 nA	10 nA	500 fA	1 pA	200 fA	100 fA	20 fA	10 fA
11.5 nA < |*range*| ≤ 115 nA	100 nA	5 pA	10 pA	2 pA	1 pA	200 fA	100 fA
115 nA < |*range*| ≤ 1.15 μA	1 μA	50 pA	100 pA	20 pA	10 pA	2 pA	1 pA
1.15 μA < |*range*| ≤ 11.5 μA	10 μA	500 pA	1 nA	200 pA	100 pA	20 pA	10 pA
11.5 μA < |*range*| ≤ 115 μA	100 μA	5 nA	10 nA	2 nA	1 nA	200 pA	100 pA
115 μA < |*range*| ≤ 1.15 mA	1 mA	50 nA	100 nA	20 nA	10 nA	2 nA	1 nA
1.15 mA < |*range*| ≤ 11.5 mA	10 mA	500 nA	1 μA	200 nA	100 nA	20 nA	10 nA
11.5 mA < |*range*| ≤ 100 mA	100 mA	5 μA	10 μA	2 μA	1 μA	200 nA	100 nA

Note 1 You can select ADC by using set_smu_ch function.

Table 2-22 Current Measurement Range (RFU)

range	Range Value
0 < |*range*| ≤ 115 nA	100 nA
115 nA < |*range*| ≤ 1.15 μA	1 μA
1.15 μA < |*range*| ≤ 11.5 μA	10 μA
11.5 μA < |*range*| ≤ 115 μA	100 μA
115 μA < |*range*| ≤ 1.15 mA	1 mA
1.15 mA < |*range*| ≤ 11.5 mA	10 mA
11.5 mA < |*range*| ≤ 115 mA	100 mA
115 mA < |*range*| ≤ 200 mA	200 mA

- For SMU in the VS (V_SOURCE) mode or RFU:

  If you specify 0 for the *range*, auto-ranging is used, which performs measurement in range that gives the best resolution.

  If you specify a non-zero value for the *range*, limited auto-ranging is used. The specified range is used if possible, but changes to a higher range if not possible.

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

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

- For SMU in the IS (I_SOURCE) mode:

  range has no meaning and the measurement range is same as the output current range. Refer to force_i function for details about output range.

- *time_stamp*

  If you execute measure_it function, the accumulated time from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command to measurement start is returned in seconds to *time_stamp*.

If reset_timestamp function has been executed after last init_system or /opt/hp4070/bin/hp4070 -start, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode This function returns simulated measurement values to the *current* argument as shown in the following table. Returned values are determined by the *range* value or present source mode of the SMU.

Source Mode	*range*	Returned Value	Polarity
VS	0	I compliance	same as SMU V output value
	≠0	range value	
	omitted	last valid range value	
IS	n.a.	SMU I output value	same as SMU I output value

Example

```
if (measure_i(12, &current, 1E-3) == -1) error_rep();
```

See Also

- force_meas
- force_i
- measure_v

measure_ileak

Revision	A.01.00 or later
Control	SMU

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.

Synopsis

```
int measure_ileak (int port, double *current, int *status, int interval);
```

Argument	Range Restrictions/Description	
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address:	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number:	1 to 49
current	Pointer to double variable in which to return the measured current.	
status	Pointer to integer variable in which to return the measurement status. One of following is returned: ILEAK_NORMAL (= 0) ILEAK_COMP_OTHER (= 1) ILEAK_COMP (= 2) ILEAK_OSC (= 3) ILEAK_OVERFLOW (= 4) ILEAK_TIMEOUT (= 6) ILEAK_TOO_SLOW(= 7)	
interval	Measurement interval. Specify one of the following macros: ILEAK_INTVL_SHORT (= 0) ILEAK_INTVL_LONG (= 1)	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port*

  To specify the SMU, use either the *port address* of SMU, or the *pin number* of any SWM pin connected to the SMU port.

- *current* and *status*

  The measurement value is returned to the *current* variable.

  The status value is returned to the *status* variable as follows:

Status Code	Condition
ILEAK_NORMAL	The quasi-pulsed measurement ended normally.
ILEAK_COMP_OTHER	Another unit reached compliance.
ILEAK_COMP	This unit reached compliance.
ILEAK_OSC	This unit is oscillating.
ILEAK_OVERFLOW	Measurement overflow occurred while monitoring the output voltage or measuring the leakage current.
ILEAK_TIMEOUT	The quasi-pulsed measurement did not reach *output* (specified by set_ileak) within the timeout. See below.
ILEAK_TOO_SLOW	The monitored slew rate of the output voltage is too slow. See below.

- *interval* As described for the set_ileak function, the gradient of voltage increase is monitored to determine when to measure the current.

  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 (300 μs interval). If the *interval* is set to ILEAK_INTVL_LONG, the gradient is calculated after every 8 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 timeout is set to 3 seconds for ILEAK_INTVL_SHORT mode and 12 seconds for ILEAK_INTVL_LONG mode.

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

Revision	A.01.00 or later
Control	SMU

This function performs multi-channel DC voltage or current measurements using up to eight SMU ports. SMU measurements will be performed sequentially in the specified port order.

This function *cannot* set the measurement mode, so you should use force_i or force_v functions to set the desired measurement mode for each SMU port before measure_m function.

The force_i function sets the SMU port to voltage measurement mode, and force_v sets the SMU port to current measurement mode.

Synopsis

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

Argument	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 ports. This array must have *n* elements. The array should contain the SMU port addresses or SWM pin numbers that are connected to ports.
meas_vals	Specify a pointer to double array in which to return the measured values. This array must have *n* elements.
ranges	Specify a pointer to double array that contains the measurement range for each port. This array must have *n* elements.
time_stamps	Specify a pointer to a double array in which to return the time stamps for measured values.

- *n*

  This function can perform up to 8 simultaneous measurements. *n* argument specifies how many ports will perform measurement. You must declare array arguments for this function as in the following example:

  ```
  int n=3, ports[3];
  double meas_vals[3], ranges[3], time_stamps[3];
  ```

- *port*

  Pointer to an integer array that determines the measurement ports. This array must have *n* elements.

  The array elements can contain the SMU port addresses or SWM pin numbers that are connected to ports as follows:

  port address: You can specify the following macros (or values):

  `SMU1` (20001), `SMU2` (20002), `SMU3` (20003), `SMU4` (20004), `SMU5` (20005), `SMU6` (20006), `SMU7` (20007), `SMU8` (20008).

  pin number: 1 to 49

- *meas_vals*

  Measurement results are returned to *meas_vals* array in order that corresponds to elements of the *port* array.

- *ranges*

  You should specify values in *ranges* array in order that corresponds to elements of the *ports* array.

  Specify range values according to the measurement modes of the corresponding SMUs. Refer to the measure_i or measure_v for measurement range values.

- *time_stamps*

  This function returns accumulated time at measurement start for each SMU in seconds (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) to the *time_stamps* array.

  If reset_timestamp function has been executed after last init_system or /opt/hp4070/bin/hp4070 -start, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode See measure_v or measure_i.

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

- force_i
- force_v
- measure_it
- measure_vt

measure_p

Revision	A.01.00 or later
Control	SMU

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

Argument	Range Restrictions/Description	
port	Specify the SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)):
		SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
mode	Specify whether to perform a voltage or current measurement. Use one of the following macros: V_MEAS (= 1) I_MEAS (= 2)	
value	Specify a pointer to a double variable in which to return the measurement result.	
range	Specify the measurement range.	
	For voltage measurements [V]:	–100 to 100 –200 to 200 (HPSMU) If 0.0 is specified, voltage compliance setting of SMU measure port is used.
	For current measurements [A]:	–0.1 to 0.1 –1 to 1 (HPSMU) If 0.0 is specified, current compliance setting of SMU measure port is used.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port*

  To specify the SMU, use either the *port address* of SMU, or the *pin number* of any SWM pin connected to the SMU port.

  If you specify 0, no measurement is executed and this function just triggers to force the pulsed bias that is set by set_pbias function.

- *range*

  range specifies the range value as follows:

  - For SMU in the V_MEAS mode: See Table 2-23.

    Always uses high-speed ADC and number of sample for digital integration is set to 1. The resolution of returned value is (range value)/20000.

  - For SMU in the I_MEAS mode: See Table 2-24.

    Always uses high-speed ADC and number of sample for digital integration is set to 1. The resolution of returned value is (range value)/20000.

Table 2-23 Voltage Measurement Range

range	Range Value
0 V < |*range*| ≤ 2 V	2 V
2 V < |*range*| ≤ 20 V	20 V
20 V < |*range*| ≤ 40 V	40 V
40 V < |*range*| ≤ 100 V	100 V
100 V < |*range*| ≤ 200 V (for HPSMU)	200 V

Table 2-24 **Current Measurement Range**

range	Range Value
0 A < |*range*| ≤ 10.5 pA (for HRSMU)	10 pA
10.5 pA < |*range*| ≤ 105 pA (for HRSMU)	100 pA
105 pA < |*range*| ≤ 1.15 nA (for HRSMU)	1 nA
0 A < |*range*| ≤ 1.15 nA	
1.15 nA < |*range*| ≤ 11.5 nA	10 nA
11.5 nA < |*range*| ≤ 115 nA	100 nA
115 nA < |*range*| ≤ 1.15 μA	1 μA
1.15 μA < |*range*| ≤ 11.5 μA	10 μA
11.5 μA < |*range*| ≤ 115 μA	100 μA
115 μA < |*range*| ≤ 1.15 mA	1 mA
1.15 mA < |*range*| ≤ 11.5 mA	10 mA
11.5 mA < |*range*| ≤ 100 mA (for MPSMU or HRSMU)	100 mA
11.5 mA < |*range*| ≤ 115 mA (for HPSMU)	100 mA
115 mA < |*range*| ≤ 1 A (for HPSMU)	1 A

NOTE

SMU Filter

During pulse spot measurement, the SMU filter is set to OFF. If SMU filter was ON before the measurement, it will be set to ON after the measurement.

Offline Mode

This function returns measurement values to *value* variable as follows:

Table 2-25 **For current measurements:**

Meas.Range	Returned Value	Polarity
≠ 0	Range Value	Same as SMU *output value*
0	I compliance	Same as SMU *output value*
omitted	last valid range setting value	

Table 2-26 **For voltage measurements:**

Meas.Range	Returned Value	Polarity
≠ 0	V compliance	Same as SMU *output value*
0	V compliance	Same as SMU *output value*
omitted	last valid range setting value	

Example

```
int err;
double current;

……

err=set_pbias(12, V_SOURCE, 20, -3, 15, 0.4, 1, 0.5, 1E-3);
err=measure_p(12, I_MEAS, &current, 1E-3);
```

See Also

- set_pbias
- set_piv

measure_spa

Revision	D.05.16 or later
Control	SPA

This function measures the oscillation frequency and the amplitude at the peak frequency. The gate propagation delay time can also be calculated.

Synopsis

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

int measure_spa (double *osc_freq, double *amp, double *delay, double n,
                 double fcr);
```

Argument	Range Restrictions/Description
osc_freq	Return variable for the peak frequency in the searched frequency range.
amp	Return variable for the measured peak amplitude.
delay	Return variable for the calculated gate delay (Tr + Tf) of the ring oscillator is returned. (The number of inverters must be specified to obtain this value.)
n	Number of inverters in the ring oscillator.
fcr	Resolution of the marker frequency counter. 0 ≤ *fcr* ≤ 1E+5

- *fcr*

  If you do not wish to use the marker frequency counter function, specify 0 as *fcr*.

  * Keysight N9000

  The frequency counter resolution is always 0.001Hz regardless of fcr. On the other hand, N9000 gate time is set to 1 / fcr. If fcr is lower than 2 the gate time is rounded to its upper limit, 500 ms.

* Keysight E4411B

The frequency counter resolution is rounded to E4411B internal values as shown below.

Marker frequency counter resolution specified as input parameter *fcr*	Actual marker frequency counter resolution set by Keysight E4411B
$0 < fcr < 9.5$	1 Hz
$9.5 \leq fcr \leq 99.5$	10 Hz
$99.5 \leq fcr < 999.5$	100 Hz
$999.5 \leq fcr \leq 9999.5$	1 kHz
$9999.5 \leq fcr < 99999.5$	10 kHz
$99999.5 \leq fcr \leq 1E+5$	100 kHz

If *fcr*, the resolution of the marker frequency counter, is specified to be greater than zero, the marker frequency counter function in the spectrum analyzer is used. The approximate frequency is obtained with a resolution of ((*maxf* – *minf*) / 400), then the marker frequency counter function is activated to measure the oscillation frequency more precisely.

NOTE

Throughput optimization tip

Do not use the frequency counter function unless the measurement requires very high resolution. The measurement time increases significantly when the frequency counter function is used.

- *delay*, *osc_freq*, *N*

  The gate propagation delay time can be calculated based on the following equation.

  delay = 1 / ($N \times$ *osc_freq*)

Where:
delay = tpLH + tpHL
tpLH: Gate propagation delay time going from Low to High
tpHL: Gate propagation delay time going from High to Low

N must be represented as shown in the following equation, otherwise the ring oscillator will not oscillate.

$N = 2 \times n + 1$ (where n is an integer)

If *N* is equal to or less than zero, then this function returns –1 with a *delay* of –999999.9999.

If an input parameter error occurs, the function returns –1 and the following sentinel values (indicating an error condition) are returned to the output parameters.

osc_freq = –999999.9999
amp = –999999.9999
delay = –999999.9999
stat = –1

See Also

- set_spa
- sweep_spa

measure_v, measure_vt

Revision	A.01.00 or later
Control	SMU, RFU

These functions measure DC voltage using the specified SMU or DC bias source (SMU) of RFU and return the measurement value.

Synopsis

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

Argument	Range Restrictions/Description	
port	Specify the SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008), RFU11(20301), RFU12(20302).
	pin number	1 to 49 for SMU 101 to 105 for RFU11 201 to 205 for RFU12
voltage	Specify a pointer to a double variable in which to return the measurement result.	
range	Specify the measurement range.	
time_stamp (Note 2)	Specify a pointer to a double variable in which to return the time stamp of measurement.	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 Only for measure_vt function. If DC bias source of RFU, measurement timestamp value is not returned.

- *port*

  To specify the port, use either the *port address* of port, or *pin number* of any SWM pin connected to the port.

- *voltage*

  The measurement value is returned to the *voltage* variable.

- *range*

  For SMU, *range* specifies the range value as follows:

range	Range Value	Measurement Resolution (Note 1)	
		High-speed ADC	High-resolution ADC
0 V < |*range*| ≤ 2 V	2 V	100 μV	2 μV
2 V < |*range*| ≤ 20 V	20 V	1 mV	20 μV
20 V < |*range*| ≤ 40 V	40 V	2 mV	40 μV
40 V < |*range*| ≤ 100 V	100 V	5 mV	100 μV
100 V < |*range*| ≤ 200 V (for HPSMU)	200 V	10 mV	100 μV

Note 1 You can select ADC by using the set_smu_ch function.

For RFU, *range* specifies the range value as follows:

range	Range Value
0 V < |*range*| ≤ 2 V	2 V
2 V < |*range*| ≤ 20 V	20 V

range	Range Value
20 V < |*range*| ≤ 40 V	40 V
40 V < |*range*| ≤ 100 V	100 V

If you specify 0.0 for the *range*, auto-ranging is used, which performs the measurement in range that gives the best resolution.

If you specify a non-zero value for the *range*, limited auto-ranging is used. The specified range is used if possible, but changes to a higher range if not possible.

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

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

- *time stamp*

  measure_vt function returns accumulated time at measurement start in seconds (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) to the *time_stamp* variable.

  If reset_timestamp function has been executed after init_system function or `/opt/hp4070/bin/hp4070 -start` command, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode This function returns measurement values to *voltage* variable as shown in the following table. Returned values are determined by the *range* value or present source mode of the SMU.

Source Mode	*range*	Returned Value	Polarity
IS	0	V compliance	same as SMU I output value
	≠0	range value	
	omitted	last valid range value	
VS	n.a.	SMU V output value	same as SMU V output value

Example

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

See Also

- force_meas
- measure_i

measure_vth

Revision	A.01.00 or later
Control	SMU

This function performs FET gate-source threshold voltage measurement by using linear search with step skipping.

Returned threshold voltage is the X intercept of line that is tangent to maximum gradient point of Vgs-Ids or Vgs-√{Ids} characteristics curve.

The set_vth function is used to set the conditions of this measurement.

Before performing threshold voltage measurement, you must connect SMU ports and SWM pins to be used, and set the voltage bias for drain, source, and substrate of an FET. Refer to the following example:

```
connect_pin(drain, drain_pin);
connect_pin(gate, gate_pin);
connect_pin(source, source_pin);
connect_pin(substrate, substrate_pin);
force_v(drain, vd, vd_range, d_comp);
force_v(source, vs, vs_range, s_comp);
force_v(substrate, vsb, vsbrange, sb_comp);
set_vth(gate, drain, vg_start, vg_stop, vg_step,id_start, meas_range, 3, 2,
0.0, gate_comp, 0, type);
measure_vth(1, &vth, &status, &gm, &id, num_samples, vd);
```

Synopsis

```
int measure_vth (int region, double *vth, int *status, double *gm, double
                 *id, int num_samples, double vd);
```

Argument	Range Restrictions/Description
region	Specify whether the measurement is performed in linear region or saturated region. You may choose from one of following macros: VTH_LIN (=0): Linear region of FET VTH_SAT (=1): Saturated region of FET
vth	Specify a pointer to double variable in which to return the found vth.

Argument	Range Restrictions/Description
status	Specify a pointer to integer variable in which to return the measurement status code. The status returned is one of the following: NORMAL_MEAS (=0) VTH_DOMAIN (=1) VTH_ABNORMAL (=2) VTH_FAILED (=4) VTH_NOGOAL (=256) VTH_NOTENOUGH_POINT (=512) VTH_NOMAX_GRADIENT (=1024) VTH_ZERO_GRADIENT (=2048)
gm	Specify a pointer to double variable in which to return the maximum gradient value (=Gm).
id	Specify a pointer to double variable in which to return the drain current for threshold voltage.
num_samples	Specify the number of samples used to calculate and find maximum gradient of measured curve. 2 to 100.
vd	Specify a drain voltage used to calculate Vth if *region* is VTH_LIN. If *region* is VTH_SAT, this is dummy parameter and is ignored.

- *region*

  region specifies whether threshold voltage measurement is performed in linear region or saturated region of FET.

  If linear region (VTH_LIN) is specified, Vgs-Ids characteristics curve is used to search for gate-source threshold voltage.

  If saturated region (VTH_SAT) is specified, Vgs-√{Ids} characteristics curve is used to search for gate-source threshold voltage.

- *vth*

  Threshold voltage is returned to *vth*.

  If *region* is 1 (saturated region), X intercept of tangent line (described previously) is returned to *vth*.

If *region* is 0 (linear region) and *vd* is specified, X intercept – *vd*/2 is returned to *vth*.

- *status*

  One of following measurement statuses is returned to *status*.

Status Code	Condition
NORMAL_MEAS (=0)	Measurement ended normally.
VTH_DOMAIN (=1)	*id_start* (specified by set_vth) is not found between *vg_start* and *vg_stop*
VTH_ABNORMAL (=2)	Abnormal data (for example, oscillating, measurement abort)
VTH_FAILED (=4)	Abnormal compliance loop.
VTH_NOGOAL (=256)	*id_start* (specified by set_vth) is not found between *vg_start* and *vg_stop*
VTH_NOTENOUGH_POINT (=512)	Search ended (reached *vg_stop* before measuring at *num_samples* measurement points).
VTH_NOMAX_GRADIENT (=1024)	No maximum gradient found. Gradient of Vgs-Ids or Vgs-Ö{Ids} curve continued to increase during search.
VTH_ZERO_GRADIENT (=2048)	Maximum gradient is 0.

- *gm* Maximum gradient value is returned to *gm*. This is maximum gradient of Vgs-Ids or Vgs-√{Ids} characteristics curve.
- *id* The drain current for threshold voltage is returned to *id* variable.
- *num_samples* This is number of samples used to calculate and find maximum gradient of Vgs-Ids or Vgs-√{Ids} curve. Least squares method is used for this calculation.
- *vd* This is not a force value, but is used for following calculation.

  If *region* is VTH_LIN (linear region), the following is returned to *vth*:

  X intercept of tangent line – *vd*/2.

  If *region* is VTH_SAT (saturated region), X intercept of tangent line is returned to *vth*.

See Also

- set_vth

measure_v3458

Revision	A.01.00 or later
Control	DVM

The function measures DC voltage using the DVM and returns the measurement value in volts (V).

Synopsis

```
int measure_v3458 (double *voltage);
```

Argument	Range Restrictions/Description
voltage	Specify a pointer to double variable in which to return the measurement result.

Measurement is performed at the range that provides highest resolution (auto ranging), and measurement result is returned to *voltage* variable.

Offline Mode This function returns 0.0 in offline mode.

measure_ztr, measure_ztrt

Revision	C.04.00 or later
Control	HSCMU

This function measures impedance and phase (θ) of a series equivalent circuit by using the HSCMU and returns error compensated values.

Synopsis

```
int measure_ztr (double *z, double *theta, double range);
int measure_ztrt (double *z, double *theta, double range, double
          *time_stamp);
```

Argument	Range Restrictions/Description
z	Pointer to double variable in which to return the measured impedance.
theta	Pointer to double variable in which to return the measured phase. To discard the measured value, specify NULL.
range	Specify the impedance measurement range. Specify 0.0 for Auto range mode or a Numeric expression [Ω]: real value
time_stamp (Note 1)	Specify a pointer to a double array in which to return the time stamps for measured values.

Note 1 Only for measure_ztrt function

- *z* and *theta*

  This function performs a impedance measurement and returns the error compensated impedance and phase values to *z* and *theta*, respectively.

  If measurement failed, 9.99999E+9 is returned to each variable.

- *range*

  An actual impedance range for impedance measurement is determined from the calculated impedance range value (*range*), test signal level (Vosc) and measurement frequency (fm) as shown in following table:

range value	Measurement Frequencies (fm), Test Signal level (Vosc), and Corresponding Impedance Ranges			
	Vosc = 30, 50, 100 mV_{rms}		Vosc = 10 mV_{rms}	
	fm ≤ 100 kHz	120 kHz ≤ fm	fm ≤ 100 kHz	120 kHz ≤ fm
0 ≤ *range* ≤ 300 Ω	100 Ω	100 Ω	100 Ω	100 Ω
300 Ω < *range* ≤ 1 kΩ	300 Ω	300 Ω	100 Ω	100 Ω
1 kΩ < *range* ≤ 3 kΩ	1 kΩ	1 kΩ	1 kΩ	1 kΩ
3 kΩ < *range* ≤ 10 kΩ	3 kΩ	3 kΩ	1 kΩ	1 kΩ
10 kΩ < *range* ≤ 30 kΩ	10 kΩ	10 kΩ	10 kΩ	10 kΩ
30 kΩ < *range* ≤ 100 kΩ	30 kΩ	30 kΩ	10 kΩ	10 kΩ
100 kΩ < *range* ≤ 300 kΩ	100 kΩ	30 kΩ	100 kΩ	10 kΩ
300 kΩ < *range*	300 kΩ	30 kΩ	100 kΩ	10 kΩ

  If *range* parameter is 0, the HSCMU uses the Auto ranging mode, so HSCMU automatically changes to range that gives best resolution.

- *time_stamp*

  If you execute measure_ztrt function, the accumulated time from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command to measurement start is returned in seconds to *time_stamp*.

  If reset_timestamp function has been executed after last init_system or /opt/hp4070/bin/hp4070 -start, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode In offline mode execution, this function returns data as follows:

range	Returned Capacitance Value	Returned Conductance Value
AUTO	previous C range value default = 10 pF	previous G range value default = 100 μS
other than AUTO	C range value	G range value

Example

```
int err;
double *z, *theta;

……

err = force_v(CMH, 5.0, 20.0, 1E-3);
err = measure_ztr(z, theta, 0.0);
```

See Also

- port_status

O

This section describes the C functions that begin with O.

open_corr84

Revision	A.01.00 or later
Control	CMU, (HSCMU)

NOTE

CMU Control TIS function

This function can be used to execute 4080 measurement program for using CMU on the 4080 with HSCMU.

This function performs nothing for HSCMU without an error.

Do not use this function to make new measurement program for using HSCMU.

This function performs the OPEN compensation measurement for the CMU.

Before executing this function, you must disconnect CMU connection cables from the testhead.

Synopsis

```
int open_corr84 (void);
```

This function triggers the OPEN compensation function in the CMU, and compensation data is stored in the internal non-volatile memory of the CMU.

Refer to "CMU Error Compensation" in *User's Guide* for more information about the error compensation theory for the CMU.

Example

```
open_corr84();
```

See Also

- short_corr_th

open_corr_th

Revision	A.01.10 or later
Control	HSCMU, CMU

This function performs the OPEN compensation measurement for the HSCMU on the two measurement pins which are connected to CMH and CML and enables error compensation beyond measurement pins, for example, up to the probe needles.

Before executing this function:

- Connect CMH and CML to the desired measurement pins.
- Nothing must be connected to the probe needles connected to the specified measurement pins.

Synopsis

```
int open_corr_th (void);
```

This function measures error factors beyond two capacitance measurement pins, for example, up to the probe needles.

After executing this function, the 4080 automatically compensates capacitance measurement values by using measured error factors until ending the test session.

You can execute this function for any two measurement pins (but not chuck connection pin).

By repeating this function to the different combination of measurement pins, you can measure OPEN error factors for up to 500 different combination of measurement pins.

You can store measured compensation data into the file by using store_corr_data function and re-load the data for error-compensation later or in another test session by using load_corr_data function.

NOTE

You must re-measure if the conditions change

When the conditions change, re-measure the error compensation data. If you don't re-measure, the 4080 performs error compensation incorrectly.

For example, if the probe card is changed, you must re-measure the error compensation data.

Example

```
open_corr_th();
```

See Also

- short_corr_th
- status_corr
- disable_corr

P

This section describes the C functions that begin with P.

p_ivt_measure

Revision	D.05.13 or later
Control	SMU

This function performs parallel multichannel sampling measurement according to the condition set by the ivt_set_t and p_ivt_set_iv functions.

Synopsis

```
int p_ivt_measure (int n, int port[n],int mode[n],double range[n],double
                   measure[n][x],double tstamp[x]);
```

Where *x* is the *number* of samples specified in ivt_set_t function.

Argument	Range Restrictions/Description	
n	Specify how many ports (1 to 8) will perform measurement. This value determines the size of the following arrays: *port*, *range*, and the primary index of *measure* and *tstamp*.	
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008)
	pin number	1 to 49
mode	Specify whether to perform a voltage or current measurement for sampling. Use one of the following macros: V_MEAS (=1): Voltage measurement I_MEAS (=2): Current measurement	

Argument	Range Restrictions/Description	
range	Specify the measurement ranging mode. 0.0 for Auto-ranging mode or	
	For voltage measurements [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current measurement [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)
measure	Pointer to double array where to return the measured values.	
tstamp	Pointer to double array where to return the time stamp values for measured values.	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port* array

  To specify the measurement ports for p_ivt_measure, you can specify port addresses of SMUs or pin numbers of any measurement pin connected to the SMU ports. This array should have *n* elements.

- *mode* array

  Assign the measurement mode values of measurement ports to *modes* array elements that correspond to each *port* element.

Measurement mode value	Description
V_MEAS (=1)	Voltage measurement
I_MEAS (=2)	Current measurement

- *range* array

  Assign the range values of measurement ports to *range* array elements that correspond to each *port* element. The assigned range value determines the ranging mode of measurement port when the sampling interval (*interval* set by ivt_set_t) ≥ 2 ms as follows:

  range value = 0.0 auto-ranging is used, which performs measurement in range that gives the best resolution.

  range value ≠ 0.0 limited auto-ranging is used. The specified range is used if possible, but changes to a higher range if not possible. See measure_i or measure_v for range values.

  If sampling interval (*interval* set by ivt_set_t) < 2 ms, the specified range value is ignored and the measurement range is fixed to the compliance setting set by p_ivt_set_iv.

NOTE

Type of ADC and integration time to be used

If the multiple measurement ports are specified, the high-speed ADC is used.

For the integration time, the setting by set_adc, set_adc_i, or set_adc_v function is basically used.

But, the integration time may be changed from the setting by set_adc, set_adc_i, or set_adc_v to keep the sampling interval.

- *measure* array, *tstamp* array

  The measured values are returned to *measure* array.

  The time stamp values are returned to *tstamp* array.

In Offline mode 0.0 is returned to *measure* array as the measured values.

The timestamp values calculated from the sampling interval and sampling mode specified by ivt_set_t are returned to *tstamp* array.

See Also ivt_set_t, p_set_group, p_ivt_set_iv, p_ivt_status, set_adc, set_adc_v, set_adc_i

p_ivt_set_iv

Revision	D.05.13 or later
Control	SMU

This function sets the output parameters of SMU bias sources for the synchronous parallel SMU sampling measurement.

Synopsis

```
int p_ivt_set_iv (int n, int port[n], int mode[n], double range[n], double
                  bias[n], double base[n], double compliance[n]);
```

Argument	Range Restrictions/Description	
n	Specify how many ports (1 to 8) will force bias voltage or current. This value determines the size of the following arrays: *port*, *range*, *bias*, *base* and *compliance*.	
port	Pointer to an integer array that determines SMU *bias* port addresses. You can specify the port addresses directly or specify pin numbers that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008)
	pin number	1 to 49
mode	Pointer to an integer array that specifies whether to force a voltage or current bias for sampling. Use one of the following macros: V_SOURCE (=1): Voltage force I_SOURCE (=2): Current force	
range	Pointer to a double array that specifies the voltage or current force range: 0.0 for Auto-ranging mode or	
	For voltage measurements [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current measurement [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)

Argument	Range Restrictions/Description	
bias, *base*	Pointer to double arrays that specify the bias and base values of voltage or current being forced from each port.	
	For voltage force [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current force [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)
compliance	Pointer to a double array that specifies the voltage or current compliance of each port. REMAIN macro (Note 2) or	
	For voltage force [V]:	–100 to 100 –200 to 200 (HPSMU)
	For current force [A]:	–0.1 to 0.1 –1 to 1 (HPSMU)

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1.
REMAIN means the previous compliance setting if source mode of SMU is same as the previous mode. Otherwise:
If source mode was VS mode and is now IS mode, compliance is set to 20 V.
If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

- *port*

  To specify the SMU, use either the *port address* of SMU, or the *pin number* of any measurement pin connected to the SMU port.

  For parallel sampling measurements, the multiple SMU bias sources and the multiple SMU sampling units can be synchronized with each other.

  To set the synchronization group of SMU bias sources and SMU sampling measurement units, use the p_set_group function.

If the p_set_group function is not used to specify the synchronization group, a common bias source unit can be used for multiple sampling measurement units. In this case, specify the same bias source unit for all the array element numbers of the measurement units that share the bias source unit.

- *bias*, *base*

  The *bias* determines the bias voltage/current forced from SMU.

  The *base* determines the base voltage/current that forced before bias voltage/current.

- *compliance*

  The *compliance* determines the compliance current/voltage setting of SMU.

See Also

- ivt_set_t
- p_set_group
- p_ivt_measure
- p_ivt_status
- set_adc, set_adc_v, or set_adc_i

p_ivt_status

Revision	D.05.13 or later
Control	SMU

This function returns the statuses of the data measured by p_ivt_measure.

Synopsis

```
int p_ivt_status (int n, int port[n], int status[n][x]);
```

Where *x* is the *number* of samples specified in ivt_set_t function.

Argument	Range Restrictions/Description	
n	Specify the number of measurement ports (1 to 8). This value determines the size of the following arrays: *port* and primary index of *status*.	
port	Pointer to an integer array that determines SMU *measurement* port addresses. You can specify the port addresses directly or specify pin numbers that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008)
	pin number	1 to 49
status	Pointer to an integer array in which to return measurement statuses.	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port* array

  To specify the measurement ports, you can specify port addresses of SMUs or pin numbers of any measurement pin connected to the SMU ports.

- *status* array

The measurement status for all sampling points of a port are returned to elements of this array that corresponds to the port specified in *port* array. Thus, the size of the first dimension of this array should be the same as the number of ports. For status, see the port_status_t function.

Offline Mode In offline mode, this function returns 0 as the status value.

See Also

- ivt_set_t, p_ivt_set_iv
- p_ivt_measure

p_measure_m

Revision	D.05.12 or later
Control	SMU

This function enables to perform the parallel multi-channel DC voltage or current measurements using up to eight SMU ports.

This function *cannot* set the measurement mode, so you should use force_i or force_v functions to set the desired measurement mode for each SMU port before measure_m function.

The force_i function sets the SMU port to voltage measurement mode, and force_v sets the SMU port to current measurement mode.

Synopsis

```
int p_measure_m (int para_mode, int n, int ports[n], double meas_vals[n],
                 double ranges[n], double time_stamps[n]);
```

- *para_mode*

  If 1 is specified, this function performs the multi-channel measurements in parallel.

  If 0 is specified, this function performs the multi-channel measurements sequentially. In this case, the operation is equivalent to measure_m.

Argument	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 ports. This array must have *n* elements. The array should contain the SMU port addresses or SWM pin numbers that are connected to ports.
meas_vals	Specify a pointer to double array in which to return the measured values.This array must have *n* elements.

Argument	Range Restrictions/Description
ranges	Specify a pointer to double array that contains the measurement range for each port. This array must have *n* elements.
time_stamps	Specify a pointer to a double array in which to return the time stamps for measured values.

- *n*

  This function can perform up to 8 simultaneous measurements. *n* argument specifies how many ports will perform measurement. You must declare array arguments for this function as in the following example:

```
int n=3, ports[3];
double meas_vals[3], ranges[3], time_stamps[3];
```

- *port*

  Pointer to an integer array that determines the measurement ports. This array must have *n* elements.

  The array elements can contain the SMU port addresses or SWM pin numbers that are connected to ports as follows:

  port address: You can specify the following macros (or values):

  SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).

  pin number: 1 to 49

- *meas_vals*

  Measurement results are returned to *meas_vals* array in order that corresponds to elements of the *port* array.

- *ranges*

  You should specify values in *ranges* array in order that corresponds to elements of the *ports* array.

  Specify range values according to the measurement modes of the corresponding SMUs. Refer to the measure_i or measure_v for measurement range values.

- *time_stamps*

This function returns accumulated time at measurement start for each SMU in seconds (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) to the *time_stamps* array.

If reset_timestamp function has been executed after last init_system or /opt/hp4070/bin/hp4070 -start, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode See measure_m

Example

```
int err;

……

int para_mode=1, 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=p_measure_m(para_mode, n, ports, meas_vals, ranges, time_stamps);
```

See Also

- measure_m

p_measure_vth

Revision	D.05.12 or later
Control	SMU

This function performs parallel FET gate-source threshold voltage (Vth) measurements by using linear search with step skipping.

Returned threshold voltages are the X intercepts of lines that are tangent to the maximum gradient points of Vgs-Ids or Vgs-√{Ids} characteristics curves.

The p_set_vth function is used to set the conditions of this measurement.

Vth measurement requires two independent SMU channels, and thus, up to four parallel Vth measurements can be performed.

Before performing threshold voltage measurements, you must connect the SMU ports and SWM pins to be used, and set the voltage biases for drains, sources, and substrates of the FETs.

Synopsis

```
int p_measure_vth (int region, double vths[n], int statuses[n], double
                   gms[n], double ids[n], int num_samples, double vd[n]);
```

Argument	Range Restrictions/Description
region	Specify whether the measurement is performed in linear region or saturated region. You may choose from one of following macros: VTH_LIN (=0): Linear region of FET VTH_SAT (=1): Saturated region of FET
vths	Specify the pointer to a double array in which to return the found vth values.

Argument	Range Restrictions/Description
statuses	Specify the pointer to an integer array in which to return the measurement status codes. The status returned is one of the following: NORMAL_MEAS (=0) VTH_DOMAIN (=1) VTH_ABNORMAL (=2) VTH_FAILED (=4) VTH_NOGOAL (=256) VTH_NOTENOUGH_POINT (=512) VTH_NOMAX_GRADIENT (=1024) VTH_ZERO_GRADIENT (=2048)
gms	Specify the pointer to a double array in which to return the maximum gradient values (=Gm).
ids	Specify the pointer to a double array in which to return the drain current values for threshold voltage.
num_samples	Specify the number of samples used to calculate and find maximum gradient of measured curve. 2 to 100.
vd	Specify the pointer to a double array which stores drain voltages used to calculate Vth if *region* is VTH_LIN. If *region* is VTH_SAT, this is dummy parameter and is ignored.

- *region*

  region specifies whether threshold voltage measurement is performed in linear region or saturated region of FET.

  If linear region (VTH_LIN) is specified, Vgs-Ids characteristics curve is used to search for gate-source threshold voltage.

  If saturated region (VTH_SAT) is specified, Vgs-√{Ids} characteristics curve is used to search for gate-source threshold voltage.

- *vths*

  Return array for threshold voltages.

If *region* is 1 (saturated region), X intercepts of tangent lines (described previously) are returned to *vths*.

If *region* is 0 (linear region) and *vds* is specified, X intercept – *vds*/2 are returned to *vths*.

- *statuses*

  One of following measurement statuses is returned to each *statuses* array element.

Status Code	Condition
NORMAL_MEAS (=0)	Measurement ended normally.
VTH_DOMAIN (=1)	*id_start* (specified by p_set_vth) could not be found between *vg_start* and *vg_stop*
VTH_ABNORMAL (=2)	Abnormal data (for example, oscillating, measurement abort)
VTH_FAILED (=4)	Abnormal compliance loop.
VTH_NOGOAL (=256)	*id_start* (specified by p_set_vth) could not be found between *vg_start* and *vg_stop*
VTH_NOTENOUGH_POINT (=512)	Search ended (reached *vg_stop* before measuring *num_samples* measurement points).
VTH_NOMAX_GRADIENT (=1024)	No maximum gradient found. Gradient of Vgs-Ids or Vgs-√{Ids} curve continued to increase during search.
VTH_ZERO_GRADIENT (=2048)	Maximum gradient is 0.

- *gms*

  Return array for maximum gradient values, which are the maximum gradients of Vgs-Ids or Vgs-√{Ids} characteristics curves.

- *ids*

  Return array for drain current values for threshold voltage.

- *num_samples*

  Number of samples used to calculate and find maximum gradient of Vgs-Ids or Vgs-√{Ids} curve. Least squares method is used for the calculation.

- *vds*

  These are not force values, but are used for the following calculation.

  If *region* is VTH_LIN (linear region), the following is returned to *vths*:

  X intercept of tangent line – *vds*/2.

  If *region* is VTH_SAT (saturated region), X intercept of tangent line is returned to *vths*.

See Also

- p_set_vth

PORT

Revision	A.01.00 or later
Control	n.a.

This function returns the address of the specified SWM port. Port addresses are used to designate ports in TIS functions.

You can use this function or pre-defined macros to specify port addresses.

Synopsis

```
int PORT (int type, int numb);
```

Argument	Range Restrictions/Description
type	Integer value that selects the type of port. See Table 2-27.
numb	Integer value that specifies the SWM port position. See Table 2-28.

The *type* parameter specifies one of the ports shown in Table 2-27. Macro can be used.

Table 2-28 shows the port numbers, the port addresses, and the port names for each type.

This function does not report a TIS runtime error code. If unknown *type* or *numb* is specified, 32000 is returned as port address.

Example

```
int collector, base;

collector = PORT(0,2);
base = PORT(0,1);
```

Table 2-27 Port Types

type	Macro	Description
0	INTERNAL	SMU, GNDU, GSMU, and GCMU ports
1	EXTERNAL	AUX, DVM, CMU, GAUX,and Extended Path ports
2	HFPORT	HF ports
3	THFPORT	THF ports
4	PULSESW	Pulse Switch ports
5	PGUADDR	PG of PGU or SPGU
6	RFUPORT	RFU
8	RFPORT	RF ports
10	PNAPORT	PNA

Table 2-28 Port Numbers and Port Addresses

type	Port Name	Macro (Note 1)	Description	*numb*	Port Address
0	SMU1	SMU1	SMU1 (for low current measurement)	1	20001
	SMU2	SMU2	SMU2 (for low current measurement)	2	20002
	SMU3	SMU3	SMU3 (for general DC measurement)	3	20003
	SMU4	SMU4	SMU4 (for general DC measurement)	4	20004
	SMU5	SMU5	SMU5 (for general DC measurement)	5	20005
	SMU6	SMU6	SMU6 (for general DC measurement)	6	20006
	SMU7	SMU7	SMU7 (for general DC measurement)	7	20007
	SMU8	SMU8	SMU8 (for general DC measurement)	8	20008
	GNDU	GNDU	DC Ground Unit	9	20009
	GSMU1	INGRD1	Guard of SMU1	-1	20109
	GSMU2	INGRD2	Guard of SMU2	-2	20110

type	Port Name	Macro (Note 1)	Description	*numb*	Port Address
0	GCMU	CCOM	CMS Guard Terminal	11	20113
1	AUX1	none	External AUX1	1	20101
	AUX2	none	External AUX2	2	20102
	AUX3	none	External AUX3	3	20103
	AUX4	none	External AUX4	4	20104
	DVMH (AUX5)	DVMH	DVM High Port, External AUX5	5	20105
	DVML (AUX6)	DVML	DVM Low Port, External AUX6	6	20106
	CMH (AUX7)	CMH	CMS High Port, External AUX7	7	20107
	CML (AUX8)	CML	CMS Low Port, External AUX8	8	20108
	GAUX1	EXGRD1	Guard of AUX1	-1	20111
	GAUX2	EXGRD2	Guard of AUX2	-2	20112
	Extended Path	EXIO	Extended Path	10	20114
2	HF1	HF1	HF1	1	20201
	HF2	HF1	HF2	2	20202
	HF3	HF3	HF3	3	20203
	HF4	HF4	HF4	4	20204
	HF5	HF5	HF5	5	20205
	HF6	HF6	HF6	6	20206
3	THF1	THF1	T-Connected HF1	1	20211
	THF2	THF2	T-Connected HF2	2	20212
	THF3	THF3	T-Connected HF3	3	20213
4	PSI11	PSI11	Pulse Switch IN11	11	20251
	PSI12	PSI12	Pulse Switch IN12	12	20252
	PSI21	PSI21	Pulse Switch IN21	21	20253

type	Port Name	Macro (Note 1)	Description	*numb*	Port Address
4	PSI31	PSI31	Pulse Switch IN31	31	20254
	PSI41	PSI41	Pulse Switch IN41	41	20255
	PSI51	PSI51	Pulse Switch IN51	51	20256
	PSI61	PSI61	Pulse Switch IN61	61	20257
	PSI62	PSI62	Pulse Switch IN62	62	20258
	PSI71	PSI71	Pulse Switch IN71	71	20259
	PSI72	PSI72	Pulse Switch IN72	72	20260
	PSO1	PSO1	Pulse Switch OUT1	1	20261
	PSO2	PSO2	Pulse Switch OUT2	2	20262
	PSO3	PSO3	Pulse Switch OUT3	3	20263
	PSO4	PSO4	Pulse Switch OUT4	4	20264
	PSO5	PSO5	Pulse Switch OUT5	5	20265
	PSO6	PSO6	Pulse Switch OUT6	6	20266
	PSO7	PSO7	Pulse Switch OUT7	7	20267
	PSC1	PSC1	Pulse Switch Control 1	101	20268
	PSC2	PSC2	Pulse Switch Control 2	102	20269
5 (Note 2)	PG11	none	PG Channel 11	11	20281
	PG12	none	PG Channel 12	12	20282
	PG21	none	PG Channel 21	21	20283
	PG22	none	PG Channel 22	22	20284
	PG31	none	PG Channel 31	31	20285
	PG32	none	PG Channel 32	32	20286
	PG41	none	PG Channel 41	41	20287

type	Port Name	Macro (Note 1)	Description	*numb*	Port Address
5 (Note 2)	PG42	none	PG Channel 42	42	20288
	PG51	none	PG Channel 51	51	20289
	PG52	none	PG Channel 52	52	20290
6	RFU11	RFU11	RFU1 Port1	11	20301
	RFU12	RFU12	RFU1 Port2	12	20302
8	RF11	RF11	RF Block1 Port1	11	20401
	RF21	RF22	RF Block1 Port2	12	20402
	RF31	RF31	RF Block1 Port3	13	20403
	RF41	RF41	RF Block1 Port4	14	20404
	RF12	RF12	RF Block2 Port1	21	20406
	RF22	RF22	RF Block2 Port2	22	20407
	RF32	RF32	RF Block2 Port3	23	20408
	RF42	RF42	RF Block2 Port4	24	20409
10	PNA11	PNA11	PNA1 Port1	11	20509
	PNA12	PNA12	PNA1 Port2	12	20510

Note 1 Port address can be specified by using the macro instead of using the PORT function.

Note 2 Port Address gives the unit address of PGU or SPGU. The PORT function returns this value.

For PGU (81150A), PG11 and PG12 are available.

For SPGU, the available units depend on the system configuration.

port_status, port_status_t

Revision	A.01.00 or later
Control	SMU, GNDU, HSCMU, CMU, PGU, SPGU, DVM, RFU

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

Argument	Range Restrictions/Description	
port	Specify a port address or a pin number that is connected to the port.	
	port address:	20001 to 20009, 20101 to 20114, 20201 to 20206, 20211 to 20213, 20301 to 20302, 20401 to 20409, 20509 to 20510. For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead. Refer to description after this table.
	pin number:	1 to 49 for other than RFU 101 to 105 for RFU11 201 to 205 for RFU12
status	Specify a pointer to integer variable in which to return the status code. For details about the status code, refer to the description after this table.	
time_stamp	Only for port_status_t. Specify a pointer to double variable in which to return the time stamp of measurement.	

- *port*

  To specify a port for which to return the status, specify *pin number* or *port address*. Or, you can use the following macros to specify a port:

Macro	Port Address	Description
SMU1	20001	SMU1
SMU2	20002	SMU2
SMU3	20003	SMU3
SMU4	20004	SMU4
SMU5	20005	SMU5
SMU6	20006	SMU6
SMU7	20007	SMU7
SMU8	20008	SMU8
GNDU	20009	Ground Unit
AUX1	20101	AUX1 port
AUX2	20102	AUX2 port
AUX3	20103	AUX3 port
AUX4	20104	AUX4 port
AUX5	20105	AUX5 port
AUX6	20106	AUX6 port
AUX7	20107	AUX7 port
AUX8	20108	AUX8 port
DVMH	20105	DVM high port: AUX5
DVML	20106	DVM low port: AUX6
CMH	20107	CMS high port: AUX7
CML	20108	CMS low port: AUX8

Macro	Port Address	Description
INGRD1	20109	Guard of SMU1
INGRD2	20110	Guard of SMU2
EXGRD1	20111	Guard of AUX1
EXGRD2	20112	Guard of AUX2
CCOM	20113	CMS guard terminal
EXIO	20114	Extended Path
HF1	20201	HF1 port
HF2	20202	HF2 port
HF3	20203	HF3 port
HF4	20204	HF4 port
HF5	20205	HF5 port
HF6	20206	HF6 port
THF1	20211	THF1 port
THF2	20212	THF2 port
THF3	20213	THF3 port
RFU11	20301	RFU1 port1
RFU12	20302	RFU1 port2
RF11	20401	RF11 port
RF21	20402	RF21 port
RF31	20403	RF31 port
RF41	20404	RF41 port
RF12	20406	RF12 port
RF22	20406	RF22 port
RF32	20406	RF32 port

Macro	Port Address	Description
RF42	20406	RF42 port
PNA11	20509	PNA1 port1
PNA12	20510	PNA1 port2

- *status*

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

Unit Name	Returned Value	Condition
SMU / RFU (dc measurement)	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 overflow.
	DCS_SWP_STOPPED (=5)	Sweep was aborted by compliance condition. (stop mode = 2 in set_iv or set_piv)
HSCMU	NORMAL_MEAS(=0)	Normal
	CMU_UNBALANCE(=1)	Analog bridge is unbalanced.
	CMU_DC_OVERFLOW(=2)	DC bias source is overloaded.
	CMU_ERR_DSP(=3)	A/D converter is not working.
	CMU_SWP_STOPPED(=4)	Sweep measurement has been aborted.
	CMU_OUT_OF_RNG(=5)	Measurement overflow.

Unit Name	Returned Value	Condition
CMU	NORMAL_MEAS(=0)	Normal
	CMU84_UNBALANCE(=1)	Analog bridge is unbalanced.
	CMU84_ADC_DOWN(=2)	A/D converter is not working.
	CMU84_OVERLOAD(=3)	Signal source is overloaded.
	CMU84_ALC_DOWN(=4)	Automatic Level Control (ALC) is unable to keep constant level.
DVM	NORMAL_MEAS(=0)	Normal
GNDU	NORMAL_MEAS(=0)	Normal
SPGU	SPGU_NORMAL(=0)	Normal
	SPGU_EMERGENCY(=1)	Emergency state occurs.
	SPGU_OVER_CURRENT(=2)	Overcurrent detected.
	SPGU_OVER_VOLTAGE(=4)	Overvoltage detected.
	SPGU_HIGH_TEMPERATURE(=8)	Overtemperature condition detected.
RFU (S parameter measurement)	NORMAL_MEAS(=0)	Normal
	VNA_OVERLOAD(=1)	The input signal to the VNA port exceeded the maximum input level.
	VNA_PLL_UNLOCK(=2)	PLL circuit of VNA became unlocked during measurement.
	VNA_NO_COEFF(=3)	No calibration data corresponding to the test condition.
Others	NORMAL_MEAS(=0)	Normal

The returned status of a port that has not performed measurement is always NORMAL_MEAS (Normal).

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

- SMU

  `DCS_OSC > DCS_OVERFLOW > DCS_COMP > DCS_COMP_OTHER`

- HSCMU

  `CMU_OUT_OF_RNG > CMU_SWP_STOPPED > CMU_ERR_DSP > CMU_DC_OVERFLOW > CMU_UNBALANCE`

- CMU

  `CMU84_ALC_DOWN > CMU84_OVERLOAD > CMU84_ADC_DOWN > CMU84_UNBALANCE`

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.

- *time_stamp*

  When you use port_status_t function, accumulated time at last measurement start in seconds (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) is returned to *time_stamp*.

  If reset_timestamp function has been executed after init_system function or `/opt/hp4070/bin/hp4070 -start` command, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

  If the port is not a measurement port, such as GNDU, 0.0 is always returned to *time_stamp*.

Offline Mode This function returns NORMAL_MEAS to *status* and 0.0 to *time_stamp*.

Example

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

prep_pg

Revision	B.02.00 or later
Control	PGU, SPGU

This function sets the pulse period of the pulse generators (PG).

If SPGU is used in the ALWG mode, executing this function will cause an error.

Synopsis

```
int prep_pg (double period);
```

Argument	Range Restriction/Description
period	Specify the pulse period. Numeric expression [s]. If SPGU, 0 or T to 10.0 If PGU (81150A), 0 or T to 999.0 Resolution is 1E-8. 0 automatically sets the minimum period T (see "force_pg" on page 81).

pround_cmu

Revision	C.04.00 or later
Control	HSCMU

This function determines if you can specify an inaccurate values within the allowable range for the test frequency and test signal level parameters of the following functions for the capacitance and impedance measurements using HSCMU:

- set_cmu, set_cmu84
- set_freq, frequency
- set_cvf, set_zf

Synopsis

```
int pround_cmu (int on_off);
```

Argument	Description/Default
on_off	integer expression: 1: Enable rounding (ON) other than 1: Disable rounding (OFF)

If you specify 1 to *on_off* parameter of this function, the set_cmu, set_cmu84, set_freq, frequency, set_cvf, or set_zf rounds the specified value of test frequency or test signal level to the nearest accurate setting value of the HSCMU.

The initial setting of the system is “Disable rounding”.

p_rsearch_iv

Revision	D.05.12 or later
Control	SMU

This function can perform parallel *real-time* search measurements according to the conditions set by the p_set_lsearch function.

4080 does not support parallel binary search measurements.

Real-time means that values are returned after each search step.

The p_rsearch_iv function returns measurement data and status after each search step. To use p_rsearch_iv in a measurement program, put p_rsearch_iv in a `for` loop. To abort the search measurement, use the p_rsearch_stop function.

p_search_iv performs similar searches, but *not* in real-time. Only the *final* values are returned after search completion.

If the p_set_sync function is called between p_set_lsearch and p_rsearch_iv, p_rsearch_iv will perform search measurements using the synchronous search unit(s) specified by p_set_sync.

force_i and force_v functions cannot be used between the p_rsearch_iv and p_set_lsearch functions.

Keysight recommends to execute p_rsearch_stop after completing parallel *real-time* search measurements.

Synopsis

```
int p_rsearch_iv (double searches[n], int statuses[n], double senses[n],
                  double time_stamps[n]);
```

Argument	Range Restrictions/Description
searches	Specify the pointer to a double array in which to return the present force values of search ports.
statuses	Specify the pointer to an integer array in which to return the measurement status codes. One of following integers is returned to *statuses* array element. You can use the associated macros in your program to determine which status was returned: NORMAL_MEAS (=0) SEARCH_DOMAIN (=1) SEARCH_ABNORMAL (=2) SEARCH_FAILED (=4) SEARCH_STOPPED (=8) SEARCH_COMP_OTHER (=16) SEARCH_COMP (=32) SEARCH_OVERFLOW (=64) SEARCH_OSC (=128) 4096
senses	Specify the pointer to a double array in which to return the measurement values of sense ports.
time_stamps	Specify the pointer to a double array to return the time stamps of present measurement values.

- array size

  Each array should be equal in size to each other, as well as to the arrays used in p_set_lsearch. If an array is smaller than is necessary, then the retrieved data will be incomplete (i.e. if 8 SMUs are used and the array size is 6, then only the data for the first 6 SMUs will be returned). On the other hand, if an array is larger than the required size, the extra array elements will be filled with data –9999999.99999.

- *searches* and *senses*

  The p_rsearch_iv function obtains SMU data during a search measurement and returns the following after each search step:

- Present source values of search units. (Returned to *searches* array.)
- Measurement values of sense units. (Returned to *senses* array.)

- *statuses*

  The p_rsearch_iv function returns integer values to *statuses* array. You can use the following macros in your program to determine which status was returned:

Status Macro	Condition
NORMAL_MEAS (=0)	Measured value is normal.
SEARCH_DOMAIN (=1)	Search value not found (between *start* and *stop*).
SEARCH_ABNORMAL (=2)	Search aborted.
SEARCH_FAILED (=4)	Sense value is not within specified accuracy, or an abnormal compliance loop occurred.
SEARCH_STOPPED (=8)	Search measurement was completed.
SEARCH_COMP_OTHER (=16)	Unit other than sense unit reached compliance.
SEARCH_COMP (=32)	Sense unit reached compliance.
SEARCH_OVERFLOW (=64)	A/D converter overflow.
SEARCH_OSC (=128)	SMU is oscillating
4096	Search measurement is not yet started (for sequential measurement mode only).

- *time_stamps*

  Return array for elapsed time in seconds at measurement start (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command).

  If reset_timestamp function has been executed after init_system function or `/opt/hp4070/bin/hp4070 -start` command, the elapsed time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode This function returns values to the *searches*, *statuses*, and *senses* arrays as follows. (Each parameter name is an array element.)

- For linear search measurements (p_set_lsearch function):

  The *searches* is set to the average of *starts* and *stops* specified in the p_set_lsearch function. The *statuses* is always set to 0, which in the offline mode means that the search value was found. The *senses* is set to the *targets* specified in the p_set_lsearch function.

See also

- p_set_lsearch
- p_set_sync (for synchronous search measurements)
- p_rsearch_stop
- p_search_iv

p_rsearch_stop

Revision	D.05.12 or later
Control	SMU

This function stops the parallel *real-time* search measurements.

Synopsis

```
int p_rsearch_stop (void);
```

This function is used to abort real-time search measurements that were started by the p_rsearch_iv function.

This function should also be executed after completing parallel real-time search measurements, even if the measurements complete successfully.

Example

```
p_rsearch_stop();
```

See Also

- p_rsearch_iv

p_rsweep_miv, p_rsweep_mivt

Revision	D.05.12 or later
Control	SMU

The p_rsweep_miv and p_rsweep_mivt functions are the same, except p_rsweep_mivt also returns time stamp values.

These functions can perform the following real-time, parallel multichannel sweep type measurements:

- staircase sweep
- synchronous staircase sweep

This function cannot set the measurement mode. To set desired measurement mode for each *measurement only* port, use the force_i or force_v functions before the p_rsweep_miv function.

The rsweep_miv function performs real-time, parallel multichannel staircase sweep measurements according to the measurement parameters set by the p_set_iv function.

If p_set_sync is also specified, rsweep_miv can perform real-time, parallel multichannel synchronous staircase sweep measurements according to measurement parameters set in the p_set_iv and p_set_sync functions.

The p_rsweep_miv function returns measurement data after each sweep measurement step, so the measurement results can be plotted on the screen as each measurement is performed. To use p_rsweep_miv in a measurement program, call p_rsweep_miv in a `for` loop. To abort the sweep measurement, call the p_rsweep_stop function.

Synopsis

```
int p_rsweep_miv (int n, int m, int ports[n], double ranges[n], double
            meas_vals[n], int statuses[n], double sources[m], double syncs[m]);

int p_rsweep_mivt (int n, int m, int ports[n], double ranges[n], double
            meas_vals[n], int statuses[n], double sources[m], double syncs[m],
            double time_stamps[n]);
```

Argument	Range Restrictions/Description
n	Specify how many ports (1 to 8) will perform measurement. The size of the *ports*, *ranges*, *meas_vals*, *statuses*, and *time_stamps* arrays must be greater than or equal to *n*.
m	Specify the number of sweep source ports. This value determines the size of the *sources* and *syncs* arrays.
ports	Specify the pointer to an integer array that determines the measurement ports. This array must have *n* elements. The array should contain the SMU port addresses or SWM pin numbers that are connected to ports.
ranges	Specify the pointer to a double array that contains the measurement range for each port. This array must have *n* elements. For Auto range mode, specify 0.
meas_vals	Specify the pointer to a double array in which to return the measurement values for each port. This array must have *n* elements.
statuses	Specify the pointer to an integer array in which to return the measurement status code for each measurement port. One of following integers is returned. You can use the associated macros in your program to determine which status was returned. NORMAL_MEAS (= 0) DCS_COMP_OTHER (= 1) DCS_COMP (= 2) DCS_OSC (= 3) DCS_OVERFLOW (= 4) DCS_SWP_STOPPED(= 5)
sources	Specify the pointer to a double array in which to return the present sweep source values (which are determined by p_set_iv). If you don't need this data, specify NULL (*syncs* must also be NULL).

Argument	Range Restrictions/Description
syncs	Specify the pointer to a double array in which to return the present secondary sweep source values (which are determined by p_set_sync function). If you don't need this data, specify NULL. If you don't use the p_set_sync function, this argument must be NULL.
time_stamps (Note 1)	Specify the pointer to a double array in which to return the time stamp values.

Note 1 Only for p_rsweep_mivt function.

- *n*

  This function starts real-time sweep and can perform up to 8 simultaneous measurements at each sweep step. The *n* argument specifies how many ports will perform measurement. You must declare array arguments for this function as in the following example:

```
int n=3, ports[3], statuses[3];
double meas_vals[3], ranges[3], time_stamps[3];
double sources[2], syncs[2];
```

  The valid range of *n* is 1 through 8.

- *ports*

  This array determines which ports will perform measurements.

  The array elements should contain the SMU port addresses or SWM pin numbers that are connected to ports as follows:

  port address: You can specify the following macros (or values):

  `SMU1` (20001), `SMU2` (20002), `SMU3` (20003), `SMU4` (20004), `SMU5` (20005), `SMU6` (20006), `SMU7` (20007), `SMU8` (20008).

  pin number: 1 to 49

  When assigning arrays, the *ports* array elements correspond to the elements of the other arrays. For example, element 1 of the *ports* array corresponds to element 1 of the *ranges* array, *meas_vals* array, *statuses* array, and *time_stamps* array.

The size of *meas_vals* array, *statuses* array, and *time_stamps* array must be greater than or equal to number of ports that will perform measurements.

- *meas_vals*, *statuses*

  The p_rsweep_miv function returns measurement value of each port to *meas_vals* array.

  This function returns integers to the *statuses* array to indicate the measurement status. You can use the associated macros in your program to determine which status was returned.

Status Value	Condition
NORMAL_MEAS (0)	Normal measurement end
DCS_COMP_OTHER (1)	Another unit reached compliance
DCS_COMP (2)	This unit reached compliance
DCS_OSC (3)	This unit is oscillating
DCS_OVERFLOW (4)	DC source measurement overflow
DCS_SWP_STOPPED (5)	Sweep was aborted by compliance condition.

  Depending on the *mode* parameter setting of the p_set_stop_mode function, p_rsweep_miv measurements may be aborted if a measurement SMU reaches compliance. If this occurs before a measurement loop ends, p_rsweep_miv returns 9999999.99999 to *meas_vals*.

- *sources*, *syncs*

  The p_rsweep_miv function returns present output values of sweep sources to *sources*.

  If you do not need to get source values, specify NULL for *sources* and *syncs* arguments.

  For synchronous sweep measurement, p_rsweep_miv also returns present output value of synchronous sweep sources to *syncs*.

  If synchronous sweep ports are not specified by p_set_sync, NULL must be specified to this argument.

- *time_stamps*

  If you use p_rsweep_mivt, elapsed time at measurement start for each SMU in seconds (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) is returned to *time_stamps* array.

  If reset_timestamp function has been executed after init_system function or `/opt/hp4070/bin/hp4070 -start` command, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode In offline mode, this function returns values as follows:

- Measurement values are calculated and returned to *meas_vals* array elements as follows:

$$\text{nth measurement value} = \frac{basevalue}{\text{number of steps} - 1} \times (n - 1)$$

Where: *number of steps* is set by the p_set_iv function, and *basevalue* is determined as follows:

Table 2-29 For current measurements

Meas. Range	*basevalue*	Polarity
0	I compliance	Same as SMU *output value*
≠ 0	Range value	

Table 2-30 For voltage measurements

Meas. Range	*basevalue*	Polarity
= 0	2.0 V	Always positive
≠ 0	V compliance	Same as SMU *output value*

- *statuses* array element is always set to 0.

- *time stamps* array element is set to elapsed time in seconds since January 1, 1970, or the first force_i, force_v, or force_meas function after reset_timestamp function.
- *sources* and *syncs*

  The primary sweep source values determined by p_set_iv are returned to *source* array.

  The synchronous sweep source values determined by p_set_sync are returned to *syncs*.

Example

```
err=p_rsweep_miv(n, m, ports, ranges, measures, statuses, sweeps, NULL);
```

See Also

- p_rsweep_stop, p_set_iv, p_set_sync, p_sweep_miv, p_set_stop_mode

p_rsweep_stop

Revision	D.05.12 or later
Control	SMU

This function stops parallel *real-time* sweep measurements.

Synopsis

```
int p_rsweep_stop (void);
```

This function is used to abort the parallel real-time sweep measurements that were started by p_rsweep_miv function.

This function should also be executed after completing parallel real-time sweep measurements, even if the measurements complete successfully.

After this function executes, the sweep source SMU is set to the sweep start value, but sweep settings are not changed.

Example

```
for (i=0; i<number; i++) {
err=p_rsweep_miv(n, m, ports, ranges, measures, statuses, sweeps, NULL);
if(err==-1){
p_rsweep_stop();
break;
}
p_rsweep_stop();
}
```

See Also

- p_rsweep_miv

p_search_iv

Revision	D.05.12 or later
Control	SMU

This function performs parallel search measurements according to the conditions set by the p_set_lsearch or p_set_bdv_srch function. After searches are completed, the final output values of search units, as well as the measurement values of sense units are returned.

4080 does not support parallel binary search measurements.

If the p_set_sync function is called between p_set_lsearch and p_search_iv, p_search_iv will perform search measurements using the synchronous search unit(s) specified by p_set_sync.

Breakdown voltage search measurement using p_set_bdv_srch does not support synchronous port set by p_set_sync. So, if p_set_sync is called between p_set_bdv_srch and p_search_iv, p_search_iv will reset the settings set by p_set_sync.

The force_i or force_v function cannot be used between the p_search_iv and p_set_lsearch functions.

For real-time search, refer to the p_rsearch_iv function.

NOTE

Differences between p_search_iv and p_search_iv2

p_search_iv2 returns the actual measured value to the *senses* array element even if a following measurement status is detected during a search measurement:

- Another unit than sense unit has reached compliance
- Sense unit has reached compliance
- Measurement overflow
- Sense unit is oscillating

p_search_iv returns –9999999.99999 to the *senses* array element if one of the above statuses is detected.

Synopsis

```
int p_search_iv (double searches[n], int statuses[n], double senses[n]);
```

Argument	Range Restrictions/Description
searches	Specify the pointer to a double array in which to return the search values.
statuses	Specify the pointer to an integer array in which to return the search measurement statuses. One of the following integers is returned to *statuses* array element. You can use the associated macros in your program to determine which status was returned. NORMAL_MEAS (=0) SEARCH_DOMAIN (= 1) SEARCH_ABNORMAL (= 2) SEARCH_FAILED (= 4)
senses	Specify the pointer to a double array in which to return the target values measured by the sense units.

- array size

  Each array should be equal is size to each other, as well as to the arrays used in p_set_lsearch or p_set_bdv_srch. If an array is smaller than is necessary, then the retrieved data will be incomplete (i.e. if 8 SMUs are used and the array size is 6, then only the data for the first 6 SMUs will be returned). On the other hand, if an array is larger than the required size, the extra array elements are filled with data –9999999.99999.

- *searches*

  This function performs search measurements, and returns the search values to the *searches* array. The search value is the value output by the search unit when the target value is found by the sense unit.

  If a search value is not found, one of the following is returned to the *searches* array element:

–9999999.99999	Search value was not found.
9999999.99999	Search measurement was aborted.

- *senses*

  The target values measured by the sense units are returned to the *senses* array.

  In the case of p_set_bdv_srch measurements, *breakdown_currents* values specified by p_set_bdv_srch are returned to the *senses* array.

- *statuses*

  The p_search_iv function returns an integer (0, 1, 2, or 4) to each *statuses* array element. You can use the following macros in your program to determine which status was returned:

  - For linear or breakdown voltage search measurements (p_set_lsearch or p_set_bdv_srch function):

    NORMAL_MEAS (=0): Search value successfully found.

    SEARCH_DOMAIN (=1): Search value not found (between the *start* and *stop* values specified in p_set_lsearch or p_set_bdv_srch).

    SEARCH_ABNORMAL (=2): Abnormal data

    SEARCH_FAILED (=4): Abnormal compliance loop

Offline Mode This function returns values to the *searches*, *statuses*, and *senses* arrays as follows:

- For linear search measurements (p_set_lsearch function):

  The *searches* array element is set to the average of *starts* and *stops* array elements specified in the p_set_lsearch function.
  The *statuses* array element is always set to 0, which in the offline mode means that the search value was found.
  The *senses* array element is set to the *targets* array element specified in the p_set_lsearch function.

Example

```
if (p_search_iv(searches, statuses, senses) == -1) error_rep();
```

See Also

- p_set_lsearch
- p_set_sync (for synchronous search measurements)
- p_search_iv2
- p_rsearch_iv

p_search_iv2

Revision	D.05.12 or later
Control	SMU

This function performs parallel search measurements according to the conditions set by the p_set_lsearch or p_set_bdv_srch function. After searches are completed, the final output values of search units, as well as the measurement values of sense units are returned.

4080 does not support parallel binary search measurements.

If the p_set_sync function is called between p_set_lsearch and p_search_iv2, the p_search_iv2 function will perform a search measurement using the synchronous search unit(s) specified by p_set_sync.

Breakdown voltage search measurement using p_set_bdv_srch does not support synchronous port set by p_set_sync. So, if p_set_sync is called between p_set_bdv_srch and p_search_iv2, p_search_iv2 will reset the settings set by p_set_sync.

The force_i or force_v function cannot be used between the p_search_iv2 and p_set_lsearch functions.

For real-time search, refer to the p_rsearch_iv function.

NOTE

Differences between p_search_iv and p_search_iv2

p_search_iv2 returns the actual measured value to the *senses* array element even if a following measurement status is detected during a search measurement:

- Another unit than sense unit has reached compliance
- Sense unit has reached compliance
- Measurement overflow
- Sense unit is oscillating

p_search_iv returns –9999999.99999 to the *senses* array element if one of the above statuses is detected.

Synopsis

```
int p_search_iv2 (double searches[n], int statuses[n], double senses[n],
                  double time_stamps[n]);
```

Argument	Range Restrictions/Description
searches	Specify the pointer to a double array in which to return the search values.
statuses	Specify the pointer to an integer array in which to return the search measurement statuses. One of the following integers is returned to *statuses* array element. You can use the associated macros in your program to determine which status was returned. NORMAL_MEAS (=0) SEARCH_DOMAIN (= 1) SEARCH_ABNORMAL (= 2) SEARCH_FAILED (= 4) SEARCH_COMP_OTHER(=16) SEARCH_COMP (=32) SEARCH_OVERFLOW (=64) SEARCH_OSC (=128)
senses	Specify the pointer to a double array in which to return the target values measured by the sense units.
time_stamps	Specify the pointer to a double array in which to return the time stamps for measured values.

- array size

  Each array should be equal in size to each other, as well as to the arrays used in p_set_lsearch or p_set_bdv_srch. If an array is smaller than is necessary, then the retrieved data will be incomplete (i.e. if 8 SMUs are used and the array size is 6, then only the data for the first 6 SMUs will be returned). On the other hand, if an array is larger than the required size, the extra array elements are filled with data –9999999.99999.

- *searches*

  This function performs a search measurements, and returns the search values to the *searches* array. The search value is the value output by the search unit when the target value is found by the sense unit.

  If search value is not found, one of the following is returned to the *searches* array element:

–9999999.99999	Search value was not found.
9999999.99999	Search measurement was aborted.

- *senses*

  The target values measured by the sense units are returned to the *senses* array.

  In the case of p_set_bdv_srch measurements, *breakdown_currents* values specified by p_set_bdv_srch are returned to the *senses* array.

- *statuses*

  The search_iv2 function returns an integer (0, 1, 2, 4, 16, 32, 64, or 128) to each *statuses* array element. You can use the following macros in your program to determine which status was returned:

  - For linear or breakdown voltage search measurements (p_set_lsearch or p_set_bdv_srch function):

NORMAL_MEAS (=0):	Search value successfully found.
SEARCH_DOMAIN (=1):	Search value not found (between the *start* and *stop* values specified in p_set_lsearch or p_set_bdv_srch).
SEARCH_ABNORMAL (=2):	Abnormal data
SEARCH_FAILED (=4):	Abnormal compliance loop
SEARCH_COMP_OTHER (=16):	Another unit than sense unit has reached compliance.

SEARCH_COMP (=32):	Sense unit has reached compliance.
SEARCH_OVERFLOW (=64):	Measurement Overflow.
SEARCH_OSC (=128):	Sense unit is oscillating.

- *time_stamps*

  Return array for elapsed time at measurement start in seconds (since last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) to the *time_stamps* array.

  If reset_timestamp function has been executed after init_system function or `/opt/hp4070/bin/hp4070 -start` command, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode This function returns values to the *searches*, *statuses*, and *senses* arrays as follows:

- For linear or breakdown voltage search measurements (p_set_lsearch or p_set_bdv_srch function):

  The *searches* array element is set to the average of *starts* and *stops* array elements specified in the p_set_lsearch function. The *statuses* array element is always set to 0, which in the offline mode means that the search value was found. The *senses* array element is set to the *targets* array element specified in the p_set_lsearch function.

p_set_bdv_srch

Revision	D.05.12 or later
Control	SMU

p_set_bdv_srch function sets the parameters for parallel breakdown voltage measurements using linear search technique.

To start this search measurement, use the p_search_iv or p_search_iv2 function.

For breakdown voltage search measurement:

1. Set the current compliance of the specified SMU to the *breakdown_currents* value.
2. The specified SMU sweeps from the *starts* voltage value to the *stops* voltage value while determining whether it has reached the compliance (i.e. the *breakdown_currents* value has been reached).
3. The output voltage setting value at current compliance is returned as a breakdown voltage.

Breakdown voltage measurement requires one SMU channel, and therefore up to eight parallel breakdown voltage measurements can be performed.

NOTE

Difference between p_set_bdv_srch and p_set_lsearch with compliance stop

p_set_bdv_srch is specifically optimized to shorten the wait time before compliance check at each voltage step so as to reduce the stresses to the DUT.

Synopsis

```
int p_set_bdv_srch (int para_mode, int n, int ports[n], double starts[n],
            double stops[n], double steps[n], double breakdown_currents[n],
            double comp_chk_wait_adjust, double comp_chk_delay, double
            delay, int skips[n], double fine_starts[n], int skip_backs[n], double
            skip_back_delays[n]);
```

Argument	Range Restrictions/Description	
para_mode	Specify parallel or sequential measurement mode. 1: parallel measurement mode (Note 1) 0: sequential measurement mode	
n	Specify the number of ports (1 to 8).	
ports (Note 2)	Pointer to an integer array that determines the SMU search port addresses. You can specify the port addresses directly or specify pin numbers that are connected to the ports:	
	port addresses	You can specify the following macros (or values (Note 3)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin numbers	1 to 49
starts, *stops*	Pointers to double arrays that specify the search start and stop voltages of SMUs. –100 to 100 (MPSMU and HRSMU) or –200 to 200 (HPSMU) [V]	
steps	Pointer to a double array that specifies the step sizes for fine search. 0 < *step size* ≤ 100 (MPSMU and HRSMU) or 0 < *step size* ≤ 200 (HPSMU) [V]	
breakdown_currents	Pointer to a double array that specifies the compliances of SMUs and is used to determine if the breakdown occurs. –0.1 to 0.1 (MPSMU and HRSMU) or –1 to 1 (HPSMU) [A]	
comp_chk_wait_adjust, *comp_chk_delay*	Optimizes the compliance check wait time. *comp_chk_wait*: 0.0 to 10.0 with 0.1 resolution. *comp_chk_delay* [s]: 0.0000 to 65.5350 with 0.0001 resolution.	
delay	Increases measurement delay time. 0.0000 to 65.5350 [s] with 0.0001 resolution.	

Argument	Range Restrictions/Description
skips	Pointer to an integer array that determines large step sizes. 1 to 20000
fine_starts	Pointer to a double array that specifies current values at which to start fine search. –0.1 to 0.1 (MPSMU and HRSMU) or –1 to 1 (HPSMU) [A]
skip_backs	Pointer to an integer array that specifies the numbers of large steps to go back after finding *fine_starts*. 0 to 100
skip_back_delays	Pointer to a double array that specifies the delay time values between skip back and start of fine search. 0.0000 to 65.5350 [s] with 0.0001 resolution.

Note 1 Use parallel measurement mode unless sequential measurement is required, i.e. for debugging purposes.

Note 2 The size of each array must be equal to the size of *ports* array. Array elements that have the same element number are for a single linear search measurement setup.

Note 3 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

In the following explanation of parameters, array names are used to refer to corresponding array element unless otherwise noted.

- *para_mode*

  In the sequential measurement mode, execution is almost the same as executing measurement by set_bdv_search multiple times. For the parallel mode, high-speed ADC is used for the parallel measurements even when high-resolution ADC is selected.

- *starts*, *stops*, and *steps*

  The *starts* can be either less than or greater than, but not equal to the *stops*. The *steps* must be positive.

If *starts* is less than *stops*, *steps* voltage is added to *starts* voltage at each iteration. If *starts* is greater than *stops*, *steps* voltage is subtracted at each iteration.

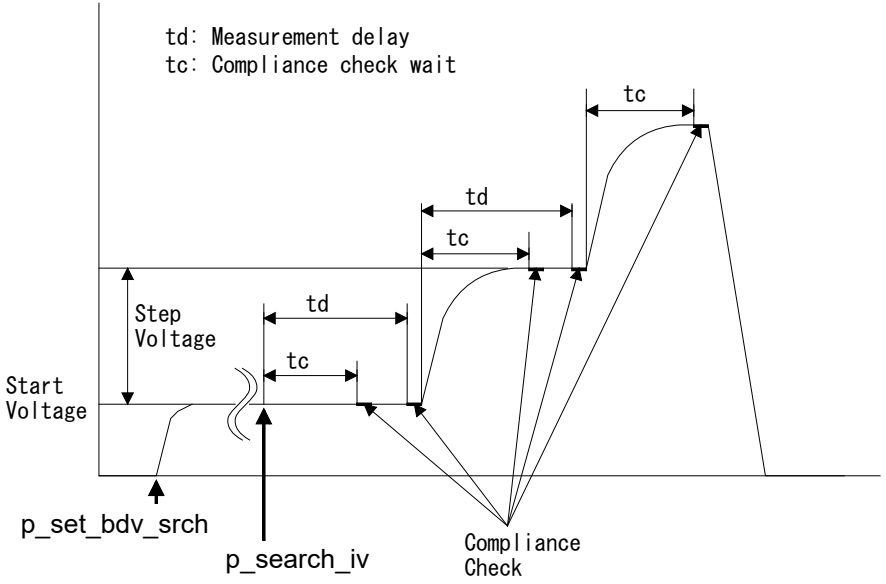
If *steps* is greater than the differences between *starts* and *stops*, the search will be performed at *starts* voltage and *stops* voltage only.

The output range of the specified SMU is set to the lowest range that includes the *starts* voltage, *stops* voltage, and *steps* voltage. Refer to force_v or force_i for output value, range, and compliance relationships.

Also, after the breakdown voltage is searched or SMU output is reached to the *stops* voltage, the SMU is set to the *starts* voltage.

Refer to Figure 2-3 on page 369.

Figure 2-1 Breakdown Voltage Search Output



- *comp_chck_wait_adjust* and *comp_chck_delay*

  For a reliable search measurement, an adequate wait time is necessary for the SMU to step up to the next voltage and stabilize its output before compliance check.

Large current for charging stray capacitance on the measurement path will flow from the SMU immediately after forcing a new step voltage, so an insufficient wait time for compliance check may cause the SMU to inappropriately detect this transient current as the *breakdown_currents*.

On the other hand, longer compliance check wait time means that DUT is stressed for longer durations and may become damaged.

You can optimize the compliance check wait time by using *comp_chck_wait_adjust* and *comp_chck_delay*.

The compliance check wait time is calculated as follows:

$$\text{comp check wait time} = Ws \times Wc + Dc$$

Where:

Ws: System's predefined wait time

Wc: *comp_chck_wait_adjust*

Dc: *comp_chck_delay*

- *delay*

  This parameter is valid only if the value of *skips* is greater than or equal to 2 (meaning step skipping method is enabled).

  If step skipping method is enabled, the current measurement will be performed at each voltage step.

  The measurement delay time is set to the system's predefined value + *delay*, or *delay* + measurement wait time adjusted by set_wait_time function.

  For reliable measurement, you can specify additional delay by using this parameter.

- *skips*, *fine_search_starts*, *skip_backs*, and *skip_back_delays*

  If *skips* is set to greater than or equal to 2, step skipping method is enabled and the current measurement will be performed at each voltage step.

  Breakdown voltage search with step skipping method is performed as follows:

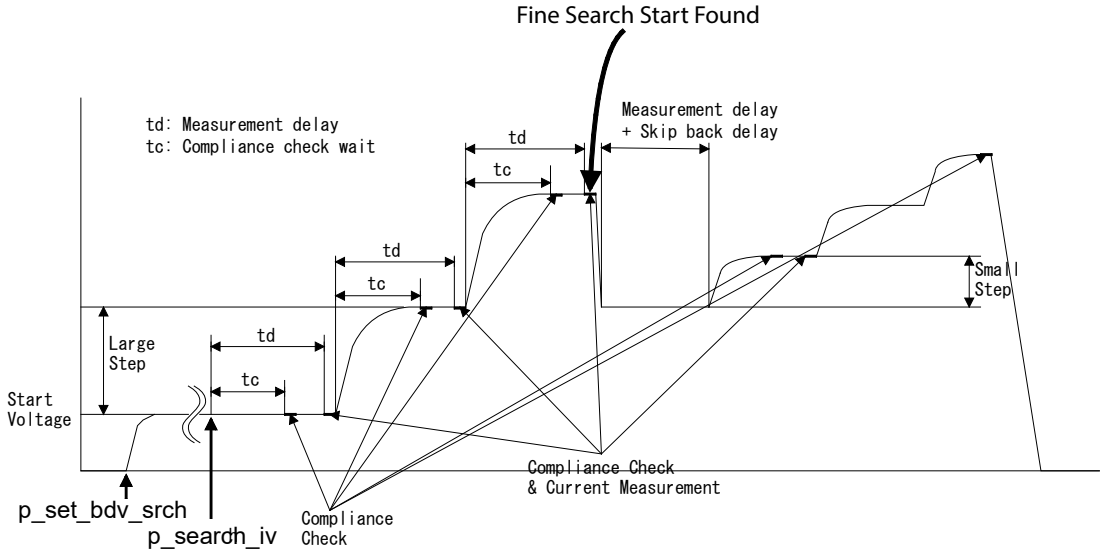
  1. SMU starts to sweep with large steps from *starts* voltage until measured value is greater than *fine_search_starts*.

Size of large step is calculated from *skips* as follows:

skips × *steps*

2. SMU goes back the specified number of large steps (*skip_backs*).
3. After waiting for *skip_back_delays*, SMU starts to sweep with small steps (*step*) until SMU reaches the compliance (*breakdown_currents*).

Figure 2-2 Breakdown Voltage Search with Step Skipping



See Also · p_search_iv, p_search_iv2, p_set_lsearch

p_set_group

Revision	D.05.13 or later
Control	SMU

This function sets the synchronous ports group for parallel sampling measurement by p_ivt_set_iv and p_ivt_measure functions.

Synopsis

```
int p_set_group (int n_of_p, int ports[n_of_p], int group);
```

Argument	Range Restrictions/Description	
n_of_p	Specify the number of ports (1 to 8) in the ports group. This value determines the size of the *ports* array.	
ports	Pointer to an integer array that specifies the SMU port addresses in the ports group. You can specify the port addresses directly or specify pin numbers that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008)
	pin number	1 to 49
group	Specify the group id of the ports group.	1 to 8 or 0 (clear setting)

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *ports*

  To specify the ports of a synchronous group for parallel sampling measurement, you can specify port addresses of SMUs or pin numbers of any measurement pin connected to the SMU ports.

- *group*

  If 0 is specified, the port group settings are cleared and 1 is assigned to all SMUs as a port group id.

  Also the init_system and connect() functions clear the port group settings.

  If the same SMU is assigned to the different port group, the error occurs when p_ivt_measure is executed.

See Also

- p_ivt_set_iv
- p_ivt_measure

p_set_iv

Revision	D.05.12 or later
Control	SMU

This function sets the sweep parameters for parallel I-V measurements that are started by p_rsweep_miv or p_sweep_miv function. This function can also set up one point (spot) parallel measurements by setting the *number* parameter to 1.

Synopsis

```
int set_iv (int para_mode, int n, int ports[n], int sweep_modes[n], double
            ranges[n], double starts[n], double stops[n], int number, double
            hold, double delay, double compliances[n], double
            power_compliances[n]);
```

Argument	Range Restrictions/Description	
para_mode	Specify parallel or sequential measurement mode. 1: parallel measurement mode (Note 1) 0: sequential measurement mode	
n	Specify the number of ports (1 to 8).	
ports	Pointer to an integer array that determines SMU *sweep source* port addresses. You can specify the port addresses directly or specify pin numbers that are connected to the ports:	
	port addresses	You can specify the following macros (or values (Note 2)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin numbers	1 to 49

Argument	Range Restrictions/Description
sweep_modes	Pointer to an integer array that specifies the sweep mode for each port (linear or logarithmic, voltage or current, and single or double). Use one of the following macros (or values): LINEAR_V (= 1) LINEAR_I (= 2) LINEAR_V_DBL (= 3) LINEAR_I_DBL (= 4) LOG_V (= –1) LOG_I (= –2) LOG_V_DBL (= –3) LOG_I_DBL (= –4)
ranges	Pointer to a double array that specifies voltage or current output ranges: 0 for auto-ranging mode or For voltage source: numeric expression [V]: –100 to 100 or –200 to 200(HPSMU) For current source: numeric expression [A]: –0.1 to 0.1 or –1 to 1(HPSMU)
starts, *stops* (Note 3)	Pointers to double arrays that specify sweep start and stop values for each port: For voltage source: numeric expression [V]: –100 to 100 or –200 to 200(HPSMU) For current source: numeric expression [A]: –0.1 to 0.1 or –1 to 1(HPSMU) For a log sweep, *start* and *stop* must be the same polarity, and cannot be 0.
number	Number of sweep steps: 1 to 1001 For single sweep, there will be *number* steps from *start* to *stop*. For double sweep, there will be *number* steps from *start* to *stop*, then *number* steps from *stop* to *start*.
hold	Specify time to wait after trigger outputs the *start* value. If a non-zero *delay* is set, an additional *delay* time follows after the *hold* time. Numeric expression [s]: 0 to 655.350 Resolution: 0.001
delay	Specify time to wait after each new output before making measurement: Numeric expression [s]: 0 to 65.5350 Resolution: 0.0001

Argument	Range Restrictions/Description
compliances	Pointer to a double array that specifies a voltage or current compliance value depending on the *sweep_modes* value for each port: REMAIN macro (Note 4) or For voltage compliance: numeric expression [V]: –100 to 100 or –200 to 200(HPSMU) For current compliance: numeric expression [A]: –0.1 to 0.1 or –1 to 1(HPSMU)
power_compliances	Pointer to a double array that specifies the limit of the power (voltage × current being forced and measured) applied to each port. Numeric expression [W]: 0.001 to 2 or 0.001 to 14(HPSMU) Resolution: 0.001 0 means no power compliance. See description.

Note 1 Use parallel measurement mode unless sequential measurement is required, i.e. for debugging purposes.

Note 2 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 3 If *number* is 1, specify the same value for both *starts* and *stops*.

Note 4 REMAIN can be specified only if optimization level is 0 or 1.
REMAIN means the previous compliance setting if source mode of SMU is same as the previous mode. Otherwise:
If source mode was VS mode and is now IS mode, compliance is set to 20 V.
If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

In the following explanation of parameters, array names are used to refer to the corresponding array element unless otherwise noted.

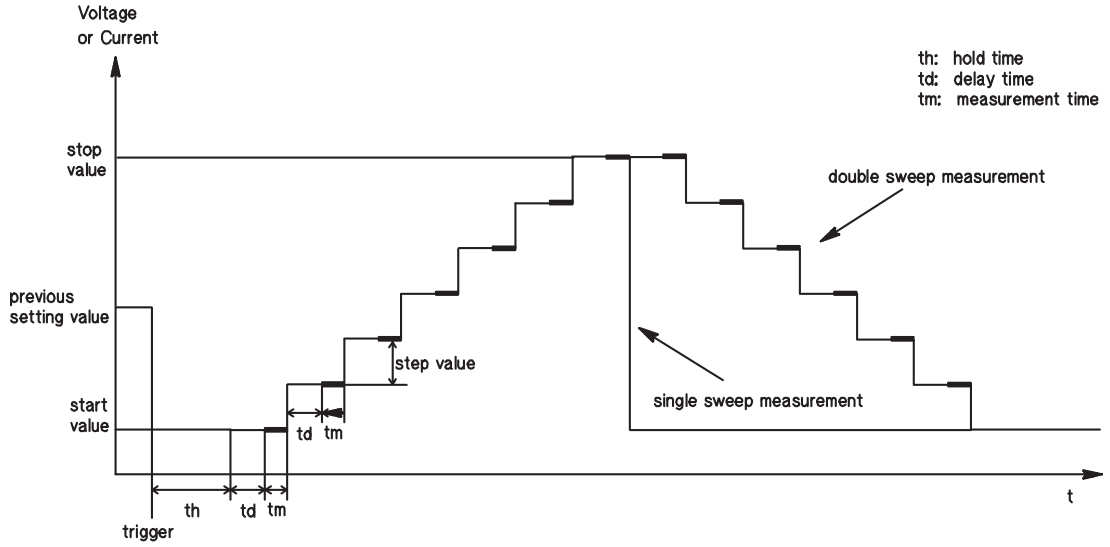
- *parallel mode*

  In the sequential measurement mode, the execution is almost the same as executing measurement by set_iv multiple times. For the parallel mode, high-speed ADC is used for the parallel measurements even when high-resolution ADC is selected.

- *starts* (voltage or current), *stops*, *number*, *hold*, and *delay*

  For log *sweep mode*, the *start* and *stop* values must have the same polarity and cannot be 0.

  When p_rsweep_miv or p_sweep_miv executes, *starts* value (voltage or current), *stops* value, *hold* time, and *delay* time determine the sweep measurement conditions as shown in the following figure.



The start, stop, and step values are determined as shown in the following table:

Table 2-31 In the linear sweep mode

Item	Description
start value	resolution: depends on the set voltage or current range.
stop value (Note 1)	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*

Note 1 The actual stop value may differ slightly from the specified value, as the step value may be rounded off depending on the setting resolution. Refer to force_v or force_i for setting resolution.

Table 2-32 In the log sweep mode

Item	Description
start value	resolution: depends on the set voltage or current range.
stop value (Note)	$Vstop = Vstart \times 10^{\frac{Rstep \times (Nstep-1)}{20}}$
step ratio	$Rstep = \frac{20 \times \log \frac{Vstop}{Vstart}}{Nstep-1}$ resolution: 0.02 dB/decade maximum: 20 dB/decade

Note 1 The actual stop value may differ slightly from the specified value as the step value may be rounded off depending on the setting resolution. Refer to force_v or force_i for setting resolution.

- *ports*

  To specify the SMU, use either the port address of SMU, or the pin number of any measurement pin connected to the SMU port.

  For parallel sweep measurements, a common sweep source unit can be used for multiple measurement units. This is called a "group". In this case, specify the same sweep source unit for all the array element numbers of the measurement units that share the sweep source unit. For more information, see p_set_lsearch.

- *sweep_modes*

  The *sweep_modes* determines whether the sweep is linear or logarithmic, double or single, and current or voltage:

LINEAR_V (=1):	linear single voltage sweep
LINEAR_I (=2):	linear single current sweep
LINEAR_V_DBL (=3):	linear double voltage sweep
LINEAR_I_DBL (=4):	linear double current sweep
LOG_V (=–1):	log single voltage sweep
LOG_I (=–2):	log single current sweep
LOG_V_DBL (=–3):	log double voltage sweep
LOG_I_DBL (=–4):	log double current sweep

  For log *sweep modes*, the *starts* and *stops* values must have the same polarity and cannot be 0.

NOTE

Sweep Modes for Parallel Search Measurements

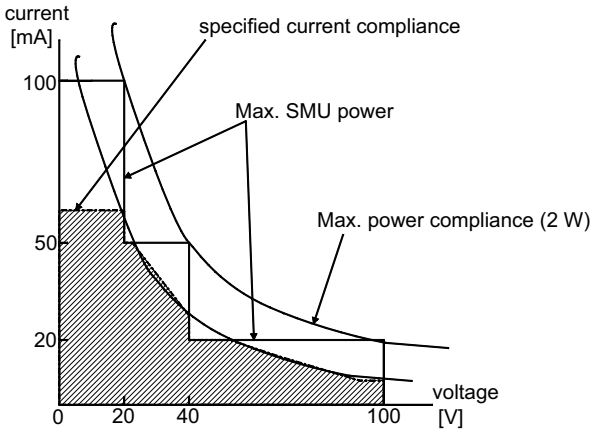
The sweep mode can be specified for each port. Single sweep and double sweep cannot be mixed, however, so specify all ports as single sweep, or all ports as double sweep. Voltage sweep and current sweep can be mixed.

- *ranges*
  - If *ranges* is 0:
    If the *sweep_modes* is 1,2,3 or 4 (linear), the voltage or current range during the sweep is set to lowest range that includes both *start* and *stop* voltage/current. The output range does not change during the sweep.

    If the *sweep_modes* is –1, –2 –3 or –4 (log), the range changes during the sweep to provide the optimum output range.
  - If *ranges* is not 0:
    - Voltage sweep

      Output range is fixed to the specified *ranges*.
    - Current sweep

      If the specified *ranges* includes both *start* and *stop* current, the specified *ranges* is used. If not, the SMU shifts the range to a range that can force both *start* and *stop* current. The output range does not change during the sweep.
- *power_compliances* and *compliances*
  - If 0.0 is specified for *power_compliances*
    The maximum voltage *compliances* or current *compliances* you can specify depends on the maximum output (*start* or *stop* value) during the sweep. Refer to force_v or force_i for details.
  - If non-zero value is specified for *power compliances*
    The maximum voltage *compliances* or current *compliances* can be set up to the maximum value of SMU, independent of maximum output during the sweep. The SMU can be swept at its maximum limits as shown in the following figure.



NOTE

p_set_iv does not support the stop mode, which is supported by set_iv. For parallel sweep/search measurements, use p_set_stop_mode instead.

Example

```
p_set_iv(1, 3, ports, modes, ranges, starts, stops, 51, 0.1, 0.05, comps,
p_comps);
```

See Also

- p_sweep_miv, p_rsweep_miv, p_set_sync (for sweep measurements), p_set_stop_mode

p_set_lsearch

Revision	D.05.12 or later
Control	SMU

This function sets the parameters for parallel linear search measurements, which can then be started by the p_search_iv, p_search_iv2, or p_rsearch_iv function.

For a linear search measurement, the search unit sweeps from the *starts* value to the *stops* value while the *sense_ports* determines if the *targets* value has been reached. (Parameters such as *starts* and *targets* are names of arrays that contain array elements that correspond to a single measurement.)

Synopsis

```
int p_set_lsearch (int para_mode, int n, int search_ports[n], int
                   sense_ports[n], int modes[n], double starts[n], double stops[n],
                   double steps[n], double targets[n], double ranges[n], double
                   search_comps[n], double delay);
```

Argument	Range Restrictions/Description	
para_mode	Specify parallel or sequential measurement mode. 1: parallel measurement mode (Note 1) 0: sequential measurement mode	
n	Specify the number of ports (1 to 8).	
search_ports (Note 2)	Pointer to an integer array that specifies SMU search port addresses. You can specify port addresses directly or specify pin numbers that are connected to the ports.	
	port addresses	You can specify the following macros (or values (Note 3)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin numbers	1 to 49

Argument	Range Restrictions/Description	
sense_ports	Pointer to an integer array that specifies SMU sense port addresses. You can specify port addresses directly or specify pin numbers that are connected to the ports.	
	port addresses	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin numbers	1 to 49
modes	Pointer to an integer array that determines search conditions. Choose from one of the following macros for each array element: VS_NORM_ABOVE (= 0) IS_NORM_ABOVE (= 1) VS_NORM_BELOW (= 4) IS_NORM_BELOW (= 5) VS_NORM_STAT (= 8) IS_NORM_STAT (= 9)	
starts, *stops*	Pointers to double arrays that specify the search start and stop values. Search unit is SMU in VS mode [V]: –100 to 100 or –200 to 200 (HPSMU) Search unit is SMU in IS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU)	
steps	Pointer to a double array that specifies the step sizes. Search unit is SMU in VS mode [V]: 0 < *step size* ≤ 100 or 0 < *step size* ≤ 200 (HPSMU) Search unit is SMU in IS mode [A]: 0 < *step size* ≤ 0.1 or 0 < *step size* ≤ 1 (HPSMU)	
targets	Pointer to a double array that specifies the target values of the sense ports. Sense unit is SMU in VS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) Sense unit is SMU in IS mode [V]: –100 to 100 or –200 to 200 (HPSMU)	
ranges	Pointer to a double array that specifies measurement ranges of the sense ports. 0 for Auto range mode or Sense SMU in VS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) Sense SMU in IS mode [V]: –100 to 100 or –200 to 200 (HPSMU)	

Argument	Range Restrictions/Description
search_comps	Pointer to a double array that specifies the voltage or current compliance values of search ports. REMAIN macro (Note 4) or Search SMU is in VS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) Search SMU is in IS mode [V]: –100 to 100 or –200 to 200 (HPSMU)
delay	Specify a settling wait time between search port output and sense port measurement. Numeric expression [s]: 0 to 65.5350 Resolution: 0.0001

Note 1 Use parallel measurement mode unless sequential measurement is required, i.e. for debugging purposes.

Note 2 The size of each array must be equal to the size of *search_ports* array. Array elements that have the same element number are for a single linear search measurement setup.

Note 3 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 4 REMAIN can be specified only if optimization level is 0 or 1. REMAIN means the previous compliance setting if source mode of search SMU is same as the previous mode. Otherwise: If source mode was VS mode and is now IS mode, compliance is set to 20 V. If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

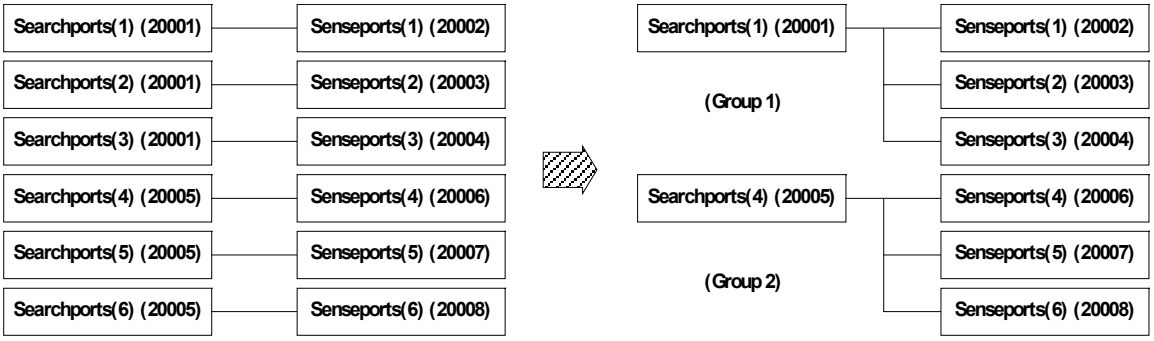
In the following explanation of parameters, array names are used to refer to the corresponding array element unless otherwise noted.

- *para_mode*

  In the sequential measurement mode, the execution is almost the same as executing measurement by set_lsearch multiple times. For the parallel mode, high-speed ADC is used for the parallel measurements even when high-resolution ADC is selected.

- Search units, sense units, array element numbers, and groups

  For parallel linear search measurements, a common search unit can be used for multiple sense units. This is called a "group". In this case, specify the same search unit for all the array element numbers of the sense units that share the search unit as in the following example. (There are two search units, six sense units, and two groups.)



- *modes* and *targets*

  The *modes* parameter determines the following conditions:

modes	Search Unit VS/IS mode	Stop Mode	Measurement/ Compliance Mode
VS_NORM_ABOVE	VS	sense value > *targets* value	Measurement mode
IS_NORM_ABOVE	IS	sense value > *targets* value	Measurement mode
VS_NORM_BELOW	VS	sense value < *targets* value	Measurement mode
IS_NORM_BELOW	IS	sense value < *targets* value	Measurement mode
VS_NORM_STAT (Note 1)	VS	sense value > *targets* value	Compliance mode
IS_NORM_STAT (Note 1)	IS	sense value > *targets* value	Compliance mode

Note 1 For *mode* VS_NORM_STAT or IS_NORM_STAT, you must set the sense unit to VS mode, and *target* must be a current value.
To set the sense unit to the desired mode, use the force_i or force_v function before p_set_lsearch.

- Stop Mode

  The *modes* setting determines the stop mode. The stop mode specifies when the search will stop, in terms of the comparison between sense value and *targets* value as listed in the above table.

- Measurement Mode or Compliance Mode

  If separate source and sense SMUs are used for search, you must set the sense unit to the desired source mode by using force_v or force_i.

  For measurement mode, the sense unit measures for every step and compares the result with *targets* value.

  For compliance mode, you must set the sense unit to VS mode, and the specified *targets* value must be a current value. The sense unit compliance is set to the specified *targets* value. No measurement is performed, and the search stops when the sense unit reaches the compliance value.

- *starts*, *stops*, and *steps*

  The *starts* value can be less than or greater than, but not equal to the *stops* value. The *steps* value must be positive.

  If *starts* value is less than *stops* value, *steps* value is added to *starts* value at each iteration. If *starts* value is greater than *stops* value, *steps* value is subtracted at each iteration.

  The output range of the search unit is set to the lowest range that includes both the *starts* and *stops* values. Refer to force_v or force_i for output value, range, and compliance relationships.

  Also, after the search is finished, the search unit is set to the *starts* value.

- *ranges*

  This sets the measurement range for the sense unit.

  The *ranges* should be greater than the *targets*. If *ranges* is 0 or is less than the *targets*, it will be set to lowest range that includes the *targets*. If the sense unit is HRSMU and you wish to use 10 pA or 100 pA range, you will need to execute set_rangemode to set FULL_AUTORANGE_ON before executing this function.

- *search_comps*

  This sets the compliance of the search unit.

When using the measurement mode, if you wish to set the compliance, execute the force_v or force_i function for the sense unit before executing p_set_lsearch.

When using the compliance mode, the sense unit compliance is set to the *targets*.

- *delay*

  This is the time between the output from the search unit and measurement by the sense unit. This is not an array. A common *delay* time will be used for all search outputs of each step.

NOTE

To Set the Parameters for Step Skipping Method

To set the parameters for step skipping method, use the p_set_skip function.

Example

```
p_set_lsearch(1, 4, searchports, senseports, modes, starts, stops, steps,
targets, sense_ranges, search_comps, hold);
```

See Also

- p_search_iv
- p_rsearch_iv
- p_set_sync (for search measurements)
- p_set_skip
- p_set_post_mode
- p_set_stop_mode

p_set_post_mode

Revision	D.05.12 or later
Control	SMU

This function sets the post state mode for multi-channel sweep/search/sampling measurements set by P_set_iv, P_set_lsearch, or P_ivt_set_iv.

The P_set_iv, P_set_lsearch, or P_ivt_set_iv function must be executed *before* this function.

Synopsis

```
int p_set_post_mode (int mode);
```

Argument	Range Restrictions/Description
mode	1: Set to the measurement start state. 2: Set to the same state as disable_port. (0 V / 100 μA compliance) (Note1) 3: Set to the bias voltage or current.(Note 2 and 3)

Note 1 Valid only for linear search and sweep measurements. If sampling measurement, an error occurs.

Note 2 Valid only for sampling measurement. If linear search or sweep measurement, an error occurs.

Note 3 Effective on software revision D.05.13 or later.

p_set_skip

Revision	D.05.12 or later
Control	SMU

This function sets the parameters for step skipping method of parallel linear search measurements. The other parameters of the linear search measurements are set by the p_set_lsearch function.

This function must be executed after p_set_lsearch.

Synopsis

```
int p_set_skip (int n, int skips[n], int skip_backs[n], double sb_delays[n]);
```

Argument	Range Restrictions/Description
n	Specify the number of ports (1 to 8).
skips	Pointer to an integer array that determines large step sizes. 1 to 20000
skip_backs	Pointer to an integer array that determines skip back sizes after coarsely finding the target values. 0 to 100
sb_delays	Pointer to a double array that specifies delay time values before starting fine search. 0 to 65.5350 [s] Resolution: 100 μs

Using step skipping method makes linear search faster. This step skipping method is possible when there is only one sense unit for a search unit. Step skipping performs the following:

1. Search unit starts to sweep with large steps until sense unit can approximate the target value.

   Size of large step is calculated from *skips* and *steps*, which are specified by p_set_lsearch, as follows:

skips × *steps*

2. Search unit goes back a specified number of large steps (*skip_backs*).
3. After waiting for *sb_delays*, search unit starts to sweep with small steps (*steps*) until it finds the target value accurately. For a single search unit, only one *sb_delays* value can be used.

Example

```
p_set_lsearch(1, 4, searchports, senseports, modes, starts, stops, steps,
targets, sense_ranges, search_comps, delay);
p_set_skip(4, skips, skip_backs, sb_delays);
p_search_iv(searches, statuses, senses);
```

See Also

- p_set_lsearch

p_set_stop_mode

Revision	D.05.12 or later
Control	SMU

This function sets the automatic stop mode for multi-channel sweep/search/sampling measurements set by P_set_iv, P_set_lsearch, or P_ivt_set_iv.

The P_set_iv, P_set_lsearch, or P_ivt_set_iv function must be executed *before* this function.

Synopsis

```
int p_set_stop_mode (int mode);
```

Argument	Range Restrictions/Description
mode	1: No automatic stop. 2: Stop compliance or abnormal status channel only. 3: Stop all channels of group (Note 1) that contains compliance or abnormal status channel. 4: Stop all channels if there is compliance or abnormal status channel.

Note 1 A group is a reference to all channels associated with a particular search channel For details, see p_set_lsearch.

p_set_sync (for synchronous search measurements)

Revision	D.05.12 or later
Control	SMU

p_set_sync can be called *after* p_set_lsearch to set up parallel synchronous search measurements. p_set_lsearch sets up the search port, and p_set_sync sets up the synchronous search port.

The search is started by p_search_iv or p_rsearch_iv.

You must set the source mode (VS or IS) of synchronous search port to the same source mode as search port set by p_set_lsearch by using force_i or force_v.

For synchronous staircase *sweep* measurements, refer to next section: "p_set_sync (for synchronous sweep measurements)".

Synopsis

```
int p_set_sync (int n, int ports[n], int modes[n], double offsets[n], double
                ratios[n], double compliances[n], double power_compliances[n]);
```

Argument	Range Restrictions/Description	
n	Specify the number of ports (1 to 8).	
ports	Pointer to an integer array that specifies SMU port addresses. You can specify the port addresses directly or specify pin numbers that are connected to the ports:	
	port addresses	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin numbers	1 to 49
modes	Pointer to an integer array that specifies the polarity of the synchronization. You can choose from the following macros: POS_SYNC (= 0) NEG_SYNC (= 1)	

Argument	Range Restrictions/Description
offsets	Pointer to a double array that specifies offsets used to calculate the synchronous values: Synchronous search port is SMU in VS mode [V]: -100 to 100 or -200 to 200 (HPSMU) Synchronous search port is SMU in IS mode [A]: -0.1 to 0.1 or -1 to 1 (HPSMU)
ratios, compliances, power_comps	Dummy parameters. Ignored for search measurements. Used only for sweep measurements.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

In the following explanation of parameters, array names are used to refer to the corresponding array element unless otherwise noted.

- *ports*

  To specify the synchronous search port (SMU), use either the port address of SMU, or the pin number of any measurement pin connected to the SMU port.

  The size of *ports* array must be the same as the size specified in p_set_lsearch.

  Use the same array element number for both a sync search port set by p_set_sync and the corresponding primary search port set by p_set_lsearch. For example, a primary search port and its sync search port can by specified as searchports[2] and syncports[2]. Only one sync search port can be specified for a primary search port.

  When there are no sync search ports for some primary search ports, use 0 as the array elements for the related sync ports array (*ports*). In this case, other array elements (*modes* and *offsets* elements) for the ports are ignored.

- *modes* and *offsets*

  The *modes* and *offsets* parameters set the output of the synchronous search port, as follows:

  - *Mode* = POS_SYNC:
    Synchronous search port output = *offsets* + search port output

- *Mode* = NEG_SYNC:
  Synchronous search port output = *offsets* – search port output

The output range of the synchronous search port is the same as the range for search port.

The specified *offsets* must not cause S (see following) to be greater than ±100V (±200V) or ±0.1A (±1A), depending on the SMU type. S determines the maximum compliance you can specify as follows.

For p_set_lsearch: S=MAX(|*start*±*offset*|, |*stop*±*offset*|)

Table 2-33 In VS mode (MPSMU, HRSMU)

S (in V)	Maximum |Compliance| in mA
$0 < \|S\| \leq 20$	≤ 100
$20 < \|S\| \leq 40$	≤ 50
$40 < \|S\| \leq 100$	≤ 20

Table 2-34 In VS mode (HPSMU)

S (in V)	Maximum |Compliance| in mA
$0 < \|S\| \leq 14$	≤ 1000
$14 < \|S\| \leq 20$	≤ 700
$20 < \|S\| \leq 40$	≤ 350
$40 < \|S\| \leq 100$	≤ 125
$100 < \|S\| \leq 200$	≤ 50

Table 2-35 In IS mode (MPSMU, HRSMU)

S (in mA)	Maximum |Compliance| in V
0 < | S | ≤ 20	≤ 100
20 < | S |≤ 50	≤ 40
50 < | S |≤ 100	≤ 20

Table 2-36 In IS mode (HPSMU)

S (in mA)	Maximum |Compliance| in V
0 < | S | ≤ 50	≤ 200
50 < | S | ≤ 125	≤ 100
125 < | S | ≤ 350	≤ 40
350 < | S | ≤ 700	≤ 20
700 < | S | ≤1000	≤ 14

where compliance is the same as before this function executed.

Example

```
p_set_sync(3,ports,modes,offsets,NULL,NULL,NULL);
```

See Also

- p_set_lsearch
- p_search_iv
- p_search_iv2
- p_rsearch_iv

p_set_sync (for synchronous sweep measurements)

Revision	D.05.12 or later
Control	SMU

p_set_sync can be called *after* p_set_iv to set up parallel synchronous sweep measurements. p_set_iv sets up the primary sweep port, and p_set_sync sets up the synchronous sweep port.

A synchronous sweep port can be set up for staircase sweep measurements (p_set_iv)

The synchronous sweep measurement is started by p_sweep_miv or p_rsweep_miv function.

The output mode of the synchronous sweep port 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 p_set_iv.

For synchronous *search* measurements, refer to the previous section: "p_set_sync (for synchronous search measurements)".

Synopsis

```
int p_set_sync (int n, int ports[n], int modes[n], double offsets[n], double
                ratios[n], double compliances[n], double power_compliances[n]);
```

Argument	Range Restrictions/Description	
n	Specify the number of ports (1 to 8).	
ports	Pointer to an integer array that specifies SMU synchronous sweep port addresses. You can specify the port addresses directly or specify pin numbers that are connected to the ports:	
	port addresses	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin numbers	1 to 49

Argument	Range Restrictions/Description
modes	Pointer to an integer array that specifies the polarity of the synchronization. Choose from one of the following macros: POS_SYNC (= 0) NEG_SYNC (= 1)
offsets	Pointer to a double array that specifies offsets used to calculate the synchronous values. For voltage synchronous source: numeric expression [V]: –200 to 200 or –400 to 400 (HPSMU) For current synchronous source: numeric expression [A]: –0.2 to 0.2 or –2 to 2 (HPSMU)
ratios	Pointer to a double array that specifies the ratios used to calculate the synchronous values. If a negative ratio is desired, use NEG_SYNC for the *modes* value. Numeric expression: 0.01 to 10 Resolution: 0.01
compliances	Pointer to a double array that specifies the voltage or current compliance values on the synchronous ports. REMAIN macro (Note 2) or For voltage synchronous source: numeric expression [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) For current synchronous source: numeric expression [V]: –100 to 100 or –200 to 200 (HPSMU)
power_compliances	Pointer to a double array that specifies the limit of the power (voltage × current being forced and measured) applied to each port. Numeric expression [W]: 0.001 to 2 or 0.001 to 14 (HPSMU) Resolution: 0.001 0 means no power compliance. See description in p_set_iv function.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1.
REMAIN means the previous compliance setting if source mode of synchronous SMU is same as the previous mode. Otherwise:
If source mode was VS mode and is now IS mode, compliance is set to 20 V.
If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

In the following explanation of parameters, array names are used to refer to the corresponding array element unless otherwise noted.

- *ports*

  To specify the SMU, use either the port address of SMU, or the pin number of any measurement pin connected to the SMU port.

  The size of *ports* array must be the same as the size specified in p_set_iv.

  Use the same array element number for both the sync sweep port set by p_set_sync and the corresponding primary sweep port set by p_set_iv. For example, a primary sweep port and its sync sweep port can by specified as sweepports[2] and syncports[2]. Only one sync sweep port can be specified for a primary sweep port.

  When there are no sync sweep ports for some primary sweep ports, use 0 as the array elements for the related sync ports array (*ports*). In this case, other array elements (such as *modes* and *offsets* elements) for the ports are ignored.

- *modes*, *offsets*, and *ratios*

  These parameters are used to calculate the start and stop values of the synchronous sweep port as follows:

  - *modes* = POS_SYNC:

    $sync\_start = offsets + ratios \times primary\_start$
    $sync\_stop = offsets + ratios \times primary\_stop$

  - *modes* = NEG_SYNC:

    $sync\_start = offsets - ratios \times primary\_start$
    $sync\_stop = offsets - ratios \times primary\_stop$

  Where *primary_start* and *primary_stop* are the *start* and *stop* values specified in p_set_iv.

The specified *offsets* and *ratios* must not cause *sync_start* or *sync_stop* to be greater than ±100V (±200V) or ±0.1A (±1A) depending on the SMU type.

The number of steps for the synchronous sweep port is the same as specified by *number* in p_set_iv.

- *compliances*

The output range used and the allowable voltage *compliances* or current *compliances* settings for the synchronous output port depend on the calculated *sync_start* and *sync_stop* voltages as follows:

Table 2-37 Allowable current *compliances* Settings (MPSMU and HRSMU)

Start and Stop Values [V]	Allowable Compliance
0 ≤ |*sync_start* and *sync_stop*| ≤ 20 V	0 to ±100 mA
20 < |*sync_start* and *sync_stop*| ≤ 40 V	0 to ±50 mA
40 < |*sync_start* and *sync_stop*| ≤ 100 V	0 to ±20 mA

Table 2-38 Allowable current *compliances* Settings (HPSMU)

Start and Stop Values [V]	Allowable Compliance
0 ≤ |*sync_start* and *sync_stop*| ≤ 14 V	0 to ±1 A
14 < |*sync_start* and *sync_stop*| ≤ 20 V	0 to ±700 mA
20 < |*sync_start* and *sync_stop*| ≤ 40 V	0 to ±350 mA
40 < |*sync_start* and *sync_stop*| ≤ 100 V	0 to ±125 mA
100 < |*sync_start* and *sync_stop*| ≤ 200 V	0 to ±50 mA

Table 2-39 Allowable voltage *compliances* Settings (MPSMU and HRSMU)

Start and Stop Values	Allowable Compliance
0 ≤ |*sync_start* and *sync_stop*| ≤ 20 mA	0 to ±100 V
20 mA < |*sync_start* and *sync_stop*| ≤ 50 mA	0 to ±40 V
50 mA < |*sync_start* and *sync_stop*| ≤ 100 mA	0 to ±20 V

Table 2-40 Allowable voltage *compliances* Settings (HPSMU)

Start and Stop Values	Allowable Compliance
0 ≤ |*sync_start* and *sync_stop*| ≤ 50 mA	0 to ±200 V
50 mA < |*sync_start* and *sync_stop*| ≤ 125 mA	0 to ±100 V
125 mA < |*sync_start* and *sync_stop*| ≤ 350 mA	0 to ±40 V
350 mA < |*sync_start* and *sync_stop*| ≤ 700 mA	0 to ±20 V
700 mA < |*sync_start* and *sync_stop*| ≤ 1 A	0 to ±14 V

- *power_compliances*

  Refer to the p_set_iv function.

Example

```
p_set_sync(3, syncports, modes, offsets, ratios, comps, p_comps);
p_sweep_miv(3, 3, ports, ranges, meas, swp1, swp2);
```

See Also

- p_set_iv
- p_sweep_miv
- p_rsweep_miv

p_set_vth

Revision	D.05.12 or later
Control	SMU

This function sets the parameters for parallel FET gate-source threshold voltage measurements that use linear search with step skipping and least squares method. Use the p_measure_vth function to start the threshold voltage measurement. Vth measurement requires two independent SMU channels, and therefore up to four parallel Vth measurements can be performed.

Synopsis

```
int p_set_vth (int para_mode, int n, int gate_ports[n], int drain_ports[n],
               double vg_starts[n], double vg_stops[n], double vg_steps[n], double
               id_starts[n], double id_measranges[n], int vg_skips[n], int
               vg_stepbacks[n], double sb_delays[n], double gate_comps[n], double
               delay, int chan_types[n]);
```

Argument	Range Restrictions/Description	
para_mode	Specify parallel or sequential measurement mode. 1: parallel measurement mode (Note 1) 0: sequential measurement mode	
n	Specify how many Vth measurements (1 to 4) will be performed.	
gate_ports (Note 2)	Pointer to an integer array that specifies gate SMU port addresses. You can specify the port addresses directly or specify pin numbers that are connected to the ports:	
	port addresses	You can specify the following macros (or values (Note 3)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin numbers	1 to 49

Argument	Range Restrictions/Description	
drain_ports	Pointer to an integer array that specifies drain SMU port addresses. You can specify the port addresses directly or specify pin numbers that are connected to the ports:	
	port addresses	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin numbers	1 to 49
vg_starts	Pointer to a double array that specifies gate voltage values to start searching *id_starts* values. –100 to 100 [V] or –200 to 200 [V] (HPSMU)	
vg_stops	Pointer to a double array that specifies maximum gate voltage values of search for *id_start* values. –100 to 100 [V] or –200 to 200 [V] (HPSMU)	
vg_steps	Pointer to a double array that specifies step sizes of gate voltage. 0 < |*vg_steps* value|≤ 100 0 < |*vg_steps* value|≤ 200 (HPSMU)	
id_starts	Pointer to a double array that specifies drain current values at which to start fine search for maximum gradient points. –0.1 to 0.1 [A] or –1 to 1 [A] (HPSMU)	
id_measranges	Pointer to a double array that specifies current measurement ranges of drain SMUs. –0.1 to 0.1 [A] or –1 to 1 [A] (HPSMU)	
vg_skips	Pointer to an integer array that specifies factors used to determine large step sizes. 1 to 20000	
vg_skipbacks	Pointer to an integer array that specifies numbers of large steps to go back after finding *id_starts* values. 0 to 100	

Argument	Range Restrictions/Description
sb_delays	Pointer to a double array that specifies delay time values between skip back and start of fine search for maximum gradient point. 0 to 65.5350 [s] with 0.0001 resolution
gate_comps	Pointer to a double array that specifies gate current compliance values. –0.1 to 0.1 [A]
delay	Delay time for measurement at each step. 0 to 65.5350 [s] with 0.0001 resolution
chan_types	Pointer to an integer array that specifies FET channel types. N type: 0 or positive integer. P type: negative integer.

Note 1 Use parallel measurement mode unless sequential measurement is required, i.e. for debugging purposes.

Note 2 The size of each array must be equal to the size of *gate_ports* array. Array elements that have the same element number are for one Vth measurement setup.

Note 3 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

In the sequential measurement mode, execution is almost the same as executing measurement by set_vth multiple times. For the parallel mode, high-speed ADC is used for the parallel measurements even when high-resolution ADC is selected.

After p_set_vth sets the search parameters, p_measure_vth performs the threshold voltage measurements as follows. (Here, array names are used to refer to the corresponding array elements.)

1. Gate SMU starts to sweep with large steps from *vg_starts* until drain SMU finds *Id_starts*.

   Size of large step is calculated from *vg_skips* as follows:

   vg_skips × *vg_steps*

id_starts must be at drain current before FET reaches threshold. To make measurement faster, specify *id_starts* to be close to the threshold, but before the threshold.

2. Gate SMU goes back a specified number of large steps (*vg_skipbacks*)
3. After waiting for *sb_delays*, gate SMU starts to sweep with small steps (*vg_steps*) until the gradient of Vgs-Ids or Vgs-√{Ids} curve is maximum.
4. The X-intercept of line tangent to this maximum gradient point is returned to p_measure_vth as the threshold voltage. Refer to p_measure_vth.

Example

See Also

- p_measure_vth
- p_set_lsearch
- p_set_skip

p_status_miv

Revision	D.05.12 or later
Control	SMU

Returns the status of the data measured by p_sweep_miv.

Synopsis

```
int p_status_miv (int n, int ports[n], int statuses[n][x], double times[n][x]);
```

x: number of sweep steps

Argument	Range Restrictions/Description
n	Specify the number of ports: 1 to 8.
ports	Specify a pointer to an integer array that contains the SMU port addresses or pin numbers connected to the ports.
statuses	Specify a pointer to an integer array in which to return the measurement statuses.
times	Specify a pointer to a double array in which to return the time stamp values.

- *ports* array

  To specify the measurement ports, specify port addresses of SMUs or pin numbers of any measurement pin connected to the SMU ports.

- *statuses* array and *times* array

  Measurement status and time stamp values are returned to elements of these arrays that correspond to the ports specified in *ports* array. For status and time stamp, refer to the port_status_t function.

NOTE

When an SMU has reached compliance during parallel multi-channel measurement

When an SMU of primary sweep port, synchronous sweep port, or measurement port has reached compliance, the SMU reached compliance returns the status value 2, which means “this unit has reached compliance”, and other SMUs within the corresponding sweep returns the status value 1, which means “another unit has reached compliance”.
The returned status of SMUs of the parallel multi-channel sweep measurement that is not related to the corresponding sweep are not affected.

When an SMU that is not used in the parallel multi-channel sweep measurement has reached compliance during the parallel multi-channel sweep measurement, all SMUs of the parallel multi-channel sweep measurement are returned the status value 1 (“another unit has reached compliance”).

See also

- port_status_t
- sweep_miv

p_sweep_miv

Revision	D.05.12 or later
Control	SMU

This function can perform the following parallel multichannel sweep type measurements:

- staircase sweep
- synchronous staircase sweep

This function cannot set the measurement mode, To set desired measurement mode for each *measurement only* port, use the force_i or force_v function before the p_sweep_miv function.

- For Staircase Sweep Measurements

  This function performs parallel multichannel staircase sweep measurements according to the measurement parameters set by the p_set_iv function.

- For Synchronous Staircase Sweep Measurements

  This function performs parallel multichannel synchronous staircase sweep measurements according to the measurement parameters set by the p_set_iv and p_set_sync functions.

Synopsis

```
int p_sweep_miv (int n, int m, int ports[n], double ranges[n], double
                 meas_vals[n][x], double source[m][x], double sync[m][x]);
```

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

Argument	Range Restrictions/Description
n	Specify how many ports (1 to 8) will perform measurement. This value determines the size of the following arrays: *ports*, *ranges*, and the primary index of *meas_vals*.
m	Specify the number of sweep source ports. This value determines the size of the primary index of *source* and *sync*.

Argument	Range Restrictions/Description
ports	Specify a pointer to an integer array that determines the measurement ports. This array must have *n* elements. The array should contain the SMU port addresses or SWM pin numbers that are connected to ports.
ranges	Specify a pointer to double array that contains the measurement range for each port. This array must have *n* elements. For Auto range mode, specify 0.
meas_vals	Specify a pointer to a double array in which to return the measurement values.
source	Specify a pointer to a double array in which to return the sweep source data (which is determined by set_iv). If you don't need this data, 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). If you don't need this data, specify NULL. If you don't use the set_sync function, this argument must be NULL.

- *ports* array

  To specify the measurement ports for p_sweep_miv, specify port addresses of SMUs or pin numbers of any measurement pin connected to the SMU ports. This array should have *n* elements.

- *ranges* array

  Assign range values to *ranges* array elements that correspond to each *ports* array element. This array should have *n* elements.

  For voltage measurement, refer to the measure_v function for the measurement range.

  For current measurement, refer to the measure_i function for the measurement range.

- *meas_vals* array

  The measurement values for each measurement port are returned to the *meas_vals* array. Hence, the size of *meas_vals* array should be *n*×*number* for a single sweep, or *n*×*2*×*number* for a double sweep. Where, *number* is the number of steps specified in p_set_iv function.

- *source* and *sync* arrays

  The source values output by the sweep source (set by p_set_iv) are returned to the *source* array.

  Also, for synchronous sweep measurement, the values output by the synchronous source (set by p_set_sync) are returned to the *sync* array.

  The size of *source* and *sync* arrays should be *number* for a single sweep, or *2*×*number* for a double sweep. Where, *number* is specified in p_set_iv function.

Use the status_miv function to get the measurement statuses.

Offline Mode In offline mode, this function returns values as follows:

- Measurement values are calculated and returned to *meas_vals* array elements as follows:

$$\text{nth measurement value} = \frac{basevalue}{\text{number of steps} - 1} \times (n - 1)$$

  Where: number of *steps* is set by the p_set_iv function, and *basevalue* is determined as follows:

Table 2-41 For current measurements

Meas. Range	*basevalue*	Polarity
0	I compliance	Same as SMU *output value*
≠ 0	Range value	

Table 2-42 For voltage measurements

Meas. Range	*basevalue*	Polarity
0	V compliance	Same as SMU *output value*
≠ 0	Range value	

- *source* array

  Sweep source values determined by p_set_iv are returned to the *sources* array.

- *sync* array

  Synchronous sweep source values determined by p_set_sync are returned to the *sync* array.

Example

```
err = p_sweep_miv(n, m, ports, ranges, meas_vals, sweep, NULL);
```

See Also

- p_set_iv
- p_set_sync (for sweep measurements)
- p_rsweep_miv
- p_status_miv

Q

This section describes the C functions that begin with Q.

query_cmu

Revision	C.04.00 or later
Control	–

This function returns the unit type of capacitance measurement subsystem (CMS) installed in the system.

Synopsis

```
int query_cmu (int *type);
```

Argument	Description/Default
type	Pointer to int variable in which to return the unit type. Type of the capacitance measurement unit is returned to the *type* variable as follows: 0: The capacitance measurement unit does not exist. 1: Capacitance Measurement Unit (CMU) 2: High-Speed Capacitance Measurement Unit (HSCMU)

query_pgu

Revision	E.05.15 or later
Control	–

This function returns the type of pulse generator unit (PGU) connected to the specified port.

Synopsis

```
int query_pgu (int port, int *type);
```

Argument	Range Restrictions/Description	
port	Specify the port connected to the PG. You can specify the HF, THF, AUX, or PSI port address or pin number that currently connects to the port.	
	port address	You can specify the following values (Note 1): 20201 to 20206, 20211 to 20213, 20251 to 20260, 20101 to 20108
	pin number	1 to 48 (Note 2)
type	Pointer to int variable in which to return the pulse generator type. PGU_NOT_EXIST(=0): No PGU is installed. PGU_SPGU(=4): SPGU PGU_81150A(=5): PGU (81150A)	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use PORT function or MACRO instead.

Note 2 A specified measurement pin must be connected to an HF, THF, or AUX port.

R

This section describes the C functions that begin with R.

range_mode

Revision	D.05.12 or later
Control	SMU

This function can change the auto-ranging operation for the current measurement.

Synopsis

```
int set_smu_ch (int port, int mode, int ratio);
```

Argument	Range Restrictions/Description	
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008). Specify 0 for all SMU ports.
	pin number	1 to 49
	0	All SMUs.

Argument	Range Restrictions/Description
mode	Auto-ranging mode: 1: Normal operation 2: Change upper boundary for range change 3: Change upper and lower boundaries for range change
ratio	Ratio value to calculate the boundary for range change: 11 to 100 This parameter is ignored if *mode* = 1.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port*

  The *port address* of SMU or the *pin number* of any measurement pin connected to the SMU port must be specified.

- *mode* and *ratio*

  - If *mode*=2, the operation of auto-ranging is changed as follows:

    If measured data ≥ *current1*, the range changes up after measurement.

  - If *mode*=3, the operation of auto-ranging is changed as follows:

    If measured data ≤ *current2*, the range changes down after measurement and if measured data ≥ *current1*, the range changes up after measurement.

  Where *current1* and *current2* are calculated from *ratio* by the following formula:

  current1 = measurement range × *ratio*/100

  current2 = measurement range × *ratio*/1000

  For the normal operation, set *mode*=1. Then, *ratio* is ignored.

report_event

Revision	A.01.00 or later
Control	SMU, SWM

This function returns the event status.

Synopsis

```
int report_event (int *event);
```

Argument	Range Restrictions/Description
event	Pointer to integer variable in which to return event status.

This function returns an integer to indicate present event status. You can use the associated macros in your program to determine which status was returned.

Event	Event Value	Description
No event	0	
High voltage	EVENT_HIGH_VOLTAGE(=1)	SMU outputs voltage of 40 V or more.
Power fail	EVENT_PWF(=2)	Testhead power failed.
I/V over	EVENT_OVER_IV(=4)	Output voltage or current of SMU is over its limit.
Fixture open	EVENT_FIXOPEN(=8)	Fixture is open.
INTLK open	EVENT_INTLOPEN(=16)	Interlock is open.

reset_timestamp

Revision	B.01.20 or later
Control	SMU

This function sets the time origin of the measurement time stamp at the execution of first force_i, force_v, force_meas, or force_meas_t function after this function.

Synopsis

```
int reset_timestamp (void);
```

Don't execute init_system between reset_timestamp and force_i, force_v, or force_meas functions, or init_system function resets the timer for the measurement time stamp and the setting by this function is ineffective.

rfs_calc_coef1

Revision	D.05.10 or later
Control	RFU

This function searches the data in the memory measured by rfs_sol_cal or rfs_cmu_cal for the specified measurement conditions and calculates the calibration coefficient of SOL probe calibration for 1 port S parameter measurement, Cp/G measurement, Cp/G-V measurement, or Cp/G-f measurement for the same measurement conditions.

Synopsis

```
int rfs_calc_coef1 (int port, int pin, int sw_type, double f_start, double
                    f_stop, int points, double power, double bw, int alg);
```

Argument	Range Restrictions/Description
port	RFU where the substrate structures are measured by rfs_sol_cal or rfs_cmu_cal.
pin	Measurement pin connected to *port*.
sw_type	Measurement condition for the stimulus frequency sweep type to be calculated. Log sweep: LOG_F (=–1) Linear sweep: LINEAR_F (=1)
f_start	Measurement condition for the sweep start of stimulus frequency to be calculated. (Note 1)
f_stop	Measurement condition for the sweep stop of stimulus frequency to be calculated. (Note 1)
points	Measurement condition for the number of sweep steps to be calculated. (Note 1)
power	Measurement condition for the maximum signal power level of stimulus to be calculated.
bw	Measurement condition for the IF bandwidth of receiver to be calculated.
alg	Always COEF_SOL (=1).

Note 1 For probe calibration on Cp/G or Cp/G-V, specify 1 for *points* and the stimulus frequency for *f_stop*. *f_start* is dummy parameter.

The calculated calibration coefficient is stored in the memory and is automatically used by rfs_measure_s1, rfs_measure_cmu, rfs_sweep_cf, or rfs_sweep_cv for the same measurement conditions.

Example

```
/* COEF1 */
for (i = 0; i < 2 ; i++){
  rfs_calc_coef1(RFU11,pin1,type[i],fs[i],fe[i],pts[i],pwr[i],bw[i],COEF_SO
L);
}
```

See Also

- rfs_sel_calkit
- rfs_sol_cal, rfs_cmu_cal
- rfs_measure_s1, rfs_measure_cmu, rfs_sweep_cv, rfs_sweep_cf
- rfs_store_coef, rfs_load_coef_r

rfs_calc_coef2

Revision	D.05.10 or later
Control	RFU

This function searches the data in the memory measured by rfs_sol_cal and rfs_thru_cal for the specified measurement conditions and calculates the calibration coefficient of SOLT probe calibration for 2 port S parameter measurement for the same measurement conditions.

Synopsis

```
int rfs_calc_coef2 (int port1, int pin1, int port2, int pin2, int sw_type,
                    double f_start, double f_stop, int points, double power, double
                    bw, int alg);
```

Argument	Range Restrictions/Description
port1, *port2*	RFU ports where the substrate structures are measured by rfs_sol_cal and rfs_thru_cal.
pin1, *pin2*	Measurement pin connected to *port1* and pin connected to *port2*.
sw_type	Measurement condition for the stimulus frequency sweep type to be calculated. Log sweep: LOG_F (=–1) Linear sweep: LINEAR_F (=1)
f_start	Measurement condition for the sweep start of stimulus frequency to be calculated.
f_stop	Measurement condition for the sweep stop of stimulus frequency to be calculated.
points	Measurement condition for the number of sweep steps to be calculated.
power	Measurement condition for the maximum signal power level of stimulus to be calculated.
bw	Measurement condition for the IF bandwidth of receiver to be calculated.
alg	Always COEF_SOLT (=2).

The calculated calibration coefficient is stored in the memory and is automatically used by rfs_measure_s2 for the same measurement conditions.

Example

```
/* COEF2 */
for (i = 0; i < 2 ; i++){
  rfs_calc_coef2(RFU11,pin1,RFU12,pin2,type[i],fs[i],fe[i],pts[i],pwr[i],bw
[i],COEF_SOLT);
}
```

See Also

- rfs_sel_calkit
- rfs_sol_cal, rfs_thru_cal
- rfs_measure_s2
- rfs_store_coef, rfs_load_coef_r

rfs_clear_coef

Revision	D.05.10 or later
Control	RFU

This function discards the current calibration coefficient data stored and managed in the memory.

Synopsis

```
int rfs_clear_coef (void);
```

After this function is executed, probe calibration for RF measurement is unavailable until an applicable calibration coefficient is loaded into the memory by rfs_calc_coef1, rfs_calc_coef2, or rfs_load_coef_r.

rfs_cmu_cal

Revision	D.05.10 or later
Control	RFU

This function performs 1 port S parameter measurement on an open, short, or load substrate structure for probe calibration of RF Cp/G and Cp/G-V measurement. The measured data is stored into the memory.

Synopsis

```
int rfs_cmu_cal (int port, int cal, double f, double power, double bw);
```

Argument	Range Restrictions/Description	
port	RFU to execute the 1 port S parameter measurement on the substrate structure. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404) RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
cal	Type of substrate structure to be measured. Open: CAL_OPEN (=1) Short: CAL_SHORT (=2) Load: CAL_LOAD (=3)	
f	Stimulus frequency. 1.0e+7 to 2.0e+10 [Hz]	
power	Maximum signal power level of stimulus. Allowable range depends on the stimulus frequency. See Table 2-43.	
bw	IF bandwidth of receiver. 1 Hz to 40 kHz	

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

Table 2-43 Stimulus Frequency and Allowable Maximum Power Level

Stimulus Frequency	Maximum Power Level
10 MHz to 45 MHz	–87 dBm to 2 dBm
45 MHz to 10 GHz	–87 dBm to 3 dBm
10 GHz to 20 GHz	–86 dBm to 1 dBm

To make the SOL probe calibration effective on the RF Cp/G or Cp/G-V measurement, the rfs_calc_coef1 function must be executed before the measurement to calculate and prepare the correction coefficient from the data measured and stored into the memory by this function.

See Also

- rfs_sel_calkit
- rfs_calc_coef1
- rfs_measure_cmu, rfs_sweep_cv

rfs_deembed_s1

Revision	D.05.10 or later
Control	RFU

This function sends the S11 values of open and short structure substrates measured by 1 port S parameter measurement to TIS server as the correction data for de-embedding.

If you execute 1 port S parameter measurement with de-embedding, the correction data for de-embedding with an identical set of measurement conditions must be sent to TIS server before executing measurement by rfs_measure_s1.

Synopsis

```
int rfs_deembed_s1 (int port, COMPLEX s11o[], COMPLEX s11s[]);
```

Argument	Range Restrictions/Description	
port	Port that measured 1 port S parameter of open and short structure. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302) RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
s11o[]	S11 value of open structure.	
s11s[]	S11 value of short structure.	

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

If only open de-embedding has been attempted, specify NULL for *s11s* parameter.

See Also

- rfs_measure_s1

rfs_deembed_s2

Revision	D.05.10 or later
Control	RFU

This function sends the S parameter values of open and short structure substrates measured by 2 port S parameter measurement to TIS server as the correction data for de-embedding at 2 port S parameter measurement.

If you execute a 2 port S parameter measurement with de-embedding, the correction data for de-embedding with an identical set of measurement conditions must be sent to TIS server before executing measurement by rfs_measure_s2.

Synopsis

```
int rfs_deembed_s2 (int port1, int port2, COMPLEX s11o[], COMPLEX s21o[],
                    COMPLEX s12o[], COMPLEX s22o[], COMPLEX s11s[], COMPLEX s21s[],
                    COMPLEX s12s[], COMPLEX s22s[]);
```

Argument	Range Restrictions/Description	
port1, *port2*	Ports that measured 2 port S parameter of open and short structures. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
s11o[], *s21o[]*, *s12o[]*, *s22o[]*	S parameter values of open structure substrate.	
s11s[], *s21s[]*, *s12s[]*, *s22s[]*	S parameter values of short structure substrate.	

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

If only open de-embedding has been attempted, specify NULL to *s11s*, *s21s*, *s12s*, and *s22s*.

See Also · rfs_measure_s2

rfs_ft

Revision	D.05.10 or later
Control	RFU

This function calculates fT from S parameters obtained by the rfs_measure_s2 function as follows:

$$fT = |H21| \times f0$$

Synopsis

```
int rfs_ft (double f0, COMPLEX s11, COMPLEX s21, COMPLEX s12, COMPLEX
            s22, double *ft);
```

Argument	Description
f0	Frequency for calculating fT.
s11	s11 measured at *f0*.
s21	s21 measured at *f0*.
s12	s12 measured at *f0*.
s22	s22 measured at *f0*.
ft	Return variable for the calculated fT.

See Also · rfs_measure_s2

rfs_init_cal

Revision	D.05.10 or later
Control	RFU

This function performs initialization for probe calibration.

This function must be executed before executing the functions for RF probe calibration, which are rfs_sel_calkit, rfs_sol_cal, rfs_thru_cal, rfs_cmu_cal, rfs_calc_coef1, rfs_calc_coef2, rfs_store_coef, rfs_load_coef_r, and rfs_clear_coef.

Synopsis

```
int rfs_init_cal (void);
```

Example

```
rfs_init_cal();
rfs_sel_calkit("sampleISS.csv");
```

See Also

- rfs_sel_calkit
- rfs_sol_cal, rfs_thru_cal
- rfs_cmu_cal
- rfs_calc_coef1, rfs_calc_coef2
- rfs_store_coef, rfs_load_coef_r
- rfs_clear_coef

rfs_load_coef_r

Revision	D.05.10 or later
Control	RFU

This function loads the calibration coefficient data from the specified file and, and also returns the applicable measurement path and conditions read from the file.

Synopsis

```
int rfs_load_coef_r (char *filename, int *port1, int *pin1, int *port2, int
                     *pin2, int *sw_type, double *f_start, double *f_stop, int *points,
                     double *power, double *bw, int *alg);
```

Argument	Description	Value Range
filename	The calibration coefficient data file to be read. (Note 1)	
port1	Return variable for the stimulus port.	
pin1	Return variable for the pin connected to the *port1*.	
port2	Return variable for the response port.	
pin2	Return variable for the pin connected to the *port2*.	
sw_type	Return variable for the sweep type of stimulus frequency.	
	Log sweep: Linear sweep:	LOG_F (=–1) LINEAR_F (=1)
f_start	Return variable for the sweep start frequency.	
f_stop	Return variable for the sweep stop frequency.	
points	Return variable for the number of sweep steps.	
power	Return variable for the maximum signal power level.	

Argument	Description	Value Range
bw	Return variable for the IF bandwidth.	
alg	Return variable for the type of calibration algorithm.	
	SOL: SOLT:	COEF_SOL(=1) COEF_SOLT(=2)

Note 1 File can be specified using a relative path from /var/opt/hp4070/diag/vna directory.

- If the calibration coefficient data in the file is for SOL calibration, COEF_SOL(=1) is returned to *alg* and no values are returned to *port2* and *pin2*.
- If the calibration coefficient data in the file is for the Cp/G spot measurement and Cp/G-V sweep measurement:
  - stimulus frequency is returned to *f_start* and *f_stop*
  - COEF_SOL(=1) is returned to *alg*
  - 0 is returned to *port2*, and *pin2*, and 1 is returned to *points*.

rfs_measure_cmu

Revision	D.05.10 or later
Control	RFU

This function performs RF Cp/G measurement under the conditions set by rfs_set_cmu and rfs_set_average.

Synopsis

```
rfs_measure_cmu (int port, double *c, double *g, int mode, COMPLEX *s11);
```

Argument	Range Restrictions/Description	
port	RFU to execute the 1 port S parameter measurement. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
c	Return variable for the calculated capacitance value (Cp).	
g	Return variable for the calculated conductance value (G).	
mode	De-embedding mode.	
	DEMB_OFF (=0) DEMB_OPEN (=1) DEMB_SHORT (=2)	No de-embedding Open de-embedding Open and Short de-embedding
s11	Return variable for the measured S11 parameter value.	

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

NOTE

If DEMB_OPEN or DEMB_SHORT is specified for the *mode* parameter, execute the rfs_deembed_s1() function before this function to measure the open structure or the open and short structures under the exact same measurement conditions as for the correction data for de-embedding.

This function performs 1 port S parameter measurement under the conditions set by rfs_set_cmu and rfs_set_average, and returns the capacitance (Cp) and conductance values calculated from the measured S11 value.

Example

```
rfs_set_cmu(p1,freq,pwr,bw);
rfs_set_average(p1,avg);
rfs_measure_cmu(p1,&C_val,&G_val,DEMB_OFF,NULL);
```

See Also

- rfs_set_cmu, rfs_set_average
- rfs_deembed_s1

rfs_measure_s1, rfs_meas_s1_0

Revision	D.05.10 or later
Control	RFU

These functions perform 1 port S parameter measurement under the conditions set by rfs_set_s and rfs_set_average.

Synopsis

```
int rfs_measure_s1 (int port, double f[], COMPLEX s[], int mode);
int rfs_meas_s1_0 (int port, double f[], COMPLEX s[]);
```

Argument	Range Restrictions/Description	
port	RFU to execute the 1 port S parameter measurement. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
f[]	Return array for the measuring frequencies.	
s[]	Return array for the measurement results (S parameter).	
mode	De-embedding mode.	
	DEMB_OFF (=0) DEMB_OPEN (=1) DEMB_SHORT (=2)	No de-embedding Open de-embedding Open and Short de-embedding

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

The rfs_meas_s1 function performs 1 port S parameter measurement and returns the calibrated and de-embedded (optional) measurement result if correction data exists on the memory.

The rfs_meas_s1_0 function performs the same measurement, but returns the measurement result without calibration and de-embedding.

NOTE

If DEMB_OPEN or DEMB_SHORT is specified for the *mode* parameter, execute the rfs_deembed_s1 function before this function to measure the open structure or the open and short structures under the exact same measurement conditions as for the correction data for de-embedding.

Example

```
 :
 :
connect_pin(RFU11,p1);
 :
 :
rfs_set_s(p1,type,f_start,f_stop,pts,pwr,bw);
rfs_set_average(p1,avg);
rfs_measure_s1(p1,f_val,s_val,mode);
```

See Also

- rfs_set_s
- rfs_set_average
- rfs_deembed_s1

rfs_measure_s2, rfs_meas_s2_0

Revision	D.05.10 or later
Control	RFU

These functions perform 2 port S parameter measurement under the conditions set by rfs_set_s and rfs_set_average.

The rfs_measure_s2 function performs 2 port S parameter measurement and returns the calibrated and de-embedded (optional) measurement results.

The rfs_meas_s2_0 function performs the same measurement, but returns the measurement results without calibration and de-embedding.

Synopsis

```
int rfs_measure_s2 (int port1, int port2,double f[], COMPLEX s11[], COMPLEX
            s21[], COMPLEX s12[], COMPLEX s22[], int mode);

int rfs_meas_s2_0 (int port1, int port2,double f[], COMPLEX s11[], COMPLEX
            s21[], COMPLEX s12[], COMPLEX s22[]);
```

Argument	Range Restrictions/Description	
port1, *port2*	RFUs of port1 and port2 to execute the 2 port S parameter measurement. Both RFU ports must be in the same PNA. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
f[]	Return array for the measuring frequencies are returned.	
s11[], *s21[]*, *s12[]*, *s22[]*	Return array for the measurement results (S parameters).	

Argument	Range Restrictions/Description	
mode	De-embedding mode.	
	DEMB_OFF (=0)	No de-embedding
	DEMB_OPEN (=1)	Open de-embedding
	DEMB_SHORT (=2)	Open and Short de-embedding

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

NOTE

If DEMB_OPEN or DEMB_SHORT is specified for the *mode* parameter, execute the rfs_deembed_s2 function before this function to measure the open structure or the open and short structures under the exact same measurement conditions as for the correction data for de-embedding.

Example

```
 :
 :
connect_pin(RFU11,p1);
connect_pin(RFU12,p2);
 :
 :
rfs_set_s(p1,type,f_start,f_stop,pts,pwr,bw);
rfs_set_average(p1,avg);
rfs_measure_s2(p1,p2,f_val,s11,s21,s12,s22,mode);
```

See Also

- rfs_set_s
- rfs_set_average
- rfs_deembed_s2

rfs_sel_calkit

Revision	D.05.10 or later
Control	RFU

This function reads the values of the calibration substrate for probe calibration on RF measurement from a file.

Synopsis

```
int rfs_sel_calkit (char *file);
```

Argument	Description
file	Specify a file name (Note 1) of standard values for calibration substrate.

Note 1 File must be saved under the /etc/opt/hp4070/diag/vna directory.

Example

```
rfs_init_cal();
rfs_sel_calkit("sampleISS.csv");
```

See Also

- rfs_init_cal
- rfs_sol_cal
- rfs_thru_cal
- rfs_cmu_cal
- rfs_calc_coef1
- rfs_calc_coef2
- rfs_store_coef
- rfs_load_coef_r
- rfs_clear_coef

rfs_set_adc

Revision	E.05.15 or later
Control	E5262A

This function sets the A/D converter of the E5262A used for the RF subsystem.

Synopsis

```
int rfs_set_adc (int mode, double value);
```

Argument	Range Restrictions/Description
mode	Integration mode. INTEG_MANUAL(=0), INTEG_SHORT(=1), INTEG_MEDIUM(=2), or INTEG_LONG(=3)
value	For *mode*=0, set the number of averaging samples. Available values are 0 to 1023. For *mode*=1, set the α value of the following formula. The number of averaging samples given by the formula is set to the E5262A. Available values are 0 to 1023. The initial value in the formula means the number of averaging samples automatically set by the E5262A and cannot be changed by users. Number of averaging samples=initial value $\times \alpha$ For *mode*=2, setting value is ignored. Set any value. The E5262A sets the integration time to 1 power line cycle, 1/50 or 1/60 seconds. For *mode*=3, set the number of power line cycles for the integration time. Available values are 0 to 100. If *value*=0 is set, this function automatically changes the value to 1 for *mode*=0 or 1, or 16 for *mode*=3.

Example

```
rfs_set_adc(mode,avg_dc);
```

rfs_set_average

Revision	D.05.10 or later
Control	RFU

This function sets the averaging factor for RF measurement by RFU of the specified PNA.

Synopsis

```
int rfs_set_average (int port, int av);
```

Argument	Range Restrictions/Description	
port	PNA that contains the RFU to execute the RF measurement. (Note 1) Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 2)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409), PNA11(20509), PNA12(20510).
	pin number:	101 to 105, 201 to 205
av	Averaging factor. If 0 is specified, averaging is set to OFF. 0 to 1024	

Note 1 The specified settings are applicable for all RFU ports of PNA.

Note 2 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

Example

```
rfs_set_s(p1,type,f_start,f_stop,pts,pwr,bw);
rfs_set_average(p1,avg);
```

See Also

- rfs_set_cmu
- rfs_set_s

rfs_set_cmu

Revision	D.05.10 or later
Control	RFU

This function sets the stimulus and receiver conditions for RF Cp/G measurement by RFU of the specified PNA.

To start RF Cp/G spot measurement, use the rfs_measure_cmu function.

To execute RF Cp/G-V sweep measurement, use the rfs_set_cv function to set the voltage sweep condition and use the rfs_sweep_cv function to start the RF Cp/G-V sweep measurement.

Synopsis

```
int rfs_set_cmu (int port, double f, double power, double bw);
```

Argument	Range Restrictions/Description	
port	PNA that contains the RFU to execute the RF capacitance measurement. (Note 1) Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 2)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409), PNA11(20509), PNA12(20510).
	pin number:	101 to 105, 201 to 205
f	Measurement frequency. 1.0e+7 to 2.0e+10 [Hz]	
power	Maximum signal power level of the stimulus. Allowable range depends on the stimulus frequency. See Table 2-44.	
bw	IF bandwidth of the receiver. (Note 3) 1 [Hz] to 40 [kHz]	

Note 1 The specified settings are applicable for all RFU ports of PNA.

Note 2 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

Note 3 Reducing the IF receiver bandwidth reduces the effect of random noise on a measurement. However, narrower IF bandwidths cause longer sweep times.

Table 2-44 Stimulus Frequency and Allowable Maximum Power Level

Stimulus Frequency	Maximum Power Level
10 MHz to 45 MHz	–87 dBm to 2 dBm
45 MHz to 10 GHz	–87 dBm to 3 dBm
10 GHz to 20 GHz	–86 dBm to 1 dBm

Example

```
rfs_set_cmu(p1,freq,pwr,bw);
rfs_set_average(p1,avg);
```

See Also

- rfs_set_average
- rfs_measure_cmu
- rfs_set_cv
- rfs_sweep_cv

rfs_set_cv

Revision	D.05.10 or later
Control	RFU

This function specifies parameters for DC bias voltage sweep for RF Cp/G-V sweep measurement.

To specify other measurement conditions for RF Cp/G-V sweep measurement, use the rfs_set_cmu and rfs_set_average functions.

Synopsis

```
rfs_set_cv (int port, double start, double stop, int step, double hold,
            double delay, int stop_mode);
```

Argument	Range Restrictions/Description	
port	Measurement port. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
start	Start voltage for the sweep. –7 to 7 [V] (with RF matrix), –40 to 40 [V] (direct output)	
stop	Stop voltage for the sweep. –7 to 7 [V] (with RF matrix), –40 to 40 [V] (direct output)	
step	Number of sweep steps. 2 to 1001	
hold	Hold time for the first sweep step. 0 to 650.000 [s] resolution = 1 ms	

Argument	Range Restrictions/Description
delay	Delay time for each sweep step. 0 to 650.000 [s] resolution = 1 ms
stop_mode	Specify whether to continue or stop if abnormal condition occurs during measurement. Chose one from the following macros: COMP_CONT (= 1): Continue sweep even if abnormal condition occurs. COMP_STOP (= 2): Stop sweep if abnormal condition occurs. (Dummy data is returned.)

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

NOTE

Current Compliance

The current compliance of the DC bias source is set as follows:

With RF Matrix	140 mA
Direct Output (0 to 20 V) (Note 1)	200 mA
Direct Output (up to 40 V) (Note 1)	50 mA

Note 1 MAX(*start*, *stop*)

Example

```
rfs_set_cmu(p1,freq,pwr,bw);
rfs_set_cv(p1,start,stop,step.hold,delay,COMP_CONT);
rfs_sweep_cv(p1,C_val,G_val,V_val,stats,DEMB_OFF,NULL);
```

See Also

- rfs_set_average
- rfs_set_cmu
- rfs_sweep_cv

rfs_set_s

Revision	D.05.10 or later
Control	RFU

This function sets the stimulus and receiver conditions for 1 port or 2 port S-parameter measurement by RFU of the specified PNA.

To start 1 port S-parameter measurement, use the rfs_measure_s1 function.

To start 2 port S-parameter measurement, use the rfs_measure_s2 function.

Synopsis

```
int rfs_set_s (int port, int sw_type, double f_start, double f_stop, int
               points, double power, double bw);
```

Argument	Range Restrictions/Description	
port	PNA for the stimulus setting. (Note 1) Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 2)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409), PNA11(20509), PNA12(20510).
	pin number:	101 to 105, 201 to 205
sw_type	Stimulus frequency sweep type. Log sweep: LOG_F (=–1) Linear sweep: LINEAR_F (=1)	
f_start	Start frequency for the stimulus frequency sweep. 1.0e+7 to 2.0e+10 [Hz]	
f_stop	Stop frequency for the stimulus frequency sweep. 1.0e+7 to 2.0e+10 [Hz]	

Argument	Range Restrictions/Description
points	Number of sweep steps. 2 to 16001
power	Maximum signal power level of the stimulus. Allowable range depends on the value of *f_stop*. See Table 2-45.
bw	IF bandwidth of receiver. (Note 3) 1 [Hz] to 40 [kHz]

Note 1 The specified settings are applicable for all RFU ports of PNA.

Note 2 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

Note 3 Reducing the IF receiver bandwidth reduces the effect of random noise on a measurement. However, narrower IF bandwidths cause longer sweep times.

Table 2-45 ***f_stop* and Allowable Maximum Power Level**

f_stop	Maximum Power Level
10 MHz to 45 MHz	–87 dBm to 2 dBm
45 MHz to 10 GHz	–87 dBm to 3 dBm
10 GHz to 20 GHz	–86 dBm to 1 dBm

Example

```
rfs_set_s(p1,type,f_start,f_stop,pts,pwr,bw);
rfs_set_average(p1,avg);
```

See Also

- rfs_set_average
- rfs_measure_s1
- rfs_measure_s2

rfs_sol_cal

Revision	D.05.10 or later
Control	RFU

This function performs 1 port S parameter measurement on an open, short, or load substrate structure for probe calibration of 1 port S parameter measurement, 2 port S parameter measurement, and RF Cp/G-f measurement. The measured data is stored into the memory.

Synopsis

```
int rfs_sol_cal (int port, int cal_type, int sw_type, double f_start, double
                 f_stop, int points, double power, double bw);
```

Argument	Range Restrictions/Description	
port	RFU to execute the 1 port S parameter measurement on the substrate structure. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
cal_type	Type of substrate structure to be measured. Open: CAL_OPEN (=1) Short: CAL_SHORT (=2) Load: CAL_LOAD (=3)	
sw_type	Stimulus frequency sweep type. Log sweep: LOG_F (=–1) Linear sweep: LINEAR_F (=1)	
f_start	Start frequency for the stimulus frequency sweep. 1.0e+7 to 2.0e+10 [Hz]	

Argument	Range Restrictions/Description
f_stop	Stop frequency for the stimulus frequency sweep. 1.0e+7 to 2.0e+10 [Hz]
points	Number of sweep steps. 2 to 16001
power	Maximum signal power level of the stimulus. Allowable range depends on the stimulus frequency. See Table 2-46.
bw	IF bandwidth of the receiver. 1 [Hz] to 40 [kHz]

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

Table 2-46 Stimulus Frequency and Allowable Maximum Power Level

Stimulus Frequency	Maximum Power Level
10 MHz to 45 MHz	–87 dBm to 2 dBm
45 MHz to 10 GHz	–87 dBm to 3 dBm
10 GHz to 20 GHz	–86 dBm to 1 dBm

To make the SOL probe calibration effective on the 1 port S parameter measurement or RF Cp/G-f measurement, the rfs_calc_coef1 function must be executed before the measurement to calculate and prepare the correction coefficient from the data measured and stored into the memory by this function.

To make the SOLT probe calibration effective on the 2 port S parameter measurement, the rfs_calc_coef2 function must be executed before the measurement to calculate and prepare the correction coefficient from the data measured and stored into the memory by this function and the rfs_thru_cal function.

Example

```
int type[] = {1, -1};
double fs[] = {300.0E+06, 500.0E+06};
double fe[] = {1.0E+09, 5.0E+09};
int    pts[] = {101, 201};
```

```
double pwr[] = {-10.0, -20,0};
double bw[]  = {30.0E+03, 30.0E+03};
int i;
 :
 :
 :
/* OPEN */
for (i = 0; i < 2 ; i++){
  rfs_sol_cal(RFU11,1,type[i],fs[i],fe[i],pts[i],pwr[i],bw[i]);
}
for (i = 0; i < 2 ; i++){
  rfs_sol_cal(RFU12,1,type[i],fs[i],fe[i],pts[i],pwr[i],bw[i]);
}
 :
 :
 :
/* SHORT */
for (i = 0; i < 2 ; i++){
  rfs_sol_cal(RFU11,2,type[i],fs[i],fe[i],pts[i],pwr[i],bw[i]);
}
for (i = 0; i < 2 ; i++){
  rfs_sol_cal(RFU12,2,type[i],fs[i],fe[i],pts[i],pwr[i],bw[i]);
}
 :
 :
 :
/* LOAD */
for (i = 0; i < 2 ; i++){
  rfs_sol_cal(RFU11,3,type[i],fs[i],fe[i],pts[i],pwr[i],bw[i]);
}
for (i = 0; i < 2 ; i++){
  rfs_sol_cal(RFU12,3,type[i],fs[i],fe[i],pts[i],pwr[i],bw[i]);
}
```

See Also

- rfs_sel_calkit
- rfs_thru_cal
- rfs_calc_coef1, rfs_calc_coef2
- rfs_measure_s1, rfs_measure_s2, rfs_sweep_cf

rfs_store_coef

Revision	D.05.10 or later
Control	RFU

This function saves the current calibration coefficient for the probe calibration under the specified measurement conditions to the specified file.

Synopsis

```
int rfs_store_coef (char *filename, int port1, int pin1, int port2, int
                    pin2, int type, double f_start, double f_stop, int points, double
                    power, double bw, int alg);
```

Argument	Range Restrictions/Description
filename	Name of the file to save calibration coefficient data to. (Note 1)
port1, *port2*	RFU ports where the substrate structures are measured by rfs_sol_cal and rfs_thru_cal.
pin1, *pin2*	Measurement pin connected to *port1* and the pin connected to *port2*.
type	Specify the measurement condition of stimulus frequency sweep type for the calibration coefficient to be saved. Log sweep: LOG_F (=–1) Linear sweep: LINEAR_F (=1)
f_start	Measurement condition for the sweep start of stimulus frequency for the calibration coefficient to be saved.
f_stop	Measurement condition for the sweep stop of stimulus frequency for the calibration coefficient to be saved.
points	Measurement condition for the number of sweep steps for the calibration coefficient to be saved.
power	Measurement condition for the maximum signal power level of the stimulus for the calibration coefficient to be saved.
bw	Measurement condition for the IF bandwidth of the receiver for the calibration coefficient to be saved.
alg	Calibration type for the calibration coefficient to be saved. if SOL calibration, specify COEF_SOL (=1) if SOLT calibration: specify COEF_SOLT (=2)

Note 1 File can be specified using a relative path from /var/opt/hp4070/diag/vna/0 directory for PNA0 or /var/opt/hp4070/diag/vna/1 directory for PNA1.

If COEF_SOL(=1) is specified for *alg*, *port2* and *pin2* are ignored.

If you wish to save a calibration coefficient for RF Cp/G spot and Cp/G-V sweep measurements:

- specify COEF_SOL(=1) for *alg*
- specify 1 for *points*
- specify the stimulus frequency value for *f_stop*
- *port2*, *pin2*, and *f_start* are dummy parameters and are ignored.

NOTE

File Format of Saved Calibration Coefficient File

The following table explains the meaning of the characters at the line head:

#	Comment line
!1	Beginning of a data set
@	Measurement conditions in CSV (Comma Separated Value) format, for example: *@points, port1, port2, pin1, pin2, type, alg, f_start, f_stop, power, bw*
[0-9]	Calibration coefficient data for each frequency step in CSV format, for example: · if *alg*=COEF_SOL *step_no, freq, ES11, ER11, ED11* · if *alg*=COEF_SOLT *step_no, freq, ES11, ER11, ED11, ES22, ER22, ED22, EL21, ET21, EX21, EL12, ET12, EX12*

See Also

- rfs_calc_coef1, rfs_calc_coef2
- rfs_load_coef_r

rfs_sweep_cf

Revision	D.05.10 or later
Control	RFU

This function performs RF Cp/G-f sweep measurement under the conditions set by rfs_set_s and rfs_set_average.

Synopsis

```
int rfs_sweep_cf (int port, double c[], double g[], double f[], int mode,
                  COMPLEX s11[]);
```

Argument	Range Restrictions/Description	
port	Measurement port. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
c[]	Return array for the calculated capacitance (Cp) values.	
g[]	Return array for the calculated conductance (G) value.	
f[]	Return array for the measurement frequencies of the sweep.	
mode	De-embedding mode.	
	DEMB_OFF (=0) DEMB_OPEN (=1) DEMB_SHORT (=2)	No de-embedding Open de-embedding Open and Short de-embedding
s11[]	Return array for the measured S11 parameter values.	

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

NOTE

If DEMB_OPEN or DEMB_SHORT is specified for the *mode* parameter, execute the rfs_deembed_s1() function before this function to measure the open structure or, the open and short structures under the exact same measurement conditions as for the correction data for de-embedding.

This function performs 1 port S parameter measurement under the conditions set by rfs_set_s and rfs_set_average, and returns the capacitance (Cp) and conductance values at each stimulus frequency calculated from the measured S11 value.

Example

```
 :
 :
connect_pin(RFU11,p1);
 :
 :
rfs_set_s(p1,type,f_start,f_stop,pts,pwr,bw);
rfs_set_average(p1,avg);
rfs_sweep_cf(p1,C_val,G_val,f_val,DEMB_OFF,NULL);
```

See Also

- rfs_set_s
- rfs_set_average
- rfs_deembed_s1

rfs_sweep_cv

Revision	D.05.10 or later
Control	RFU

This function performs RF Cp/G-V sweep measurement under the conditions set by rfs_set_cmu, rfs_set_cv, and rfs_set_average.

Synopsis

```
int rfs_sweep_cv (int port, double c[], double g[], double v[], int status[],
                  int mode, COMPLEX s11[]);
```

Argument	Range Restrictions/Description	
port	Measurement port. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
c[]	Return array for the calculated capacitance (Cp) values.	
g[]	Return array for the calculated conductance (G) values.	
v[]	Return array for the bias voltages of the sweep.	
status[]	Return array for the measurement status of each voltage step.	
mode	De-embedding mode.	
	DEMB_OFF (=0) DEMB_OPEN (=1) DEMB_SHORT (=2)	No de-embedding Open de-embedding Open and Short de-embedding
s11[]	Return array for the measured S11 parameter values at each voltage step.	

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

NOTE

If DEMB_OPEN or DEMB_SHORT is specified for the *mode* parameter, execute the rfs_deembed_s1 function before this function to measure the open structure or, the open and short structures under the exact same measurement conditions as for the correction data for de-embedding.

By using the RFU, this function sets the DC bias voltage and performs 1 port S parameter measurement under the conditions set by rfs_set_cmu and rfs_set_average at each voltage step set by rfs_set_cv, and returns the capacitance (Cp) and conductance values calculated from the measured S11 values.

NOTE

Capacitance measurement by this function is insensitive to the capacitance calibration TISs for DC measurement (such as open_corr84).

Example

```
 :
 :
connect_pin(RFU11,p1);
 :
 :
rfs_set_cmu(p1,freq,pwr,bw);
rfs_set_average(p1,avg);
rfs_set_cv(p1,start,stop,step.hold,delay,COMP_CONT);
rfs_sweep_cv(p1,C_val,G_val,V_val,stats,DEMB_OFF,NULL);
```

See Also

- rfs_set_cv
- rfs_set_cmu
- rfs_set_average
- rfs_deembed_s1

rfs_thru_cal

Revision	D.05.10 or later
Control	RFU

This function performs 2 port S parameter measurement on the thru substrate structure for probe calibration of 2 port S parameter measurement. The measured data is stored into the memory.

Synopsis

```
int rfs_thru_cal (int port1, int port2, int sw_type, double f_start, double
                  f_stop, int points, double power, double bw);
```

Argument	Range Restrictions/Description	
port1, *port2*	RFUs of port1 and port2 to execute the 2 port S parameter measurement. Both RFU ports must be in the same PNA. Specify the port address directly or specify a pin number that is connected to the port:	
	port address:	Specify the following macros (or values (Note 1)): RFU11 (20301), RFU12 (20302), RF11 (20401), RF21 (20402), RF31 (20403), RF41 (20404), RF12 (20406), RF22 (20407), RF32 (20408), RF42 (20409).
	pin number:	101 to 105, 201 to 205
sw_type	Stimulus frequency sweep type. Log sweep: LOG_F (=–1) Linear sweep: LINEAR_F (=1)	
f_start	Start frequency for the stimulus frequency sweep. 1.0e+7 to 2.0e+10 [Hz]	
f_stop	Stop frequency for the stimulus frequency sweep. 1.0e+7 to 2.0e+10 [Hz]	
points	Number of sweep steps. 2 to 16001	

Argument	Range Restrictions/Description
power	Maximum signal power level of the stimulus. Allowable range depends on the stimulus frequency. See Table 2-47.
bw	IF bandwidth of the receiver. 1 [Hz] to 40 [kHz]

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

Table 2-47 Stimulus Frequency and Allowable Maximum Power Level

Stimulus Frequency	Maximum Power Level
10 MHz to 45 MHz	–87 dBm to 2 dBm
45 MHz to 10 GHz	–87 dBm to 3 dBm
10 GHz to 20 GHz	–86 dBm to 1 dBm

To make the SOLT probe calibration effective on the 2 port S parameter measurement, the rfs_calc_coef2 function must be executed before the measurement to calculate and prepare the correction coefficient from the data measured and stored into the memory by this function and the rfs_sol_cal function.

Example

```
/* THRU */
for (i = 0; i < 2 ; i++){
  rfs_thru_cal(RFU11,RFU12,type[i],fs[i],fe[i],pts[i],pwr[i],bw[i]);
}
```

See Also

- rfs_sel_calkit
- rfs_sol_cal
- rfs_calc_coef2
- rfs_measure_s2

rfs_z0

Revision	D.05.10 or later
Control	RFU

This function sets the characteristic impedance to be used in TIS functions for RF measurement, such as RF Cp/G measurement, de-embedding, and S parameter conversion. The initial system value is 50 Ω.

Synopsis

```
int rfs_z0 (double z0);
```

Argument	Description
z0	Characteristic impedance. [Ω]

rfs_z0_r

Revision	D.05.10 or later
Control	RFU

This function returns the current characteristic impedance Z0 for RF measurement. See rfs_z0.

Synopsis

```
int rfs_z0_r (double *z0);
```

Argument	Description
z0	Specify the pointer which is used to return the current characteristic impedance.

round_level_pg

Revision	B.03.01 with patch P.03.01.2002.0331 or later
Control	PGU, SPGU

This function determines if the pulse amplitude and pulse base values specified in set_level_pg are rounded to be sent to the pulse generator (PG) connected to the specified port.

Synopsis

```
int round_level_pg (int port, int on_off);
```

Argument	Range Restrictions/Description	
port	Specify the port connected to the PG. You can specify the HF, THF, AUX, or PSI port address or pin number that currently connects to the port.	
	port address	You can specify the following values (Note 1):
		0, 20201 to 20206, 20211 to 20213, 20251 to 20260, 20101 to 20108. Specify 0 for all PG installed.
	pin number	1 to 48
on_off	0: Rounding Disable ≠ 0: Rounding Enable	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use PORT function instead.

This function determines whether the pulse amplitude and pulse base values specified in set_level_pg are rounded in 100 mV resolution or not for specified PG port.

If the integer value other than 0 is specified to *on_off* parameter, set_level_pg function rounds the pulse amplitude and pulse base values for the specified PG port in 100 mV resolution.

The initial system setting is "Rounding Disable".

rsearch_iv

Revision	A.01.00 or later
Control	SMU

This function can perform *real-time* search measurements according to the conditions set by the set_bsearch, set_bdv_search, or set_lsearch function.

Real-time means that values are returned after each search step.

The rsearch_iv function returns measurement data and status after each search step. To use rsearch_iv in a measurement program, put rsearch_iv in a `for` loop. To abort the search measurement, use the rsearch_stop function. Refer to the programming examples.

The search_iv function performs similar search, but *not* in real-time. Only the final values are returned after search completion.

If you include a set_sync function between set_bsearch or set_lsearch and the rsearch_iv function, the rsearch_iv function performs a search measurement using the synchronous search unit specified by set_sync.

The force_i or force_v function cannot be used between the rsearch_iv and set_bsearch, set_bdv_search, or set_lsearch functions.

Keysight recommends to execute rsearch_stop after completing *real-time* search measurement.

Synopsis

```
int rsearch_iv (double *search, int *status, double *sense, double
                *time_stamp);
```

Argument	Range Restrictions/Description
search	Specify a pointer to a double variable in which to return the present force value of search port.
status	Specify a pointer to an integer variable in which to return the measurement status code. One of following integers is returned to *status*. You can use the associated macros in your program to determine which status was returned: NORMAL_MEAS (=0) SEARCH_DOMAIN (=1) SEARCH_ABNORMAL (=2) SEARCH_FAILED (=4) SEARCH_STOPPED (=8) SEARCH_COMP_OTHER (=16) SEARCH_COMP (=32) SEARCH_OVERFLOW (=64) SEARCH_OSC (=128)
sense	Specify a pointer to a double variable in which to return the measurement value of sense port.
time_stamp	Specify a pointer to a double in which to return the time stamp of present measurement value.

- *search* and *sense*

  The rsearch_iv function obtains SMU data during a search measurement and returns the following after each search step:

  - Present source value of search unit is returned to *search* variable.
  - Measurement value of sense unit is returned to *sense* variable.

- *status*

  The rsearch_iv function returns an integer to *status*. You can use the following macros in your program to determine which status was returned:

Status Macro	Condition
NORMAL_MEAS (=0)	Measured value is normal.
SEARCH_DOMAIN (=1)	Search value not found (between *start* and *stop*, or *min* and *max*).
SEARCH_ABNORMAL (=2)	Search aborted.
SEARCH_FAILED (=4)	Sense value is not within specified accuracy, or an abnormal compliance loop occurred.
SEARCH_STOPPED (=8)	Search measurement was completed.
SEARCH_COMP_OTHER (=16)	Unit other than sense unit reached compliance.
SEARCH_COMP (=32)	Sense unit reached compliance.
SEARCH_OVERFLOW (=64)	A/D converter overflow.
SEARCH_OSC (=128)	SMU is oscillating

- *time_stamp*

  Accumulated time in seconds at measurement start (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) is returned to *time_stamp* variable.

  If reset_timestamp function has been executed after init_system function or `/opt/hp4070/bin/hp4070 -start` command, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode This function returns values to the *search*, *status*, and *sense* variables as follows:

- For binary search measurements (set_bsearch function):

  The *search* variable is set to the average value of *min* and *max* specified in the set_bsearch function. The *status* variable is always set to 0, which in the offline mode means that the search value was found. The *sense* variable is set to the *target* specified in the set_bsearch function.

- For linear or breakdown voltage search measurements (set_lsearch or set_bdv_search function):

The *search* variable is set to the average value of *start* and *stop* specified in the set_lsearch function. The *status* variable is always set to 0, which in the offline mode means that the search value was found. The *sense* variable is set to the *target* specified in the set_lsearch function.

See also

- set_lsearch, set_bsearch, set_bdv_search
- set_sync (for synchronous search measurements)
- rsearch_stop
- search_iv

rsearch_stop

Revision	A.01.00 or later
Control	SMU

This function stops the *real-time* search measurements.

Synopsis

```
int rsearch_stop (void);
```

This function is used to abort real-time search measurements that were started by rsearch_iv function.

Also, Keysight recommends to execute this function after completing *real-time* search measurement, even if the measurement is completed successfully.

Example

```
rsearch_stop();
```

See Also

- rsearch_iv

rsweep_iv, rsweep_ivt

Revision	A.01.00 or later
Control	SMU

The rsweep_iv and rsweep_ivt functions are the same, except rsweep_ivt also returns a time stamp value.

These functions can perform the following *real-time* sweep type measurements:

- staircase sweep
- synchronous staircase sweep
- pulsed sweep
- staircase sweep with pulsed bias measurements

Real-time means that a measurement value can be returned after each sweep step.

The rsweep_iv function returns measurement data after each sweep measurement step is finished, so the measurement results can be plotted on the screen as each measurement is performed. To use rsweep_iv in a measurement program, put rsweep_iv in a `for` loop. To abort the sweep measurement, use the rsweep_stop function. Refer to the programming examples.

The sweep_iv function performs similar sweeps, but *not* in real-time. All measurement values are returned after the sweep is completed, not after each sweep step.

Keysight recommends to execute rsweep_stop after completing real-time sweep measurement.

The rsweep_iv function performs measurements according to measurement parameters set in the following functions:

Measurement	Function Name
Staircase sweep measurements	set_iv
Pulsed sweep measurements	set_piv
Staircase sweep with pulsed bias measurements	set_iv, set_pbias

If set_sync was also specified, rsweep_iv performs *real-time, synchronous* staircase sweep measurements using the primary sweep port (specified by set_iv or set_piv), the synchronous sweep port (specified by set_sync), and pulse bias port (if set_pbias was specified).

Synopsis

```
int rsweep_iv (int port, int mode, double range, double *measure, int
               *status, double *source, double *sync);
int rsweep_ivt (int port, int mode, double range, double *measure, int
               *status, double *source, double *sync, double *time_stamp);
```

Argument	Range Restrictions/Description	
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address:	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number:	1 to 49
mode	Specify whether *port* will perform a voltage or current measurement. Use one of the following macros: V_MEAS (= 1) I_MEAS (= 2)	

Argument	Range Restrictions/Description
range	Specify the measurement range. 0 for auto-ranging mode or –0.1 to 0.1 (MPSMU and HRSMU) or –1 to 1 (HPSMU) for current measurement or –100 to 100 (MPSMU and HRSMU) or –200 to 200 (HPSMU) for voltage measurement
measure	Specify a pointer to a double variable in which to return the present measurement result.
status	Specify a pointer to integer variable in which to return the measurement status code. One of following integers is returned to *status*. You can use the associated macros in your program to determine which status was returned. NORMAL_MEAS (= 0) DCS_COMP_OTHER (= 1) DCS_COMP (= 2) DCS_OSC (= 3) DCS_OVERFLOW (= 4) DCS_SWP_STOPPED(= 5)
source	Specify a pointer to double variable in which to return the present sweep source value (which is determined by set_iv or set_piv function).
sync	Specify a pointer to double variable in which to return the present secondary sweep source value (which is determined by set_sync function).
time_stamp	Only for rsweep_ivt function. A pointer to a double where the measurement time stamp of the corresponding point is stored.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port*

  To specify the SMU measurement port for rsweep_iv, use either the *port address* of SMU or *pin number* of any SWM pin connected to the SMU port.

- *mode* and *range*

  If *mode* is V_MEAS (voltage measurement), refer to the measure_v function for *range*.

  If *mode* is I_MEAS (current measurement), refer to the measure_i function for *range*.

- *measure*

  The present measurement value is returned to *measure*.

- *status*

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

Status Value	Condition
NORMAL_MEAS (0)	Normal measurement end
DCS_COMP_OTHER (1)	Another unit reached compliance
DCS_COMP (2)	This unit reached compliance
DCS_OSC (3)	This unit is oscillating
DCS_OVERFLOW (4)	DC source measurement overflow
DCS_SWP_STOPPED (5)	Sweep was aborted by compliance condition. (stop mode = 2 in set_iv or set_piv)

- *source*

  Present sweep source value is returned to *source*. This value is determined by the set_iv or set_piv function.

  If you do not need the sweep source value, specify NULL to discard it (*sync* must also be NULL).

- *sync*

  Present source value of the synchronous sweep port is returned to *sync*.

  This value is determined by the set_sync function. If the synchronous sweep port is not specified by set_sync, NULL must be specified to this argument.

- *time_stamp*

  If you use rsweep_ivt, the accumulated time at measurement start in seconds (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) is returned to *time_stamp*.

  If reset_timestamp function has been executed after init_system function or `/opt/hp4070/bin/hp4070 -start` command, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

If the *stop_mode* = COMP_STOP of the set_iv or set_piv function, rsweep_iv measurements are aborted if a measurement SMU reaches compliance. If this occurs before a measurement loop ends, rsweep_iv returns 9999999.99999.

If a measurement loop exceeds the number of sweep points, –9999999.99999 is returned when the sweep ends.

Offline Mode In offline mode, this function returns values as follows:

- Measurement values are calculated and returned to *measurement* as follows:

$$\text{nth measurement value} = \frac{basevalue}{\text{number of steps} - 1} \times (n - 1)$$

  Where: *number of steps* is set by the set_iv or set_piv function, and *basevalue* is determined as follows:

Table 2-48 For current measurements

Meas. Range	*basevalue*	Polarity
0	I compliance	Same as SMU *output value*
≠ 0	Range value	

Table 2-49 For voltage measurements

Meas. Range	*basevalue*	Polarity
= 0	2.0 V	Always positive
≠ 0	V compliance	Same as SMU *output value*

- *status* is always set to 0.
- *time stamp* variable is accumulated time in seconds from January 1, 1970, or the first force_i, force_v, or force_meas function after reset_timestamp function.
- *source* and *sync*

  The primary sweep source value determined by set_iv or set_piv is returned to *source*.

  The synchronous sweep source value determined by set_sync is returned to *sync*.

Example

```
int err;

……

for (i=0; i<number; i++){
err = rsweep_iv(12, I_MEAS, range, &meas, &stat, &sweep, NULL);

……

}
```

See Also

- rsweep_miv
- rsweep_stop
- set_iv
- set_piv
- set_sync (for sweep measurements)
- set_pbias
- sweep_iv
- sweep_miv

rsweep_miv, rsweep_mivt

Revision	A.01.00 or later
Control	SMU

The rsweep_miv and rsweep_mivt functions are the same, except rsweep_mivt also returns a time stamp value.

These functions can perform the following *real-time, multichannel* sweep type measurements:

- staircase sweep
- synchronous staircase sweep

This function *cannot* set the measurement mode, so you should use force_i or force_v functions to set desired measurement mode for each port before rsweep_miv function.

Note that this function *cannot* be used for pulsed sweep or staircase sweep with pulse bias measurements.

The rsweep_miv function performs *real-time, multichannel* staircase sweep measurements according to measurement parameters set in the set_iv function.

If set_sync is also specified, rsweep_miv can perform *real-time, multichannel synchronous* staircase sweep measurements according to measurement parameters set in the set_iv and set_sync functions.

The rsweep_miv function returns measurement data after each sweep measurement step is finished, so the measurement results can be plotted on the screen as each measurement is performed. To use rsweep_miv in a measurement program, put rsweep_miv in a `for` loop. Refer to the programming examples. To abort the sweep measurement, use the rsweep_stop function.

Synopsis

```
int rsweep_miv (int n, int ports, double ranges[n], double meas_vals[n], int
            statuses[n], double *source, double *sync);

int rsweep_mivt (int n, int ports, double ranges[n], double meas_vals[n],
            int statuses[n], double *source, double *sync, double
            time_stamp[n]);
```

Argument	Range Restrictions/Description
n	Specify how many ports (1 to 8) will perform measurement. The size of the *ports*, *ranges*, *meas_vals*, *statuses*, and *time_stamps* arrays must be greater than or equal to *n*.
ports	Specify the pointer to an integer array that determines the measurement ports. This array must have *n* elements. The array should contain the SMU port addresses or SWM pin numbers that are connected to ports.
ranges	Specify the pointer to double array that contains the measurement range for each port. This array must have *n* elements. For Auto range mode, specify 0.
meas_vals	Specify the pointer to double array in which to return the measurement values for each port. This array must have *n* elements.
statuses	Specify the pointer to integer array in which to return the measurement status code for each measurement port. One of following integers is returned. You can use the associated macros in your program to determine which status was returned. NORMAL_MEAS (= 0) DCS_COMP_OTHER (= 1) DCS_COMP (= 2) DCS_OSC (= 3) DCS_OVERFLOW (= 4) DCS_SWP_STOPPED(= 5)
source	Specify the pointer to double variable in which to return the present sweep source value (which is determined by set_iv). If you don't need this data, specify NULL (*sync* must also be NULL).
sync	Specify the pointer to double variable in which to return the present secondary sweep source value (which is determined by set_sync function). If you don't need this data, specify NULL. If you don't use the set_sync function, this argument must be NULL.
time_stamp	Only for rsweep_mivt function. Specify the pointer to a double array in which to return the time stamp values.

- *n*

  This function starts real-time sweep and can perform up to 8 simultaneous measurements at each sweep step. The *n* argument specifies how many ports will perform measurement. You must declare array arguments for this function as in the following example:

  ```
  int n=3, ports[3], statuses[3];
  double meas_vals[3], ranges[3], time_stamps[3];
  double source, sync;
  ```

  The valid range of *n* is 1 through 8. If only one port is required, use the rsweep_iv function.

- *ports*

  This array determines which ports will perform measurements.

  The array elements should contain the SMU port addresses or SWM pin numbers that are connected to ports as follows:

  port address: You can specify the following macros (or values):

  SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).

  pin number: 1 to 49

- *meas_vals*, *statuses*

  The rsweep_miv function returns measurement values of each port to *meas_vals* array.

  This function returns integers to the *statuses* array to indicate the measurement status. You can use the associated macros in your program to determine which status was returned.

Status Value	Condition
NORMAL_MEAS (0)	Normal measurement end
DCS_COMP_OTHER (1)	Another unit reached compliance
DCS_COMP (2)	This unit reached compliance
DCS_OSC (3)	This unit is oscillating

Status Value	Condition
DCS_OVERFLOW (4)	DC source measurement overflow
DCS_SWP_STOPPED (5)	Sweep was aborted by compliance condition. (stop mode = 2 in set_iv)

- *source*, *sync*

  The rsweep_miv function returns present output value of sweep source to *source*.

  If you do not need to get source values, specify NULL for *source* and *sync* arguments.

  For synchronous sweep measurement, rsweep_miv also returns present output value of synchronous sweep source to *sync*.

  If the synchronous sweep port is not specified by set_sync, NULL must be specified to this argument.

- *time_stamp*

  If you use rsweep_mivt, accumulated time at measurement start for each SMU in seconds (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) is returned to *time_stamp* array.

  If reset_timestamp function has been executed after init_system function or `/opt/hp4070/bin/hp4070 -start` command, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode In offline mode, this function returns values as follows:

- Measurement values are calculated and returned to *meas_vals* array elements as follows:

$$\text{nth measurement value} = \frac{basevalue}{\text{number of steps} - 1} \times (n - 1)$$

  Where: *number of steps* is set by the set_iv function, and *basevalue* is determined as follows:

Table 2-50 **For current measurements**

Meas. Range	*basevalue*	Polarity
0	I compliance	Same as SMU *output value*
≠ 0	Range value	

Table 2-51 **For voltage measurements**

Meas. Range	*basevalue*	Polarity
= 0	2.0 V	Always positive
≠ 0	V compliance	Same as SMU *output value*

- *statuses* array element is always set to 0.
- *time stamp array* element is set to accumulated time in seconds from January 1, 1970, or the first force_i, force_v, or force_meas function after reset_timestamp function.
- *source* and *sync*

  The primary sweep source value determined by set_iv is returned to *source*.

  The synchronous sweep source value determined by set_sync is returned to *sync*.

Example

```
err=set_iv(8, LINEAR_V, 20, vstart, vstop, number, hold, delay, comp, 0.0,
COMP_CONT);
for (i=0; i<number; i++){
err=rsweep_miv(n, ports, ranges, meas_vals, statuses, &sweep, NULL);

}
```

See Also

- rsweep_iv
- rsweep_stop
- set_iv
- set_sync
- sweep_i, sweep_miv

rsweep_stop

Revision	A.01.00 or later
Control	SMU

This function stops *real-time* sweep measurements.

Synopsis

```
int rsweep_stop (void);
```

This function is used to abort the real-time sweep measurements that were started by rsweep_iv or rsweep_miv function.

Also, Keysight recommends to execute this function after completing real-time sweep measurement, even if the measurement completed successfully.

After this function executes, the sweep source SMU is set to the sweep start value, but sweep settings are not changed.

Example

```
err=set_iv(8, LINEAR_V, 20, vstart, vstop, number, hold, delay, comp, 0.0,
COMP_CONT);
for (i=0; i<number; i++) {
err=rsweep_miv(n, ports, ranges, measures, stat, &sweep, NULL);
if(err==-1){
rsweep_stop();
break;
}
rsweep_stop();
}
```

See Also

- rsweep_iv
- rsweep_miv

S

This section describes the C functions that begin with S.

save_pg

Revision	B.02.00 or later
Control	PGU, SPGU

This function saves the present pulse generator (PG) settings to the specified file.

Synopsis

```
int save_pg (char *file_specifier);
```

Argument	Range Restriction/Description
file_specifier	Pointer to a character string that specifies the name of a file where the pulse generator setting data will be stored and may include an absolute path name. Maximum length: 1023 characters.

This function saves the present PG settings to the specified file. The saved file can be loaded using load_pg function, and be used to set the PGs.

For SPGU, only PG mode is supported.

Using save_pg and load_pg functions may enable fast setup of PGs.

search_iv

Revision	A.01.00 or later
Control	SMU

This function performs search measurements according to the conditions set by the set_bsearch, set_bdv_search, or set_lsearch function. After search is completed, the final output value of search unit, and the measurement value of sense unit are returned.

If you include a set_sync function between set_bsearch (or set_lsearch) and the search_iv function, the search_iv function performs a search measurement using the synchronous search unit specified by set_sync.

Breakdown voltage search measurement using set_bdv_search does not support synchronous port set by set_sync. So, if you include a set_sync function between set_bdv_search and the search_iv functions, the search_iv resets the settings set by set_sync function.

The force_i or force_v function cannot be used between the search_iv and set_bsearch (or set_lsearch) functions.

For real-time search, refer to the rsearch_iv function.

NOTE

Differences between search_iv and search_iv2

search_iv2 returns actual measured value to *sense* even if the following measurement status is detected during a search measurement:

- Another unit than sense unit has reached compliance
- Sense unit has reached compliance
- Measurement overflow
- Sense unit is oscillating

search_iv returns –9999999.99999 to *sense* if the above status is detected.

Synopsis

```
int search_iv (double *search, int *status, double *sense);
```

Argument	Range Restrictions/Description
search	Specify a pointer to a double variable in which to return the search value.
status	Specify a pointer to an integer variable in which to return the search measurement status. One of the following integers is returned to *status*. You can use the associated macros in your program to determine which status was returned. NORMAL_MEAS (=0) SEARCH_DOMAIN (= 1) SEARCH_ABNORMAL (= 2) SEARCH_FAILED (= 4)
sense	Specify a pointer to a double variable in which to return the target value measured by the sense unit.

- *search*

  This function performs a search measurement, and returns the search value to the *search* variable. The search value is the value output by search unit when the target value is found by the sense unit.

  If search value is not found, one of the following is returned to *search*:

–9999999.99999	Search value was not found.
9999999.99999	Search measurement was aborted.
–8888888.88888	(For binary search measurements only set_bsearch function) Search value not found. Change the *max* value to increase the difference between *min* and *max*.
–7777777.77777	(For binary search measurements only set_bsearch function) Search value not found. Change the *min* value to increase difference between *min* and *max*.

- *sense*

  The target value measured by the sense unit is returned to the *sense* variable.

  In case of set_bdv_search *breakdown_current* specified by set_bdv_search is returned to the *sense* variable.

- *status*

  The search_iv function returns an integer (0, 1, 2, or 4) to *status*. You can use the following macros in your program to determine which status was returned:

  - For binary search measurements (set_bsearch function):

    NORMAL_MEAS (=0): Search value successfully found.

    SEARCH_DOMAIN (=1): Search value not found (between the *min* and *max* defined in set_bsearch).

    If –7777777.77777 is returned to the *search* variable, change the *min* value to increase difference between *min* and *max*.

    If –8888888.88888 is returned to the *search* variable, change the *max* to increase the difference between *min* and *max*.

    SEARCH_ABNORMAL (=2): Abnormal data.

    SEARCH_FAILED (=4): Sense unit cannot find specified *target* within specified accuracy.

  - For linear or breakdown voltage search measurements (set_lsearch or set_bdv_search function):

    NORMAL_MEAS (=0): Search value successfully found.

    SEARCH_DOMAIN (=1): Search value not found (between the *start* and *stop* values specified in set_lsearch or set_bdv_search).

    SEARCH_ABNORMAL (=2): Abnormal data

    SEARCH_FAILED (=4): Abnormal compliance loop

Offline Mode This function returns values to the *search*, *status*, and *sense* variables as follows:

- For binary search measurements (set_bsearch function):

The *search* variable is set to the average value of *min* and *max* specified in the set_bsearch function.
The *status* variable is always set to 0, which in the offline mode means that the search value was found.
The *sense* variable is set to the *target* specified in the set_bsearch function.

- For linear search measurements (set_lsearch function):

  The *search* variable is set to the average value of *start* and *stop* specified in the set_lsearch function.
  The *status* variable is always set to 0, which in the offline mode means that the search value was found.
  The *sense* variable is set to the *target* specified in the set_lsearch function.

Example

```
int drain, gate, source, stat;
double search;

if (set_bsearch(gate, drain, VS_NORM_ACC, 0.0, 5.0, -1e-5, -1e-5, -2e-6, 0,
0.1) == -1) error_rep();
if (search_iv(&search, &stat, NULL) == -1) error_rep();
```

See Also

- set_bsearch
- set_lsearch
- set_sync (for synchronous search measurements)
- search_iv2
- rsearch_iv

search_iv2

Revision	A.01.11 or later
Control	SMU

This function performs search measurements according to the conditions set by the set_bsearch, set_bdv_search, or set_lsearch function. After search is completed, the final output value of search unit, and the measurement value of sense unit are returned.

If you include a set_sync function between set_bsearch (or set_lsearch) and the search_iv2 function, the search_iv2 function performs a search measurement using the synchronous search unit specified by set_sync.

Breakdown voltage search measurement using set_bdv_search does not support synchronous port set by set_sync. So, if you include a set_sync function between set_bdv_search and the search_iv2 functions, the search_iv2 resets the settings set by set_sync function.

The force_i or force_v function cannot be used between the search_iv2 and set_bsearch (or set_lsearch) functions.

For real-time search, refer to the rsearch_iv function.

NOTE

Differences between search_iv and search_iv2

search_iv2 returns actual measured value to *sense* even if the following measurement status is detected during a search measurement:

- Another unit than sense unit has reached compliance
- Sense unit has reached compliance
- Measurement overflow
- Sense unit is oscillating

search_iv returns –9999999.99999 to *sense* if the above status is detected.

Synopsis

```
int search_iv2 (double *search, int *status, double *sense, double
                *time_stamp);
```

Argument	Range Restrictions/Description
search	Specify a pointer to a double variable in which to return the search value.
status	Specify a pointer to an integer variable in which to return the search measurement status. One of the following integers is returned to *status*. You can use the associated macros in your program to determine which status was returned. NORMAL_MEAS (=0) SEARCH_DOMAIN (= 1) SEARCH_ABNORMAL (= 2) SEARCH_FAILED (= 4) SEARCH_COMP_OTHER(=16) SEARCH_COMP (=32) SEARCH_OVERFLOW (=64) SEARCH_OSC (=128)
sense	Specify a pointer to a double variable in which to return the target value measured by the sense unit.
time_stamp	Specify a pointer to a double variable in which to return the time stamp for measured value.

- *search*

  This function performs a search measurement, and returns the search value to the *search* variable. The search value is the value output by search unit when the target value is found by the sense unit.

  If search value is not found, one of the following is returned to *search*:

  –9999999.99999 Search value was not found.

9999999.99999	Search measurement was aborted.
–8888888.88888	(For binary search measurements only – set_bsearch function) Search value not found. Change the *max* value to increase the difference between *min* and *max*.
–7777777.77777	(For binary search measurements only – set_bsearch function) Search value not found. Change the *min* value to increase difference between *min* and *max*.

- *sense*

  The target value measured by the sense unit is returned to the *sense* variable.

  In case of set_bdv_search, *breakdown_current* specified by set_bdv_search is returned to the *sense* variable.

- *status*

  The search_iv2 function returns an integer (0, 1, 2, 4, 16, 32, 64, or 128) to *status*. You can use the following macros in your program to determine which status was returned:

  - For binary search measurements (set_bsearch function):

NORMAL_MEAS (=0):	Search value successfully found.
SEARCH_DOMAIN (=1):	Search value not found (between the *min* and *max* defined in set_bsearch). If –7777777.77777 is returned to the *search* variable, change the *min* value to increase difference between *min* and *max*. If –8888888.88888 is returned to the *search* variable, change the *max* to increase the difference between *min* and *max*.
SEARCH_ABNORMAL (=2):	Abnormal data.
SEARCH_FAILED (=4):	Sense unit cannot find specified *target* within specified accuracy.

SEARCH_COMP_OTHER (=16):	Another unit than sense unit has reached compliance.
SEARCH_COMP (=32):	Sense unit has reached compliance.
SEARCH_OVERFLOW (=64):	Measurement Overflow.
SEARCH_OSC (=128):	Sense unit is oscillating.

- For linear or breakdown voltage search measurements (set_lsearch or set_bdv_search function):

NORMAL_MEAS (=0):	Search value successfully found.
SEARCH_DOMAIN (=1):	Search value not found (between the *start* and *stop* values specified in set_lsearch or set_bdv_search).
SEARCH_ABNORMAL (=2):	Abnormal data
SEARCH_FAILED (=4):	Abnormal compliance loop
SEARCH_COMP_OTHER (=16):	Another unit than sense unit has reached compliance.
SEARCH_COMP (=32):	Sense unit has reached compliance.
SEARCH_OVERFLOW (=64):	Measurement Overflow.
SEARCH_OSC (=128):	Sense unit is oscillating.

- *time_stamp*

  This function returns accumulated time at measurement start in seconds (from last execution of init_system function or `/opt/hp4070/bin/hp4070 -start` command) to the *time_stamp* variable.

  If reset_timestamp function has been executed after init_system function or `/opt/hp4070/bin/hp4070 -start` command, the accumulated time from the first execution of force_i, force_v, or force_meas function after reset_timestamp is returned.

Offline Mode This function returns values to the *search*, *status*, and *sense* variables as follows:

- For binary search measurements (set_bsearch function):

  The *search* variable is set to the average value of *min* and *max* specified in the set_bsearch function. The *status* variable is always set to 0, which in the offline mode means that the search value was found. The *sense* variable is set to the *target* specified in the set_bsearch function.

- For linear or breakdown voltage search measurements (set_lsearch or set_bdv_search function):

  The *search* variable is set to the average value of *start* and *stop* specified in the set_lsearch function. The *status* variable is always set to 0, which in the offline mode means that the search value was found. The *sense* variable is set to the *target* specified in the set_lsearch function.

set_adc

Revision	A.01.00 or later
Control	SMU

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

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

Synopsis

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

Argument	Range Restrictions/Description	
adc	Specify the high-speed ADC or high-resolution ADC. You can use the following macros:	
	PERCH_ADC(=0): REF_ADC(=1):	High-speed ADC High-resolution ADC
mode	Specify the integration mode. You can use the following macros:	
	INTEG_MANUAL(=0): INTEG_SHORT(=1): INTEG_MEDIUM(=2): INTEG_LONG(=3):	Manual Short Medium Long

Argument	Range Restrictions/Description	
value	Specify the integration time value or number of samples:	
	For high-speed ADC:	
	if *mode* is INTEG_MANUAL:	0 or 1 to 4096 [times]
	if *mode* is INTEG_SHORT:	0 or 1 to 4096 [times]
	if *mode* is INTEG_MEDIUM:	ignored.
	if *mode* is INTEG_LONG:	0 or 1 to 100 [PLC]
	For high-resolution ADC:	
	if *mode* is INTEG_MANUAL:	0, 80E–6 to 20E–3 [s], or 1 to 100 [PLC]
	if *mode* is INTEG_SHORT:	0 or 80E–6 to 20E–3 [s] (Note 1)
	if *mode* is INTEG_MEDIUM:	ignored
	if *mode* is INTEG_LONG:	0 or 1 to 100 [PLC]
autozero	Auto-zero function of SMU (only for high-resolution ADC). If *adc* is PERCH_ADC, this is dummy parameter. You can use the following macros:	
	AUTOZERO_OFF(=0):	disable Auto-zero function of SMU.
	AUTOZERO_ON(=1):	enable Auto-zero function of SMU.

Note 1 Actual integration time is calculated as system pre-defined factor × *value* seconds.

- *adc*

  adc selects the A/D converter for which to set the parameters.

- *mode* and *value*

  mode sets the integration mode of the A/D converter. *value* sets the actual integration time or number of samples to average for integration as follows:

- High-speed A/D Converter

Integration mode	Integration time
INTEG_MANUAL	*value* is the number of samples to average for integration. If you specify 0 for *value*, default (1) is used.
INTEG_SHORT	The number of samples to average for integration is calculated: system pre-defined number of samples × *value*. If you specify 0 for *value*, 1 is used for value.
INTEG_MEDIUM	Integration time is set to 1 Power Line Cycle (PLC). *value* is ignored.
INTEG_LONG	Integration time is set to *value* PLCs. If you specify 0 for *value*, default (16 PLCs) is used.

- High-resolution A/D Converter

Integration mode	Integration time
INTEG_MANUAL	If *value* is in range from 80E–6 and 20E–3, integration time is set to *value* seconds. If *value* is in range from 1 to 100, integration time is set to *value* PLC. If you specify 0 for *value*, default integration time (240E–6 s) is used.
INTEG_SHORT	Integration time is calculated: system pre-defined factor *value* seconds. If you specify 0 for *value*, 480E–6 (s) is used for *value*.
INTEG_MEDIUM	Integration time is set to 1 Power Line Cycle (PLC). *value* is ignored.
INTEG_LONG	Integration time is set to *value* PLCs. If you specify 0 for *value*, default integration time (16 PLCs) is used.

- *autozero*

  autozero sets the Auto-zero function of SMU to ON or OFF. *autozero* is effective for high-resolution ADC only.

set_adc_i

Revision	B.03.00 or later
Control	SMU

This function sets the operating parameters of the analog-to-digital converter (high-speed ADC or high-resolution ADC) used by the SMUs for current measurements.

To select the converter used by an SMU, use the set_smu_ch function.

Synopsis

```
int set_adc_i (int adc, int mode, double value, int autozero, double
               b_range, int mode_h, double value_h);
```

Argument	Range Restrictions/Description	
adc	Specify the high-speed ADC or high-resolution ADC. The following macros are available:	
	PERCH_ADC(=0): REF_ADC(=1):	High-speed ADC High-resolution ADC
mode	Specify the integration mode for the current ranges lower than *b_range*. The following macros are available:	
	INTEG_MANUAL(=0): INTEG_SHORT(=1): INTEG_MEDIUM(=2): INTEG_LONG(=3): INTEG_SMART(=4):	Manual Short Medium Long Smart

Argument	Range Restrictions/Description	
value	Specify the integration time value or number of samples for the current ranges lower than *b_range*:	
	For high-speed ADC:	
	if *mode* is INTEG_MANUAL:	0 or 1 to 4096 [times]
	if *mode* is INTEG_SHORT:	0 or 1 to 4096 [times]
	if *mode* is INTEG_MEDIUM:	ignored.
	if *mode* is INTEG_LONG:	0 or 1 to 100 [PLC]
	if *mode* is INTEG_SMART:	0 or 1 to 100 [PLC]
	For high-resolution ADC:	
	if *mode* is INTEG_MANUAL:	0, 80E–6 to 20E–3 [s], or 1 to 100 [PLC]
	if *mode* is INTEG_SHORT:	0 or 80E–6 to 20E–3 [s] (Note 1)
	if *mode* is INTEG_MEDIUM:	ignored
	if *mode* is INTEG_LONG:	0 or 1 to 100 [PLC]
	if *mode* is INTEG_SMART:	0 or 1 to 100 [PLC]
autozero	Auto-zero function of the SMU (only for high-resolution ADC). If *adc* is PERCH_ADC, this is a dummy parameter. The following macros are available:	
	AUTOZERO_OFF(=0):	disable Auto-zero function of the SMU.
	AUTOZERO_ON(=1):	enable Auto-zero function of the SMU.
b_range	Specify the range for switching the operation mode of the A/D converter between the higher current range and the lower current range. –1.0 to 1.0	
mode_h	Specify the integration mode for the current ranges greater than, or equal to, *b_range*. Refer to *mode* for available macros.	
value_h	Specify the integration time value or number of samples for the current ranges greater than, or equal to, *b_range*. Refer to *value* for range restriction.	

Note 1 The actual integration time is calculated as the system pre-defined factor × *value* in seconds.

- *adc*

  adc selects the A/D converter to set the parameters for.

- *mode* and *value*

  The *mode* parameter sets the integration mode of the A/D converter. The *value* parameter sets the actual integration time or the number of samples to average for the integration as follows:

  - High-speed A/D Converter

Integration mode	Integration time
INTEG_MANUAL	*value* is the number of samples to average for integration. If you specify 0 for *value*, default (1) is used.
INTEG_SHORT	The number of samples to average for integration is calculated as: system pre-defined number of samples × *value*. If you specify 0 for *value*, 1 is used.
INTEG_MEDIUM	Integration time is set to 1 Power Line Cycle (PLC) and *value* is ignored.
INTEG_LONG	Integration time is set to *value* in PLCs. If you specify 0 for *value*, the default (16 PLCs) is used.
INTEG_SMART	Integration time is dynamically optimized by *value* PLCs and the ratio of the pre-measured current value to the range value. If you specify 0 for *value*, the default (16 PLCs) is used.

- High-resolution A/D Converter

Integration mode	Integration time
INTEG_MANUAL	If *value* is within the range from 80E–6 s to 20E–3 s, the integration time is set to *value* in seconds. If *value* is within the range from 1 to 100, the integration time is set to *value* in PLC. If you specify 0 for *value*, the default integration time (240E–6 s) is used.
INTEG_SHORT	Integration time is calculated as: the system pre-defined factor *value* in seconds If you specify 0 for *value*, 480E–6 (s) is used.
INTEG_MEDIUM	Integration time is set to 1 Power Line Cycle (PLC) and *value* is ignored.
INTEG_LONG	Integration time is set to *value* in PLCs. If you specify 0 for *value*, the default integration time (16 PLCs) is used.
INTEG_SMART	Integration time is dynamically optimized by *value* in PLCs and the ratio of the pre-measured current value to the range value. If you specify 0 for *value*, the default (16 PLCs) is used.

- *autozero*

  The *autozero* function sets the Auto-zero function of the SMU to ON or OFF. The *autozero* function is effective for the high-resolution ADC only.

- *b_range*, *mode_h*, and *value_h*

  You can set different operation modes for the A/D converter between the ranges greater than, or equal to, *b_range* and for the ranges less than *b_range*.

  The *mode_h* parameter sets the operation mode and *value_h* sets the actual integration time or number of samples to average for integration for the ranges greater than, or equal to, *b_range*.

  The meaning of each value of *mode_h* and *value_h* are the same as *mode* and *value* respectively.

NOTE

If you specify *b_range* as the value out of your SMU range, *mode_h* and *value_h* are ignored and *mode* and *value* are used for all ranges.

set_adc_v

Revision	B.03.00 or later
Control	SMU

This function sets the operating parameters of the analog-to-digital converter (high-speed ADC or high-resolution ADC) used for voltage measurements by the SMUs.

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

Synopsis

```
int set_adc_v (int adc, int mode, double value, int autozero);
```

Argument	Range Restrictions/Description	
adc	Specify the high-speed ADC or high-resolution ADC. The following macros are available:	
	PERCH_ADC(=0): REF_ADC(=1):	High-speed ADC High-resolution ADC
mode	Specify the integration mode. The following macros are available:	
	INTEG_MANUAL(=0): INTEG_SHORT(=1): INTEG_MEDIUM(=2): INTEG_LONG(=3):	Manual Short Medium Long

Argument	Range Restrictions/Description	
value	Specify the integration time value or the number of samples:	
	For high-speed ADC:	
	if *mode* is INTEG_MANUAL:	0 or 1 to 4096 [times]
	if *mode* is INTEG_SHORT:	0 or 1 to 4096 [times]
	if *mode* is INTEG_MEDIUM:	ignored.
	if *mode* is INTEG_LONG:	0 or 1 to 100 [PLC]
	For high-resolution ADC:	
	if *mode* is INTEG_MANUAL:	0, 80E–6 to 20E–3 [s], or 1 to 100 [PLC]
	if *mode* is INTEG_SHORT:	0 or 80E–6 to 20E–3 [s] (Note 1)
	if *mode* is INTEG_MEDIUM:	ignored
	if *mode* is INTEG_LONG:	0 or 1 to 100 [PLC]
autozero	Auto-zero function of SMU (only for high-resolution ADC). If *adc* is PERCH_ADC, this is the dummy parameter. The following macros are available:	
	AUTOZERO_OFF(=0):	disable Auto-zero function of SMU.
	AUTOZERO_ON(=1):	enable Auto-zero function of SMU.

Note 1 The actual integration time is calculated as:
the system pre-defined factor × *value* in seconds.

- *adc*

  The *adc* parameter selects the A/D converter to set the parameters for.

- *mode* and *value*

  The *mode* parameter sets the integration mode of the A/D converter. The *value* parameter sets the actual integration time or the number of samples to average for integration as follows:

- High-speed A/D Converter

Integration mode	Integration time
INTEG_MANUAL	*value* is the number of samples to average for integration. If you specify 0 for *value*, the default (1) is used.
INTEG_SHORT	The number of samples to average for integration is calculated as: the system pre-defined number of samples × *value*. If you specify 0 for *value*, 1 is used.
INTEG_MEDIUM	Integration time is set to 1 Power Line Cycle (PLC) and *value* is ignored.
INTEG_LONG	Integration time is set to *value* in PLCs. If you specify 0 for *value*, the default (16 PLCs) is used.

- High-resolution A/D Converter

Integration mode	Integration time
INTEG_MANUAL	If *value* is within the range from 80E–6 s to 20E–3 s, the integration time is set to *value* in seconds. If *value* is within the range from 1 to 100, the integration time is set to *value* of the PLC. If you specify 0 for *value*, the default integration time (240E–6 s) is used.
INTEG_SHORT	Integration time is calculated as: system pre-defined factor *value* in seconds. If you specify 0 for *value*, 480E–6 (s) is used for *value*.
INTEG_MEDIUM	Integration time is set to 1 Power Line Cycle (PLC) and *value* is ignored.
INTEG_LONG	Integration time is set to *value* in PLCs. If you specify 0 for *value*, the default integration time (16 PLCs) is used.

- *autozero*

  The *autozero* parameter sets the Auto-zero function of the SMU to ON or OFF. The *autozero* parameter is effective for high-resolution ADC only.

set_adc3458

Revision	A.01.00 or later
Control	DVM

This function sets the measurement integration time for DVM.

Synopsis

```
int set_adc3458 (double value, int autozero);
```

Argument		Range Restrictions/Description
value	Integration time	0, 0.5E–6 to 999999.9E–6 [s] resolution: 0.1E–6 for 3458A (or 0.2E–6 for 34470)
		1 to 10 [PCL] resolution: 1 [PLC]
		10 to 100 [PLC] resolution: 10 [PLC]
autozero	Auto-zero function of DVM. You can use the following macros: OFF (=0) or ON (=1)	

- *value*

  You can specify seconds (0.5E–6 to 999999.9E–6) or the number of power line cycles (1 to 100 PLC).

  If 0 is specified as *value*, integration time is set to 0.5 μs for 3458A or 0.2 ms for 34470, that is the minimum integration time.

  For 34470, if a value from 0.5E-6 to 0.2E-4 is specified, it is set to 0.2 ms.

- *autozero*

  If *autozero* is set to ON (enable), zero offset cancel is performed for each measurement.

  If *autozero* is set to OFF (disable), zero offset cancel is performed only for next measurement after this function.

set_bdv

Revision	A.01.00 or later
Control	SMU

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

The specified SMU forces the start voltage, waits the hold time, then starts to force the stop voltage. The voltage increases from start voltage at a constant rate. When breakdown starts to occur, the current increases rapidly, and quickly reaches the *current* (current compliance), and voltage increases at a slower rate.

When the rate of the voltage increase becomes less than half of the initial rate, this means the SMU has reached the breakdown. The SMU waits the delay time, then measures voltage and returns the value to the *voltage* variable specified by measure_bdv function.

After the measurement, the forced voltage is reset to *start*. The slew rate of the quasi-pulse is determined by capacitance on the DUT, cables, test fixtures, and so on.

The figure in the description shows the quasi-pulsed waveform for a breakdown voltage measurement. This measurement is useful when the DUT is designed with a very wide breakdown voltage range. When breakdown starts to occur, the current increases, but is limited to your desired *current* setting to prevent damage to the DUT.

Synopsis

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

Argument	Range Restrictions/Description	
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8(20008).
	pin number	1 to 49
range	SMU output voltage range. –100 to 100 (MPSMU and HRSMU) or –200 to 200 (HPSMU) [V]	
start	Specify the start voltage of the search. –100 to 100 (MPSMU and HRSMU) or –200 to 200 (HPSMU) [V].	
stop	Specify the stop voltage of the search. –100 to 100 (MPSMU and HRSMU) or –200 to 200 (HPSMU) [V].	
current	Specify the breakdown current. –0.1 to 0.1 (MPSMU and HRSMU) or –1 to 1 (HPSMU) [A].	
hold	Specify the hold time for starting to force the stop voltage. 0 to 655.350 [s]. resolution: 0.001	
delay	Specify the delay time for measuring voltage. 0 to 65.5350 [s]. resolution: 0.0001	

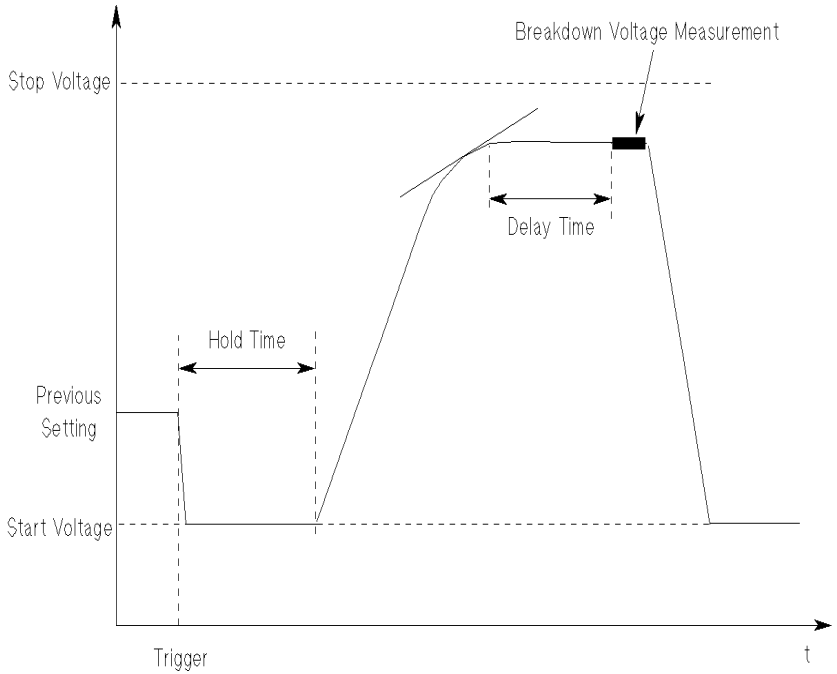
Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port*

  To specify the SMU, use either the *port address* of SMU, or the *pin number* of any measurement pin connected to the SMU port.

- *start* voltage, *stop* voltage, *hold* time, and *delay* time

  The following figure shows the relationship between these parameters.



After start trigger (measure_bdv function), the SMU forces the *start* voltage, waits the *hold* time, then starts to force the *stop* voltage.

After the voltage rate becomes half of initial rate, this means breakdown has occurred. The SMU waits *delay* time, then measures the voltage.

If breakdown does not occur, the SMU output stops when *stop* voltage is reached.

The difference between the *start* voltage and *stop* voltage must be at least 10 V.

- *current*

  The *current* specifies the current compliance of the SMU. When breakdown occurs, the current starts to increase rapidly and this limits the current to prevent damage to the DUT.

See Also

- measure_bdv

set_bdv_search

Revision	A.01.11 or later
Control	SMU

set_bdv_search function sets the parameters for breakdown voltage measurement using linear search technique.

To start this search measurement, use the search_iv or search_iv2 function.

For breakdown voltage search measurement:

1. Set the current compliance of the specified SMU to the *breakdown_current* value.
2. The specified SMU sweeps from the *start* voltage to the *stop* voltage while the specified SMU determines if the SMU has reached the compliance, which means the *breakdown_current* has been reached.
3. The output voltage setting value at current compliance is returned as a breakdown voltage.

NOTE

Difference between set_bdv_search and set_lsearch with compliance stop

The set_bdv_search is specially optimized to shorten the wait time before compliance check at each voltage step so that the stresses to the DUT can be reduced.

Synopsis

```
int set_bdv_search (int port, double start, double stop, double step,
                    double breakdown_current, double comp_chk_wait_adjust, double
                    comp_chk_delay, double delay, int skip, double fine_start, int
                    skip_back, double skip_back_delay);
```

Argument	Range Restrictions/Description	
port	SMU search port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
start, *stop*	Specify the search start and stop voltages. –100 to 100 (MPSMU and HRSMU) or –200 to 200 (HPSMU) [V]	
step	Specify the step size for fine search. 0 < *step* ≤ 100 (MPSMU and HRSMU) or 0 < *step* ≤ 200 (HPSMU) [V]	
breakdown_current	Specify the compliance of the specified SMU and is used to determine if the breakdown occurs. –0.1 to 0.1 (MPSMU and HRSMU) or –1 to 1 (HPSMU) [A]	
comp_chk_wait_adjust, *comp_chk_delay*	Optimizes the compliance check wait time. *comp_chk_wait*: 0.0 to 10.0 with 0.1 resolution. *comp_chk_delay* [s]: 0.0000 to 65.5350 with 0.0001 resolution.	
delay	Increase measurement delay time. 0.0000 to 65.5350 [s] with 0.0001 resolution.	
skip	Determines size of large step. 1 to 20000	
fine_start	Specify current at which to start fine search. –0.1 to 0.1 (MPSMU and HRSMU) or –1 to 1 (HPSMU) [A]	

Argument	Range Restrictions/Description
skip_back	Number of large steps to go back after finding *fine_start*. 0 to 100
skip_back_delay	Delay time between skip back and start of fine search. 0.0000 to 65.5350 [s] with 0.0001 resolution.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *start*, *stop*, and *step*

  The *start* can be less or greater than the *stop*, but not equal. The *step* must be positive.

  If the *start* is less than the *stop*, the *step* voltage is added to the *start* voltage at each iteration. If the *start* is greater than the *stop*, the *step* voltage is subtracted at each iteration.

  If the *step* is greater than the differences between *start* and *stop*, the search will be only performed at *start* voltage and *stop* voltage.

  The output range of the specified SMU is set to the lowest range that includes the *start* voltage, *stop* voltage, and *step* voltage. Refer to force_v or force_i for output value, range, and compliance relationships.

  Also, after the breakdown voltage is searched or SMU output is reached to the *stop* voltage, the SMU is set to the *start* voltage.

  Refer to Figure 2-3 on page 369.

- *comp_chck_wait_adjust* and *comp_chck_delay*

  For a reliable search measurement, the adequate wait time is necessary for the SMU to step up next voltage and stabilize its output before compliance check.

  Large current for charging stray capacitance on the measurement path will flow from the SMU immediately after forcing new step voltage, so an insufficient wait time for compliance check may make the SMU detect wrongly this transient current as the *breakdown_current*.

  On the other hand, longer compliance check wait time means that DUT is stressed for longer time and may be damaged.

You can optimize the compliance check wait time by using *comp_chck_wait_adjust* and *comp_chck_delay*.

The compliance check wait time is calculated as follows:

$$\text{comp check wait time} = Ws \times Wc + Dc$$

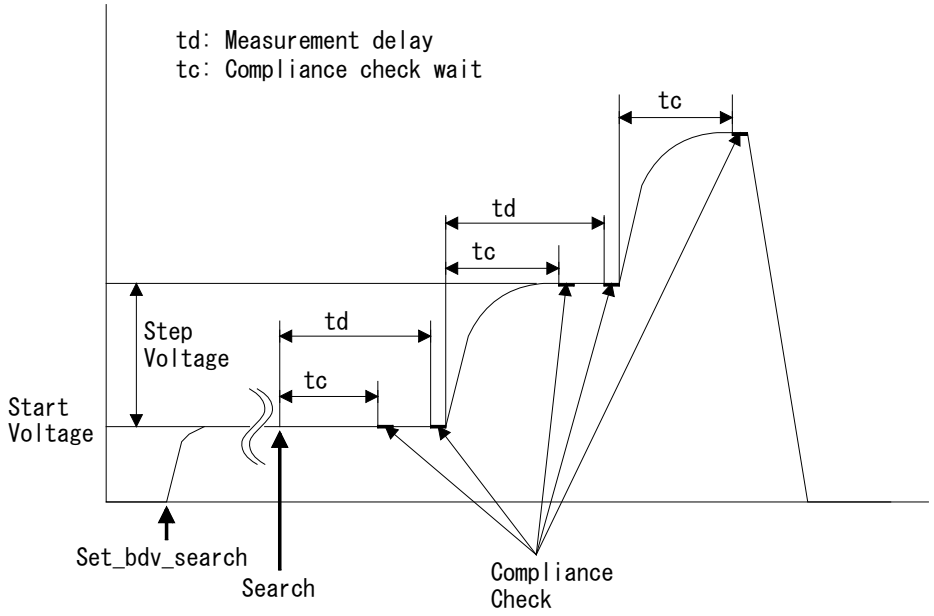
Where:

Ws: System pre-defined wait time

Wc: *comp_chck_wait_adjust*

Dc: *comp_chck_delay*

Figure 2-3 Breakdown Voltage Search Output



- *delay*

  This parameter is valid only if you specify *skip* to greater than or equal to 2 (that means step skipping method is enabled).

  If step skipping method is enabled, the current measurement will be performed at each voltage step.

The measurement delay time is set to system pre-defined value + *delay* or *delay* + measurement wait time adjusted by set_wait_time function.

For reliable measurement, you can specify additional delay by using this parameter.

- *skip*, *fine_search_start*, *skip_back*, and *skip_back_delay* If you specify *skip* to greater than or equal to 2, step skipping method is enabled and the current measurement will be performed at each voltage step.

  Breakdown voltage search with step skipping method is performed as follows:

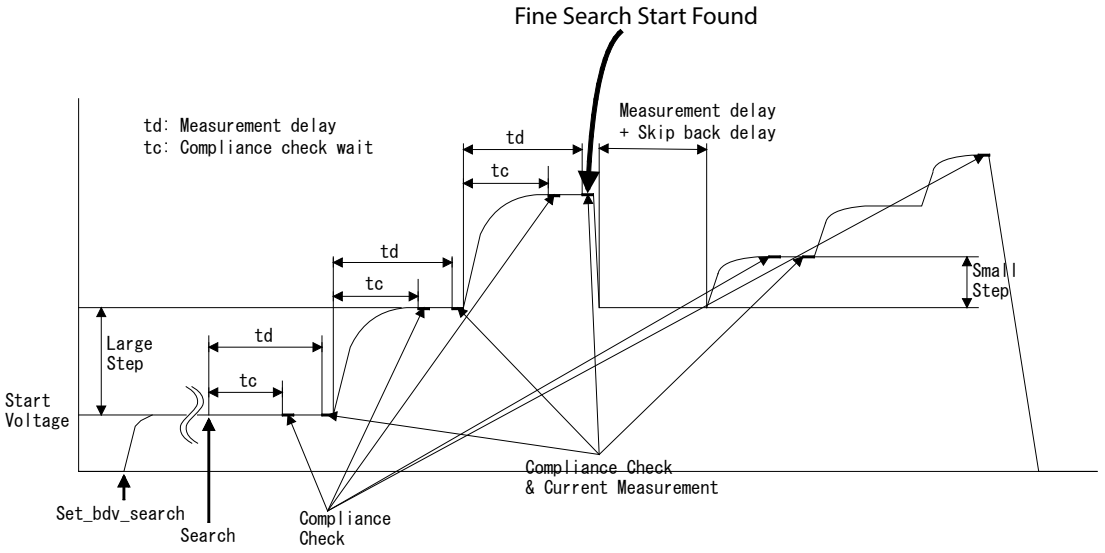
  1. SMU starts to sweep with large steps from *start* voltage until measured value is greater than *fine_search_start*.

     Size for large step is calculated from *skip* as follows:

     skip × *step*

  2. SMU goes back the specified number of large steps (*skip_back*).
  3. After waiting *skip_back_delay*, SMU starts to sweep with small steps (*step*) until SMU reaches the compliance (*breakdown_current*).

Figure 2-4 Breakdown Voltage Search with Step Skipping



See Also · search_iv, search_iv2, set_lsearch

set_bsearch

Revision	A.01.00 or later
Control	SMU

This function sets the conditions for a binary search measurement, and the search_iv or rsearch_iv function starts the search.

For a binary search measurement, the search unit outputs the *min* value, then outputs the *max* value while the sense unit measures and compares to the *target* value. The search unit continues to output up or down in smaller increments until the *conv_condition* is satisfied.

Synopsis

```
int set_bsearch (int search_port, int sense_port, int mode, double min,
                 double max, double target, double range, double search_comp,
                 double delay, double conv_condition);
```

Argument	Range Restrictions/Description	
search_port	SMU search port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
sense port	SMU sense port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49

Argument	Range Restrictions/Description
mode	Specify the source mode (VS or IS) of the *search_port* and the convergence mode (ACCURACY or TIMES) of the binary search. You can specify one of the following macros: VS_NORM_ACC (= 0) IS_NORM_ACC (= 1) VS_NORM_TIMES (= 4) IS_NORM_TIMES (= 5) Also, you can specify cautious search mode by specifying the logical addition of CAUTIOUS_SEARCH (=8) and one of the above macros. For example, VS_NORM_ACC|CAUTIOUS_SEARCH specifies voltage source mode, accuracy convergence mode, and cautious search mode.
min, *max*	Specify the initial lower and upper boundaries of the binary search: Search unit is SMU in VS mode [V]: –100 to 100 or –200 to 200 (HPSMU) Search unit is SMU in IS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU)
target	Specify the target value at *sense_port*: Sense unit is SMU in VS mode [A]: –0.1 to 0.1 or –1 to 1(HPSMU) Sense unit is SMU in IS mode [V]: –100 to 100 or –200 to 200(HPSMU)
range	Specify a measurement range: 0 for Auto range mode or Sense unit is SMU in VS mode [A]: –0.1 to 0.1 or –1 to 1(HPSMU) Sense unit is SMU in IS mode [V]: –100 to 100 or –200 to 200(HPSMU)
search_comp	Specify the voltage or current compliance on *search_port*. REMAIN macro (Note 2) or Search SMU is in VS mode [A]: –0.1 to 0.1 or –1 to 1(HPSMU) Search SMU is in IS mode [V]: –100 to 100 or –200 to 200(HPSMU)

Argument	Range Restrictions/Description
delay	Specify settling wait time between output from search unit and measurement by sense unit. Numeric expression [s]: 0 to 65.5350 Resolution: 100 μs
conv_condition	If *mode* is set to ACCURACY, *conv_condition* specifies error range (%) of convergence. If *mode* is set to TIMES, *conv_condition* specifies the number of iterations. *conv_condition* should be a double even for TIMES mode. ACCURACY mode [%]: 0 to 100 Resolution: 0.01 TIMES mode: 1 to 16

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1.
REMAIN means the previous compliance setting if source mode of search SMU is same as the previous mode. Otherwise:
If source mode was VS mode and is now IS mode, compliance is set to 20 V.
If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

- *search_port* and *sense_port*

  To specify an SMU, use either the *port address* of SMU, or the *pin number* of any measurement pin connected to the SMU port.

- *mode*

  The *mode* parameter determines the following conditions:

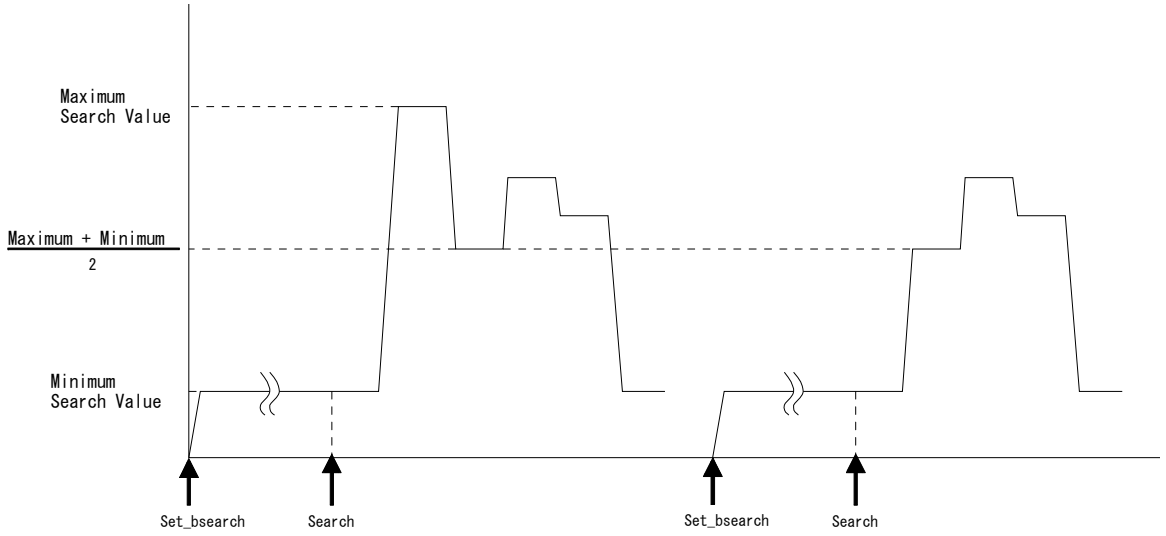
Macro	Search Port Mode	Convergence Mode
VS_NORM_ACC	VS	Accuracy
IS_NORM_ACC	IS	Accuracy

Macro	Search Port Mode	Convergence Mode
VS_NORM_TIMES	VS	Times
IS_NORM_TIMES	IS	Times

Also, you can specify cautious search mode by specifying the logical addition of CAUTIOUS_SEARCH (=8) and one of the above macros.

For example, VS_NORM_ACC|CAUTIOUS_SEARCH specifies voltage source mode, accuracy convergence mode, and cautious search mode.

If you select Cautious search mode, SMU will not output *max*imum search value but output (*max*imum search value + *min*imum search value)/2 at the first binary comparison for reducing DUT damage by *max*imum search value.



If the target can be successfully searched within your specified search range, Cautious search mode is faster and safer than Normal search mode.

If the target should be searched at the range over *maximum search value*, Cautious search mode is very slower to determine search fail.

If separate source and sense SMUs are used for search, you must set sense unit to desired source mode by using force_v or force_i function.

- *max* and *min* (the *max* must not be equal to *min* value)

  These are the maximum and minimum values that will be output by search unit during the search.

  The output range of the search unit is set to lowest range that contains both *max* or *min*. Refer to force_v or force_i for output value, range, and compliance relationships.

- *target*

  The sense port measures, and compares the measurement value to the *target*.

  target must be less than or equal to compliance of sense port.

- *range*

  If the *range* is 0, or less than the *target*, auto-ranging is used. If the sense unit is HRSMU and you want to use auto-ranging mode including 10 pA and 100 pA ranges, you need to execute set_rangemode to set FULL_AUTORANGE_ON before executing this function.

- *conv_condition*

  This parameter depends on the convergence mode (Accuracy or Times) set by the *mode* parameter:

  - If *mode* sets the Accuracy convergence mode:
    The *conv_condition* specifies a %.
    The search continues until the value measured by the sense unit is within the specified % of the target value.
    If search unit has reached its resolution limit, the binary search stops, even if sense unit has not reached convergence condition.
  - If *mode* sets Times convergence mode:
    The *conv_condition* specifies the number of binary comparisons to perform. The maximum number is 16.

- *search_comp*

  If 0 is specified for *search_comp*:

  - If source mode of search unit is same as its previous setting, search unit compliance is not changed from the last valid compliance setting.

- If source mode of search unit is different from its previous setting and the 4080's optimization setting is 2 or 3, *search_comp* must not be 0.
- If the source mode of the search unit is different from its previous setting and the 4080's optimization setting is 0 or 1:

mode change	default value
IS to VS	100 μA
VS to IS	20 V

To set compliance for sense unit, execute force_v or force_i function to set compliance of sense unit before you execute set_bsearch.

- *delay*

  Time between output from search unit and measurement by sense unit.

Example

```
int drain, gate, substrate, stat;
double search;
 :
 :
if (force_v(drain, -2.0, -5, -3e-5) == -1) error_rep();
if (set_bsearch(gate, drain, VS_NORM_ACC, 0.0, 5.0, -1e-5, -1e-5, -2e-6, 0,
0.1) == -1) error_rep();
if set_sync(substrate, POS_SYNC, 0.0, 0.0, 0.0, 0.0) == -1) error_rep();
if (search_iv(&search, &stat, NULL)== -1) error_rep();
if (disable_port3(gate, drain, substrate) == -1) error_rep();
```

See Also

- search_iv
- search_iv2
- rsearch_iv
- set_lsearch
- set_sync (for synchronous search measurements)

NOTE

Reset Mode in 4062 TIS

set_bsearch of 4080 TIS does not support reset mode (*mode*: VS_RESET_ACC,IS_RESET_ACC,VS_RESET_TIMES, or IS_RESET_TIMES).

If your 4062 measurement program execute set_bsearch function in reset mode, that means if your program set the *mode* parameter to VS_RESET_ACC, IS_RESET_ACC, VS_RESET_TIMES, or IS_RESET_TIMES, the 4080 execute it in corresponding normal mode (VS_RESET_ACC to VS_NORM_ACC, IS_RESET_ACC to IS_NORM_ACC, VS_RESET_TIMES to VS_NORM_TIMES, or IS_RESET_TIMES to IS_NORM_TIMES).

set_cmu

Revision	C.04.00 or later
Control	HSCMU

This function sets the measurement integration time and test signal level of the HSCMU.

Synopsis

```
int set_cmu (int integ_time, int multiple, double sig_level);
```

Argument	Range Restrictions/Description
integ_time	You can choose from the following macros for the integration time: INTEG_SHORT (= 1) INTEG_MEDIUM (= 2) INTEG_LONG (= 3)
multiple	1 to 75 (if *integ_time* is INTEG_SHORT) 1 to 100 (if *integ_time* is INTEG_LONG) Ignored (if *integ_time* is INTEG_MEDIUM)
sig_level	REMAIN macro (Note 1) or Numeric expression [V] 0.01, 0.03, 0.05, or 0.1

Note 1 If you specify REMAIN, the most recent signal level is used. The signal level is set to 0.03 V at initialization.

- *integ_time*

  This argument specifies the integration time: INTEG_SHORT, INTEG_MEDIUM, or INTEG_LONG.

  The integration time determines the measurement resolution. The INTEG_LONG integration time gives the best measurement resolution, but takes longest time.

If this function is not executed, INTEG_MEDIUM integration time is used for the HSCMU measurement.

- *multiple*

  This argument specifies the multiple number for integration time of INTEG_SHORT or INTEG_LONG and the integration time is calculated from "System-defined integration time" as follows:

  INTEG_SHORT "System-defined minimum INTEG_SHORT integration time" × *multiple*

  If a large *multiple* value is specified, *multiple* is automatically set to the appropriate value so that the calculated integration time is shorter than "System-defined INTEG_MEDIUM integration time" and is longest.

  INTEG_LONG "System-defined INTEG_MEDIUM integration time" × *multiple*

- *sig_level*

  This argument defines the test signal voltage.

  - If *on_off* parameter of pround_cmu is set to OFF (other than 1), which is the system initial setting, you must specify a correct test signal level from 0.01, 0.03, 0.05, or 0.1 V.
  - If *on_off* parameter of pround_cmu is set to ON (=1), you can specify any value between 0.0 and 0.1 and the specified value is assumed as the proximate test signal level as follows:

Specified *sig_level* [V]	Actual Signal Level
test signal level ≤ 0	error
0 < *sig_level* ≤ 0.01	0.01 V
0.01 < *sig_level* ≤ 0.03	0.03 V
0.03 < *sig_level* ≤ 0.05	0.05 V
0.05 < *sig_level* ≤ 0.1	0.1 V
0.1 < *sig_level*	error

NOTE

You can set any of these levels, but the 4080 guarantees the HSCMU measurement accuracy only for output level 0.03 V, and frequencies 1 kHz, 10 kHz, 100 kHz, and 1 MHz.

Example

```
set_cmu(INTEG_LONG, 20, 0.05);
```

See Also

- measure_cmu
- sweep_cv, sweep_cvf, sweep_zf

set_cmu84

Revision	A.01.00 or later
Control	CMU (HSCMU)

NOTE

CMU Control TIS function

This function can be used to execute 4080 measurement program for using CMU on the 4080 with HSCMU.

Do not use this function to make new measurement program for using HSCMU. (Use set_cmu.)

For HSCMU This function sets the measurement integration time and test signal level of the HSCMU.

Synopsis

```
int set_cmu84 (int integ_time, double sig_level);
```

Argument	Range Restrictions/Description
integ_time	You can choose from the following macros for the integration time: INTEG_SHORT (= 1) INTEG_MEDIUM (= 2) INTEG_LONG (= 3)
sig_level	REMAIN macro (Note 1) or Numeric expression [V] 0.01, 0.03, 0.05, or 0.1

Note 1 If you specify REMAIN, the most recent signal level is used. The signal level is set to 0.03 V at initialization.

- *integ_time*

  This argument specifies the integration time: INTEG_SHORT, INTEG_MEDIUM, or INTEG_LONG.

The integration time determines the measurement resolution. The INTEG_LONG integration time gives the best measurement resolution, but takes longest time.

If this function is not executed, INTEG_MEDIUM integration time is used for the HSCMU measurement.

- *sig_level*

  This argument defines the test signal voltage.

  - If *on_off* parameter of pround_cmu is set to OFF, which is the system initial setting, you must specify a correct test signal level from 0.01, 0.03, 0.05, or 0.1 V.
  - If *on_off* parameter of pround_cmu is set to ON, you can specify any value between 0.0 and 0.1 and the specified value is assumed as the proximate test signal level as follows:

Specified *sig_level* [V]	Actual Signal Level
test signal level ≤ 0	error
0 < *sig_level* ≤ 0.01	0.01 V
0.01 < *sig_level* ≤ 0.03	0.03 V
0.03 < *sig_level* ≤ 0.05	0.05 V
0.05 < *sig_level* ≤ 0.1	0.1 V
0.1 < *sig_level*	error

NOTE

You can set any of these levels, but the 4080 guarantees the HSCMU measurement accuracy only for output level 0.03 V, and frequencies 1 kHz, 10 kHz, 100 kHz, and 1 MHz.

For CMU This function sets the measurement integration time and test signal level of the CMU.

The measurement by CMU is started by the measure_cmu84 or sweep_cv84 function.

Synopsis

```
int set_cmu84 (int integ_time, double sig_level);
```

Argument	Range Restrictions/Description
integ_time	You can choose from the following macros for the integration time: INTEG_SHORT (= 1) INTEG_MEDIUM (= 2) INTEG_LONG (= 3)
sig_level	REMAIN macro (Note 1) or Numeric expression [V] 0 to 20.0 0 to 2.0 (if CMU is 4284A without option 001)

Note 1 If you specify REMAIN, the most recent signal level is used. The signal level is set to 0.03 V at initialization.

- *integ_time*

  This argument specifies the integration time: INTEG_SHORT, INTEG_MEDIUM, or INTEG_LONG.

  The integration time determines the measurement resolution. The INTEG_LONG integration time gives the best measurement resolution, but takes longest time.

  If this function is not executed, INTEG_MEDIUM integration time is used for the CMU measurement. Table 2-52 shows typical measurement times for the E4980A.

Table 2-52 Typical Measurement Times (E4980A)

	1 kHz	10 kHz	100 kHz	1 MHz
INTEG_SHORT	20 ms	8 ms	6 ms	6 ms
INTEG_MEDIUM	110 ms	100 ms	90 ms	90 ms
INTEG_LONG	240 ms	230 ms	220 ms	220 ms

- *sig_level*

  This argument defines the test signal voltage.

NOTE

You can set any of these levels, but the 4080 guarantees the CMU measurement accuracy only for output level 0.03 V, and frequencies 1 kHz, 10 kHz, 100 kHz, and 1 MHz.

The following table shows the *sig_level* you can set (if the CMU is 4284A and option 001 is installed):

Specified *sig_level* [V]	Actual Signal Level	Resolution
sig_level ≤ –0.005	error	n.a.
–0.005 < *sig_level* < 0.005	0.000 V	n.a.
0.005 ≤ *sig_level* ≤ 0.200	*sig_level*	0.001 V
0.200 < *sig_level* ≤ 2.00	*sig_level*	0.01 V
2.00 < *sig_level* ≤ 20.0	*sig_level*	0.1 V
20.0 < *sig_level* < 20.1	20.0 V	n.a.
20.1 ≤ *sig_level*	error	n.a.

The following table shows the *sig_level* you can set (if the CMU is 4284A and option 001 is *not* installed):

Specified *sig_level* [V]	Actual Signal Level	Resolution
sig_level ≤ –0.005	error	n.a
–0.005 < *sig_level* < 0.005	0.000 V	n.a.
0.005 ≤ *sig_level* ≤ 0.200	*sig_level*	0.001 V
0.200 < *sig_level* ≤ 2.00	*sig_level*	0.01 V
2.00 < *sig_level* < 2.01	2.00 V	n.a.
2.01 ≤ *sig_level*	error	n.a.

Example

```
set_cmu84(INTEG_LONG,1.0);
```

See Also

- measure_cmu84
- sweep_cv84
- set_cv84
- set_freq
- frequency

set_cv

Revision	C.04.00 or later
Control	HSCMU

This function sets the sweep parameters for C/G-V measurements by the HSCMU.

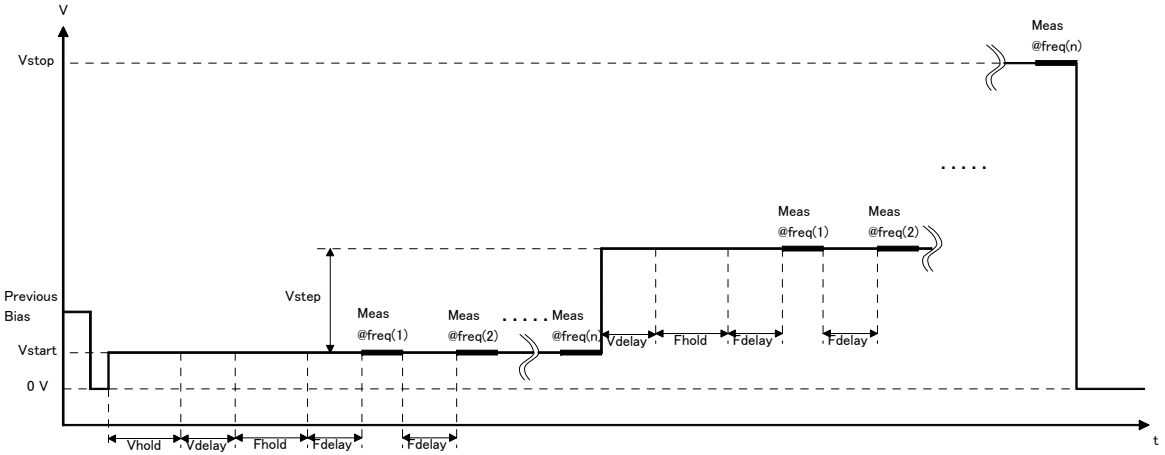
To start the sweep measurements, use the sweep_cv function.

Synopsis

```
int set_cv (int mode, double start, double stop, int number, double hold,
            double delay, int stop_mode);
```

Argument	Range Restrictions/Description
mode	Selects the sweep mode. LINEAR_V (=1) LINEAR_V_DBL (=3)
start, *stop*	Specify the start and stop values of the HSCMU DC bias voltage. Numeric expression [V]: –10.0 to 10.0
number	Specify the number of sweep steps. 2 to 1001
hold	Specify time to wait after the *start* voltage is forced. If a non-zero *delay* is set, an additional *delay* time follows after the *hold* time. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms
delay	Specify the time to wait after each bias voltage change before making measurement. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms
stop_mode	Specify whether to continue or stop if HSCMU reports an error. Choose from one of the following macros: COMP_CONT (= 1) COMP_STOP (= 2)

The C/G-V measurement concept is shown in the following figure.



- *mode*

  The *mode* selects the double or single voltage sweep:

  1: single voltage sweep

  3: double voltage sweep

- *start*, *stop*, and *number*

  The output bias voltage for each step is calculated from these three parameters as follows:

$$\text{ith output bias voltage} = start + \frac{stop - start}{number - 1} \times (i - 1)$$

  `i = 1, 2, ……, number`

  The actual output bias voltage is rounded off to 1 mV resolution.

- *hold*

  The HSCMU waits *hold* after outputting the start voltage.

- *delay*

  For each sweep step, HSCMU waits *delay*, then measures.

- *stop_mode*

  If you set *stop_mode* to COMP_CONT, the bias voltage sweep *continues* to the last specified step, even if an HSCMU reports an error.

  If you set *stop_mode* to COMP_STOP, the bias voltage sweep *aborts* and returns dummy data (9999999.99999) if an HSCMU reports an error.

Example

```
int number;
double v_start, v_stop, hold, delay;
double range, capa[21], cond[21], bias[21];
 :
 :
set_cv(LINEAR_V, v_start, v_stop, number, hold, delay, COMP_CONT);
sweep_cv(range, capa, cond, bias);
```

See Also

- sweep_cv

set_cv84

Revision	A.01.00 or later
Control	CMU, (HSCMU)

NOTE

CMU Control TIS function

This function can be used to execute 4080 measurement program for using CMU on the 4080 with HSCMU.

Do not use this function to make new measurement program for using HSCMU. (Use set_cv.)

This function sets the sweep parameters for C/G-V measurements by the HSCMU or the CMU.

Note that the if the CMU is 4284A, option 001 must be installed to perform the C·G-V measurements on the CMU.

To start the sweep measurements, use the sweep_cv84 function.

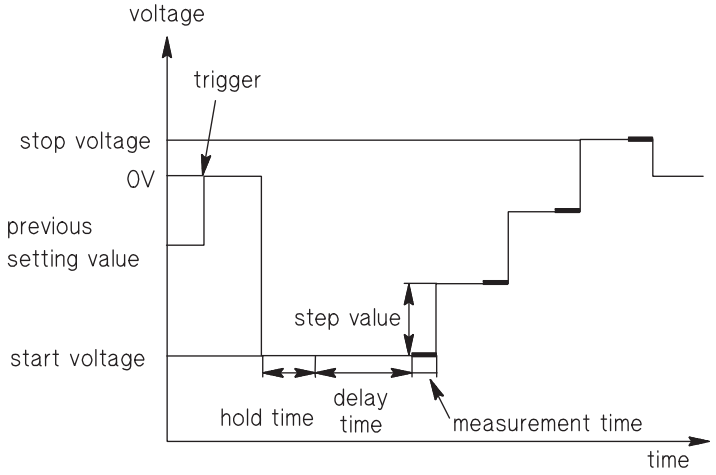
Synopsis

```
int set_cv84 (double start, double stop, int number, double hold, double
              delay);
```

Argument	Range Restrictions/Description
start, *stop*	Specify the start and stop values of the HSCMU or CMU DC bias voltage. Numeric expression [V]: –10.0 to 10.0 (for HSCMU) Numeric expression[V] : –40.0 to 40.0 (for CMU)
number	Specify the number of sweep steps. 2 to 1001

Argument	Range Restrictions/Description
hold	Specify time to wait after the *start* voltage is forced. If a non-zero *delay* is set, an additional *delay* time follows after the *hold* time. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms
delay	Specify the time to wait after each bias voltage change before making measurement. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms

The C/G-V measurement concept is shown in the following figure.



- *start*, *stop*, and *number*

  The output bias voltage for each step is calculated from these three parameters as follows:

$$\text{ith output bias voltage} = start + \frac{stop - start}{number - 1} \times (i - 1)$$

  `i = 1, 2, ……, number`

The actual output bias voltage is rounded off to 1 mV resolution.

- *hold*

  The HSCMU or CMU wait *hold* after outputting the start voltage.

- *delay*

  For each sweep step, HSCMU or CMU wait *delay*, then measures.

Example

```
int number;
double v_start, v_stop, hold, delay;
double range, capa[21], cond[21], bias[21];
 :
 :
set_cv84(v_start, v_stop, number, hold, delay);
sweep_cv84(range, capa, cond, bias);
```

See Also

- sweep_cv84
- measure_cmu84
- set_freq
- frequency

set_cvf

Revision	C.04.00 or later
Control	HSCMU

This function sets the sweep parameters for C/G-V-f measurements by the HSCMU.

To start the sweep measurements, use the sweep_cvf function.

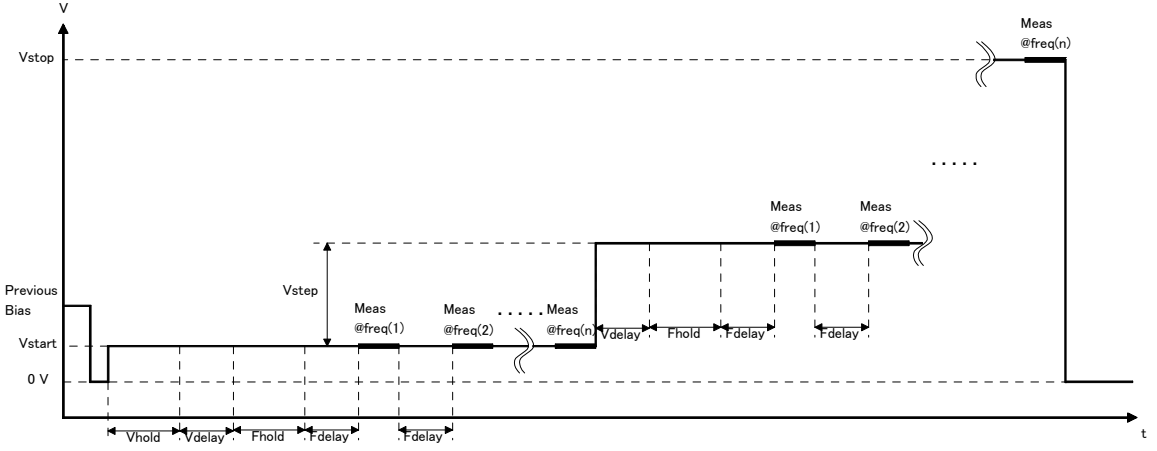
Synopsis

```
int set_cvf (int nf, double *freq, double start, double stop, int number,
             double vhold, double vdelay, double fhold, double fdelay, int
             stop_mode);
```

Argument	Range Restrictions/Description
nf	Specify the number of measurement frequencies 1 to 8
freq	Pointer to double array to specify the measurement frequencies. 1E3 to 2E6 [Hz]
start, *stop*	Specify the start and stop values of the HSCMU DC bias voltage. Numeric expression [V]: –10.0 to 10.0
number	Specify the number of sweep steps. 2 to 1001
vhold	Specify time to wait after the *start* voltage is forced. If a non-zero *vdelay* is set, an additional *vdelay* time follows after the *vhold* time. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms
vdelay	Specify the time to wait after each bias voltage change before making measurement. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms

Argument	Range Restrictions/Description
fhold	Specify time to wait after the first measurement frequency is set. If a non-zero *fdelay* is set, an additional *fdelay* time follows after the *fhold* time. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms
fdelay	Specify the time to wait after each measurement frequency change before making measurement. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms
stop_mode	Specify whether to continue or stop if HSCMU reports an error. Choose from one of the following macros: COMP_CONT (= 1) COMP_STOP (= 2)

The C/G-V-f measurement is similar to the C/G-V measurement. At every bias step, C/G measurements at multiple frequencies are executed. The following figure shows the concept of the C/G-V-f measurement.



- *nf, freq*

  The *nf* specifies the number of measurement frequencies and the *freq* array specify the measurement frequencies. You can specify a maximum of eight measurement frequencies.

  If the *freq* array has more elements than *nf*, only the first specified number of elements are effective.

  If there is a element of which value is 0, the subsequent elements are ineffective.

- *start*, *stop*, and *number*

  The output bias voltage for each step is calculated from these three parameters as follows:

  $$\text{ith output bias voltage} = start + \frac{stop - start}{number - 1} \times (i - 1)$$

  `i = 1, 2, ……, number`

  The actual output bias voltage is rounded off to 1 mV resolution.

- *vhold*

  The HSCMU waits *vhold* after outputting the start voltage.

- *vdelay*

  For each sweep step, HSCMU waits *vdelay*, then measures.

- *fhold*

  The HSCMU waits *fhold* after setting the first measurement frequency.

- *fdelay*

  For each measurement frequency step, HSCMU waits *fdelay*, then execute the C/G measurement.

- *stop_mode*

  If you set *stop_mode* to COMP_CONT, the bias voltage sweep *continues* to the last specified step, even if an HSCMU reports an error.

  If you set *stop_mode* to COMP_STOP, the bias voltage sweep *aborts* and returns dummy data (9999999.99999) if an HSCMU reports an error.

Example

```
int nf, stop_mode, number;
double v_start, v_stop, vhold, vdelay, fhold, fdelay;
double freq[4], range, capa[21], cond[21], bias[21];
double time[84];
int stat[84];
 :
 :
set_cvf(nf, freq, v_start, v_stop, number, vhold, vdelay, fhold, fdelay,
COMP_CONT);
sweep_cvf(range, capa, cond);
status_cvf(freq, bias, stat, time);
```

See Also

- sweep_cvf

set_event_signal

Revision	A.01.00 or later
Control	n.a.

This function sends a specified signal to a specified process when specified event occurs or finishes.

Synopsis

```
int set_event_signal (int event_on, int event_off, int signal, int pid);
```

Argument	Range Restrictions/Description
event_on, *event_off*	You can use the following macros to specify the event: EVENT_HIGH_VOLTAGE (=1) EVENT_PWF (=2) EVENT_OVER_IV (=4) EVENT_FIXOPEN (=8) EVENT_INTLOPEN (=16)
signal	Signal to be sent to a process. You can use macros by including <signal.h> header file.
pid	Specify id number of process.

- *event_on*, *event_off*

  You can specify the following events:

Event	Description
EVENT_HIGH_VOLTAGE (=1)	SMU outputs voltage 40 V or more.
EVENT_PWF (=2)	Testhead power failed.
EVENT_OVER_IV (=4)	Output voltage or current of SMU is over its limit.
EVENT_FIXOPEN (=8)	Fixture is open.
EVENT_INTLOPEN (=16)	Interlock is open.

Example

```
#include "/opt/hp4070/include/tis.h"
#include <signal.h>

set_event_signal(EVENT_HIGH_VOLTAGE, NULL, SIGABRT, pid);
```

set_freq

Revision	A.01.00 or later
Control	HSCMU, CMU

This function sets the measurement frequency of the HSCMU or the CMU.

Synopsis

```
int set_freq (int port, double frequency);
```

Arguments for HSCMU

Argument	Range Restrictions/Description	
port	CMH or CML port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): CMH (20107) or CML (20108)
	pin number	1 to 49
frequency	1 kHz to 2 MHz (Note 2)	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 See the following description.

- *port*

  To specify the CMU, use either the port address of CMH or CML, or the pin number of any measurement pin connected to the CMH or CML port.

- *frequency*

  The measurement frequency can be set to
  1.0, 1.2, 1.5, 2.0, 2.5, 3.0, 4.0, 5.0, 6.0, or 8.0 kHz,
  10, 12, 15, 20, 25, 30, 40, 50, 60, or 80 kHz,
  100, 120, 150, 200, 250, 300, 400, 500, 600, or 800 kHz,
  or 1.0, 1.2, 1.5, or 2.0 MHz.

If on_off parameter of pround_cmu is set to 1, you can specify any frequency between 0 Hz and 2.0 MHz and the approximate valid frequency is set as follows:

frequency	Set Frequency	*frequency*	Set Frequency
frequency ≤ 0.0 Hz	error	40 kHz ≤ *frequency* ≤ 50 kHz	50 kHz
0.0 Hz < *frequency* ≤ 1.0 kHz	1.0 kHz	50 kHz < *frequency* ≤ 60 kHz	60 kHz
1.0 kHz < *frequency* ≤ 1.2 kHz	1.2 kHz	60 kHz < *frequency* ≤ 80 kHz	80 kHz
1.2 kHz < *frequency* ≤ 1.5 kHz	1.5 kHz	80 kHz < *frequency* ≤ 100 kHz	100 kHz
1.5 kHz < *frequency* ≤ 2.0 kHz	2.0 kHz	100 kHz < *frequency* ≤ 120 kHz	120 kHz
2.0 kHz < *frequency* ≤ 2.5 kHz	2.5 kHz	120 kHz < *frequency* ≤ 150 kHz	150 kHz
2.5 kHz < *frequency* ≤ 3.0 kHz	3.0 kHz	150 kHz < *frequency* ≤ 200 kHz	200 kHz
3.0 kHz < *frequency* ≤ 4.0 kHz	4.0 kHz	200 kHz < *frequency* ≤ 250 kHz	250 kHz
4.0 kHz < *frequency* ≤ 5.0 kHz	5.0 kHz	250 kHz < *frequency* ≤ 300 kHz	300 kHz
5.0 kHz < *frequency* ≤ 6.0 kHz	6.0 kHz	300 kHz < *frequency* ≤ 400 kHz	400 kHz
6.0 kHz < *frequency* ≤ 8.0 kHz	8.0 kHz	400 kHz < *frequency* ≤ 500 kHz	500 kHz
8.0 kHz < *frequency* ≤ 10 kHz	10 kHz	500 kHz < *frequency* ≤ 600 kHz	600 kHz
10 kHz < *frequency* ≤ 12 kHz	12 kHz	600 kHz < *frequency* ≤ 800 kHz	800 kHz
12 kHz < *frequency* ≤ 15 kHz	15 kHz	800 kHz < *frequency* ≤ 1.0 MHz	1.0 MHz
15 kHz < *frequency* ≤ 20 kHz	20 kHz	1.0 MHz < *frequency* ≤ 1.2 MHz	1.2 MHz
20 kHz < *frequency* ≤ 25 kHz	25 kHz	1.2 MHz < *frequency* ≤ 1.5 MHz	1.5 MHz
25 kHz < *frequency* ≤ 30 kHz	30 kHz	1.5 MHz < *frequency* ≤ 2.0 MHz	2.0 MHz
30 kHz < *frequency* ≤ 40 kHz	40 kHz	2.0 MHz < *frequency*	error

NOTE

Measurement Accuracy Specifications

Actually, you can set *frequency* to any valid frequency from 1 kHz through 2 MHz without error.

But the 4080 guarantees the measurement accuracy only for measurement frequencies at 1 kHz, 10 kHz, 100 kHz, and 1 MHz, and test signal level at 30 mVrms.

The `frequency(`*freq*`)` function is equivalent to the `set_freq(CMH,`*freq*`)` function.

Arguments for CMU

Argument	Range Restrictions/Description	
port	CMH or CML port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): CMH (20107) or CML (20108)
	pin number	1 to 49
frequency	Measurement frequency of the CMU. You can specify the following measurement frequencies: 1E3, 1E4, 1E5, or 1E6 Hz.	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port*

  To specify the CMU, use either the port address of CMH or CML, or the pin number of any measurement pin connected to the CMH or CML port.

- *frequency*

  You can specify 1 kHz, 10 kHz, 100 kHz, or 1 MHz.

NOTE

Measurement Accuracy Specifications

Actually, you can set *frequency* to any frequency from 20 Hz through 1 MHz without error.

But the 4080 guarantees the measurement accuracy only for frequencies 1 kHz, 10 kHz, 100 kHz, and 1 MHz, and output level 30 mVrms.

The `frequency(`*freq*`)` function is equivalent to the `set_freq(CMH,`*freq*`)` function.

Example

```
set_freq(CMH,1.0e6);
```

See Also

- measure_cpg, measure_cmu84
- set_cmu, set_cmu84
- set_cv, set_cv84, set_cvf, set_zf

set_ileak

Revision	A.01.00 or later
Control	SMU

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

The quasi-pulsed waveform for a leakage current measurement is shown in the following figure.

This type of measurement is useful when a high voltage is forced to a DUT and low leakage current is measured. Because the function automatically determines when output voltage is reached, so user does not have to specify a wait time. The measurement is made soon after output voltage is reached, so device will not be damaged by having high voltage applied to DUT for a long time.

Synopsis

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

Argument	Range Restrictions/Description	
port	Specify an SMU port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
range	Specify the measurement range. Numeric expression [V]: –100 to 100 or –200 to 200(HPSMU)	
voltage	Specify the voltage at which the leakage current is measured. Numeric expression [V]: –100 to 100 or –200 to 200(HPSMU)	

Argument	Range Restrictions/Description
compliance	Specify the current compliance of the *port*. REMAIN macro (Note 2) or Numeric expression [A]: –0.1 to 0.1 or –1 to 1 (HPSMU)
start	Specify the measurement ramp start voltage. REMAIN macro (Note 3) or Numeric expression: [V]: –100 to 100 or –200 to 200(HPSMU)
hold	Specify a hold time before the output voltage starts from the *start* voltage. Numeric expression [s]: 0 to 655.350 Resolution: 0.001
delay	Specify a delay time before the current value is measured at the SMU. Numeric expression [s]: 0 to 65.5350 Resolution: 0.0001

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1.
If the SMU was in VS mode, compliance does not change from the last valid setting.
If the SMU was in IS mode, compliance is set to 100 μA.

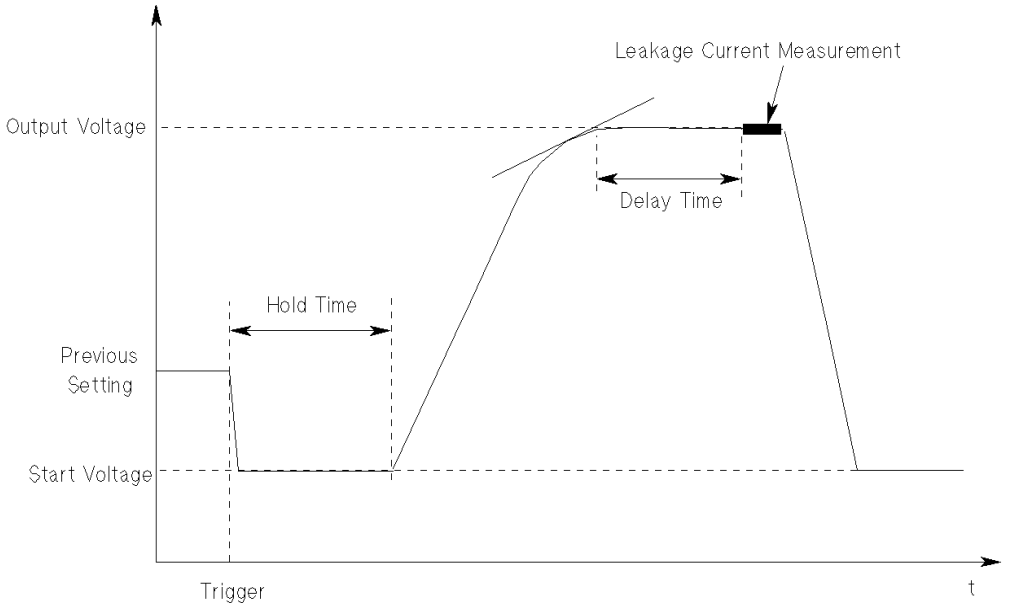
Note 3 Last valid voltage setting of the port is used.

- *port*

  To specify the SMU, use either the port address of SMU, or the pin number of any measurement pin connected to the SMU port.

- output *voltage*, *start* voltage, *hold* time, and *delay* time

  The following figure shows the relationship between these parameters.



After start trigger (measure_ileak function), the SMU forces the *start* voltage, waits *hold* time, then forces the output *voltage*.

The SMU monitors the forced voltage. After the voltage rate becomes half of initial rate (which means *voltage* has been reached), the SMU waits *delay* time, then measures the current.

The slew rate of the quasi-pulse is determined by capacitance on the DUT, cables, test fixtures, and so on.

After the measurement, the forced voltage is reset to the *start* voltage.

The difference between the *start* voltage and output *voltage* must be at least 10 V.

Example

```
set_ileak(SMU1,100.0,60.0,1E-5,10.0,0.5,0.01);
```

See Also

- measure_ileak

set_iv

Revision	A.01.00 or later
Control	SMU

This function sets the sweep parameters for I-V measurements that are started by rsweep_iv, rsweep_miv, sweep_iv, or sweep_miv functions.

Synopsis

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

Argument		Range Restrictions/Description
port	SMU *sweep source* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
sweep_mode	Specify whether sweep is linear or logarithmic, voltage or current, and single or double. Use one of the following macros (or values): LINEAR_V (= 1) LINEAR_I (= 2) LINEAR_V_DBL (= 3) LINEAR_I_DBL (= 4) LOG_V (= –1) LOG_I (= –2) LOG_V_DBL (= –3) LOG_I_DBL (= –4)	

Argument	Range Restrictions/Description
range	Voltage or current output range: 0 for auto-ranging mode or For voltage source: numeric expression [V]: –100 to 100 or –200 to 200(HPSMU) For current source: numeric expression [A]: –0.1 to 0.1 or –1 to 1(HPSMU)
start, *stop*	Start and stop values of the sweep: For voltage source: numeric expression [V]: –100 to 100 or –200 to 200(HPSMU) For current source: numeric expression [A]: –0.1 to 0.1 or –1 to 1(HPSMU) For a log sweep, *start* and *stop* must be the same polarity, and cannot be 0.
number	Number of sweep steps: 2 to 1001 For single sweep, there will be *number* steps from *start* to *stop*. For double sweep, there will be *number* steps from *start* to *stop*, then *number* steps from *stop* to *start*.
hold	Specify time to wait after trigger outputs the *start* value. If a non-zero *delay* is set, an additional *delay* time follows after the *hold* time. Numeric expression [s]: 0 to 655.350 Resolution: 0.001
delay	Specify time to wait after each new output before making measurement: Numeric expression [s]: 0 to 65.5350 Resolution: 0.0001
compliance	Specify a voltage or current compliance value depending on the *sweep_mode*: REMAIN macro (Note 2) or For voltage compliance: numeric expression [V]: –100 to 100 or –200 to 200(HPSMU) For current compliance: numeric expression [A]: –0.1 to 0.1 or –1 to 1(HPSMU)
power_compliance	Limits the power (voltage × current being forced and measured) applied to the *port*: Numeric expression [W]: 0.001 to 2 or 0.001 to 14(HPSMU) Resolution: 0.001 0 means no power compliance. See description.
stop_mode	Specify whether to continue or stop if compliance is reached. Ignored if non-zero *power_compliance* is specified. In this case, *stop_mode* defaults to COMP_STOP mode. To use COMP_CONT, *power_compliance* must be 0. Choose from one of the following macros: COMP_CONT (= 1) COMP_STOP (= 2)

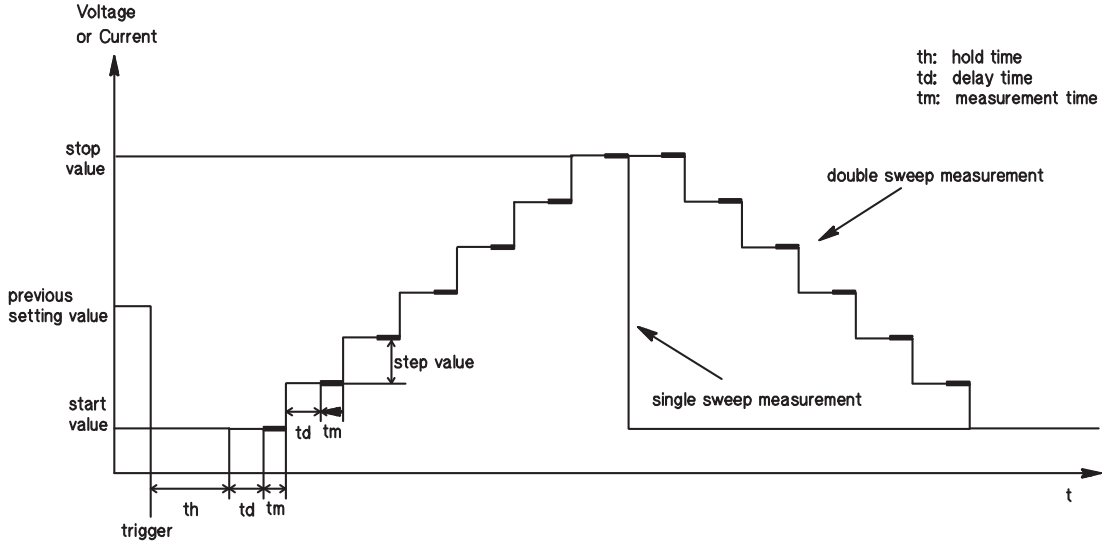
Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1.
REMAIN means the previous compliance setting if source mode of SMU is same as the previous mode. Otherwise:
If source mode was VS mode and is now IS mode, compliance is set to 20 V.
If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

- *start* (voltage or current), *stop*, *number*, *hold*, and *delay*

  For log *sweep mode*, the *start* and *stop* values must be same polarity and cannot be 0.

  When rsweep_iv, rsweep_miv, sweep_miv, or sweep_iv executes, *start* value (voltage or current), *stop* value, *hold* time, and *delay* time determine the sweep measurement conditions as shown in the following figure.



The start, stop, and step values are determined as shown in the following table:

Table 2-53 In the linear sweep mode

Item	Description
start value	resolution: depends on the set voltage or current range.
stop value (Note 1)	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*

Note 1 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. Refer to force_v or force_i for setting resolution.

Table 2-54 In the log sweep mode

Item	Description
start value	resolution: depends on the set voltage or current range.
stop value (Note 1)	$Vstop = Vstart \times 10^{\frac{Rstep \times (Nstep-1)}{20}}$
step ratio	$Rstep = \frac{20 \times \log \frac{Vstop}{Vstart}}{Nstep-1}$ resolution: 0.02 dB/decade maximum: 20 dB/decade

Note 1 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. Refer to force_v or force_i for setting resolution.

- *port*

  To specify the SMU, use either the port address of SMU, or the pin number of any measurement pin connected to the SMU port.

- *sweep_mode*

  The *sweep_mode* determines whether the sweep is linear or logarithmic, double or single, and current or voltage:

LINEAR_V (=1):	linear single voltage sweep
LINEAR_I (=2):	linear single current sweep
LINEAR_V_DBL (=3):	linear double voltage sweep
LINEAR_I_DBL (=4):	linear double current sweep
LOG_V (=–1):	log single voltage sweep
LOG_I (=–2):	log single current sweep
LOG_V_DBL (=–3):	log double voltage sweep
LOG_I_DBL (=–4):	log double current sweep

  For log *sweep mode*, the *start* and *stop* values must be same polarity and cannot be 0.

- *range*

  - If *range* is 0:
    If the *sweep_mode* is 1,2,3 or 4 (linear), the voltage or current range during the sweep is set to lowest range that includes both *start* and *stop* voltage/current. The output range does not change during the sweep.

    If the *sweep_mode* is –1, –2 –3 or –4 (log), the range changes during the sweep to give optimum output range.

  - If *range* is not 0:

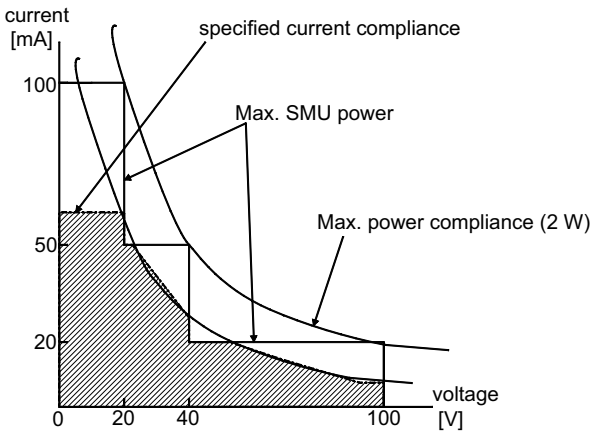
    - Voltage sweep

      Output range is fixed to specified *range*.

    - Current sweep

      If the specified *range* includes both *start* and *stop* current, the specified *range* is used. If not, the SMU ranges up to range that can force both *start* and *stop* current. The output range does not change during the sweep.

- *power_compliance* and *compliance*
  - If 0.0 is specified for *power_compliance*
    The maximum voltage *compliance* or current *compliance* you can specify depends on the maximum output (*start* or *stop* value) during the sweep. Refer to force_v or force_i for details.
  - If non-zero value is specified for *power compliance*
    The maximum voltage *compliance* or current *compliance* can be set up to the maximum value of SMU, independent of maximum output during the sweep. The SMU can be swept at its maximum limits as shown in the following figure.



- *stop_mode*

  If you set *stop_mode* to COMP_CONT, voltage or current sweep *continues* to the last specified step, even if an SMU reaches voltage *compliance* or current *compliance*.

  If you set *stop_mode* to COMP_STOP, the voltage or current sweep *aborts* and returns dummy data (9999999.99999) if an SMU reaches voltage or current *compliance*.

  If you set *power_compliance* to non-zero value, *stop_mode* is ignored. The sweep aborts and returns dummy data when an SMU reaches *power, voltage*, or *current compliance*.

Example

```
set_iv(drain, 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
- rsweep_iv, rsweep_miv
- set_sync (for sweep measurements)
- set_pbias

set_level_pg

Revision	B.02.00 or later
Control	PGU, SPGU

This function sets the voltage and impedance parameters of pulse that will be forced from the specified port connected to the pulse generator (PG).

For the image of output pulses and the pulse setup parameters, see Figure 2-10 on page 460.

If SPGU is used in the ALWG mode, executing this function will cause an error. (Use spgu_alwg_r for specifying load impedance.)

Synopsis

```
int set_level_pg (int port, double amp1, double amp2, double base,
                  double impedance);
```

Argument		Range Restrictions/Description
port	Specify the port connected to the PG. You can specify the HF, THF, AUX, or PSI port address or pin number that currently connects to the port.	
	port address	You can specify the following values (Note 1):
		20201 to 20206, 20211 to 20213, 20251 to 20260, 20101 to 20108
	pin number	1 to 48
amp1, *amp2* (Note 2)	Amplitude from the base voltage to the peak voltage:	
	Numeric expression [V]:	If SPGU: –40.0 to 40.0 @ 1 MΩ load, –20.0 to 20.0 @ 50 Ω load If PGU (81150A): –19.0 to 19.0 @ 1 MΩ load, –10.0 to 10.0 @ 50 Ω load (Note 3)
base	Pulse base voltage:	
	Numeric expression [V]:	If SPGU: –40.0 to 40.0 @ 1 MΩ load, –20.0 to 20.0 @ 50 Ω load If PGU (81150A): –19.0 to 19.0 @ 1 MΩ load, –10.0 to 10.0 @ 50 Ω load

Argument	Range Restrictions/Description	
impedance	Load impedance:	
	Numeric expression [Ω]:	If SPGU: 0.1 to 1E6 If PGU (81150A): 2.5 to 1E6

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use PORT function instead.

Note 2 For 2-level pulse mode, *amp2* is ignored.

Note 3 Minimum value of pulse amplitude is 0.2 V at 1 MΩ load or 0.1 V at 50 Ω load.

NOTE

Restrictions for 3-level pulse output by PGU (81150A)

- *amp1* [V]:

  -19.0 to 19.0 at 1 MΩ load
  -9.9 to 9.9 at 50 Ω load

- *amp2* [V]:

  -19.0 to 10.0 at 1 MΩ load
  -9.9 to 5.0 at 50 Ω load

- |*amp1*| + |*amp2*| [V]:

  0.4 to 20.0 at 1 MΩ load
  0.2 to 10.0 at 50 Ω load

This function sets the pulse base and pulse amplitudes of pulse that will be forced from the specified port by the *port address* or *pin number*, and specifies the load impedance connected to the specified port.

- *amp1*, *amp2*, and *base*

  Pulse amplitude is the amplitude from the *base* voltage (pulse base) to the peak voltage. For the setting resolution of the output level, see Table 2-55.

The round_level_pg function can determine whether the pulse amplitude and pulse base values specified in set_level_pg are rounded or not. The specified pulse amplitude and pulse base values are not rounded as the system initial setting.

For the relationship between pulse voltage setting and *impedance*, see the description of *impedance* in this section.

For setting pulse generator, see "set_type_pg".

- *impedance*

  Load impedance is the impedance value of the device under test (DUT) and is used to adjust the output voltage from PG. The voltage output range depends on the *impedance* and the maximum output current.

  For SPGU, note that the maximum output current is 400 mA. The output impedance of SPGU is 50 Ω (nominal value) and thus the voltage output range at DUT is as follows:

  $\pm V_{set} \times Z_{load} / (Z_{load} + 50)$

  V_{set}: absolute voltage value set in SPGU (maximum 40)
  Z_{load}: actual load impedance

  $V_{set} / (Z_{load} + 50) \leq 0.4$

  If the actual load impedance Z_{load} is not the same as the *impedance* setting, the actual pulse voltage forced to the DUT is different from the pulse voltage setting.

  It is recommended to use the actual load impedance of 50 Ω or Hi-Z (>5 kΩ) for better waveform quality.

Table 2-55 Resolution of Output Level (at 1 MΩ Load)

Output level (Vout)	SPGU	PGU (81150A)
|Vout| ≤ 10 V	0.4 mV	1.0 mV
|Vout| > 10 V	1.6 mV	1.0 mV

set_lsearch

Revision	A.01.00 or later
Control	SMU

This function sets the parameters for a linear search measurement, and the search_iv, search_iv2, or rsearch_iv function starts the search.

For a linear search measurement, the search unit sweeps from the *start* to the *stop* while the *sense_port* determines if the *target* has been reached.

Synopsis

```
int set_lsearch (int search_port, int sense_port, int mode, double start,
                 double stop, double step, double target, double range, double
                 search_comp, double delay);
```

Argument		Range Restrictions/Description
search_port	Specify an SMU search port address. You can specify port address directly or specify a pin number that is connected to the port.	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
sense_port	Specify an SMU sense port address. You can specify port address directly or specify a pin number that is connected to the port.	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49

Argument	Range Restrictions/Description
mode	Choose from one of the following macros for this argument: VS_NORM_ABOVE (= 0) IS_NORM_ABOVE (= 1) VS_NORM_BELOW (= 4) IS_NORM_BELOW (= 5) VS_NORM_STAT (= 8) IS_NORM_STAT (= 9)
start, *stop*	Specify the search start and stop values. Search unit is SMU in VS mode [V]: –100 to 100 or –200 to 200 (HPSMU) Search unit is SMU in IS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU)
step	Specify the step size. Search unit is SMU in VS mode [V]: 0 < *step* ≤ 100 or 0 < *step* ≤ 200 (HPSMU) Search unit is SMU in IS mode [A]: 0 < *step* ≤ 0.1 or 0 < *step* ≤ 1 (HPSMU)
target	Specify the target value of the *sense_port*. Sense unit is SMU in VS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) Sense unit is SMU in IS mode [V]: –100 to 100 or –200 to 200 (HPSMU)
range	Specify a measurement range of the sense port. 0 for Auto range mode or Sense SMU in VS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) Sense SMU in IS mode [V]: –100 to 100 or –200 to 200 (HPSMU)
search_comp	Specify the voltage or current compliance of *search_port*. REMAIN macro (Note 2) or Search SMU is in VS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) Search SMU is in IS mode [V]: –100 to 100 or –200 to 200 (HPSMU)
delay	Specify a settling wait time between *search_port* output and *sense_port* measurement. Numeric expression [s]: 0 to 65.5350 Resolution: 0.0001

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1. REMAIN means the previous compliance setting if source mode of search SMU is same as the previous mode. Otherwise: If source mode was VS mode and is now IS mode, compliance is set to 20 V. If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

- *mode* and *target*

  The *mode* parameter determines the following conditions:

mode	Search Unit VS/IS mode	Stop Mode	Measurement/ Compliance Mode
VS_NORM_ABOVE	VS	sense value > *target*	Measurement mode
IS_NORM_ABOVE	IS	sense value > *target*	Measurement mode
VS_NORM_BELOW	VS	sense value < *target*	Measurement mode
IS_NORM_BELOW	IS	sense value < *target*	Measurement mode
VS_NORM_STAT (Note 1)	VS	sense value > *target value*	Compliance mode
IS_NORM_STAT (Note 1)	IS	sense value > *target value*	Compliance mode

Note 1 For *mode* VS_NORM_STAT or IS_NORM_STAT, you must set sense unit to VS mode, and *target* must be a current value.
To set the sense unit to the desired mode, use the force_i or force_v function before set_lsearch.

- Stop Mode

  The *mode* setting determines stop mode. The stop mode specifies that search will stop when sense value is greater or less than *target* as listed in above table.

- Measurement Mode or Compliance Mode

  If separate source and sense SMUs are used for search, you must set sense unit to desired source mode by using force_v or force_i function.

For measurement mode, the sense unit measures for every step and compares to *target*.

For compliance mode, you must set the sense unit to VS mode, and specified *target* must be a current value. The sense unit compliance is set to the specified *target*. No measurement is performed and search stops when sense unit reaches the compliance value.

- *start*, *stop*, and *step*

  The *start* value can be less or greater than the *stop* value, but not equal. The *step* must be positive.

  If the *start* is less than the *stop*, the *step* is added to the *start* at each iteration. If the *start* is greater than the *stop*, the *step* is subtracted at each iteration.

  The output range of the search unit is set to the lowest range that includes both the *start* and *stop*. Refer to force_v or force_i for output value, range, and compliance relationships.

  Also, after the search is finished, the search unit is set to the *start* value.

- *range*

  This sets the measurement range of the sense unit.

  The *range* should be greater than the *target*. If the *range* is 0 or is less than the *target*, the *range* is set to lowest range that includes the *target*. If the sense unit is HRSMU and you want to use 10 pA or 100 pA range, you must execute set_rangemode to set FULL_AUTORANGE_ON before executing this function.

- *search_comp*

  This sets the compliance of the search unit.

  For sense unit when you use measurement mode, if you want to set the compliance, execute a force_v or force_i function for sense unit before you execute a set_lsearch.

  For sense unit when you use compliance mode, the sense unit compliance is set to the *target*.

- *delay*

  This is the time between the output from search unit and measurement by the sense unit.

NOTE

To Set the Parameters for Step Skipping Method

To set the parameters for step skipping method, use set_skip function.

Example

```
set_lsearch(base, collector, IS_NORM_ABOVE, start, stop,step, target,
sense_range, search_comp, hold);
search_iv(&ib, &stat, &ic);
```

See Also

- search_iv
- rsearch_iv
- set_bsearch
- set_sync (for search measurements)
- set_skip

NOTE

Reset Mode in 4062 TIS

set_lsearch of 4080 TIS does not support reset mode (*mode*: VS_RESET_ABOVE, IS_RESET_ABOVE, VS_RESET_BELOW, IS_RESET_BELOW, VS_RESET_STAT, or IS_RESET_STAT).

If your 4062 measurement program execute set_lsearch function in reset mode, that means if your program set the *mode* parameter to VS_RESET_ABOVE, IS_RESET_ABOVE, VS_RESET_BELOW, IS_RESET_BELOW, VS_RESET_STAT, or IS_RESET_STAT, the 4080 execute it in corresponding normal mode (VS_RESET_ABOVE to VS_NORM_ABOVE, IS_RESET_ABOVE to IS_NORM_ABOVE, VS_RESET_BELOW to VS_NORM_BELOW, IS_RESET_BELOW to IS_NORM_BELOW, VS_RESET_STAT to VS_NORM_STAT, or IS_RESET_STAT to IS_NORM_STAT).

set_pbias

Revision	A.01.00 or later
Control	SMU

This function sets the parameters for a pulsed spot measurement or a staircase sweep with pulsed bias measurement.

Pulsed spot measurement is started by using measure_p function, and staircase sweep with pulsed bias measurement is started by rsweep_iv or sweep_iv.

Synopsis

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

Argument	Range Restrictions/Description	
port	Specify an SMU port address. You can specify port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
mode	Specify either the voltage or current source mode. Choose from the following macros: V_SOURCE (= 1) I_SOURCE (= 2)	
range	Specify the voltage or current force range. 0.0 for Auto range mode or For voltage source: numeric expression [V]: –100 to 100 or –200 to 200 (HPSMU) For current source: numeric expression [A]: –0.1 to 0.1 or –1 to 1 (HPSMU)	

Argument	Range Restrictions/Description
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 be same polarity. For voltage source: numeric expression [V]: –100 to 100 or –200 to 200 (HPSMU) For current source: numeric expression [A]: –0.1 to 0.1 or –1 to 1 (HPSMU)
width (Note 2)	Pulse width. Numeric expression [s]: 0.0005 to 2.0000 (Note 3) Resolution: 1E–4
period	Pulse period. Numeric expression [s]: 0.0050 to 5.0000 (Note 4) Resolution: 1E–4 If you specify 0.0, pulse period is pulse width + TIS execution overhead seconds.
hold	Specify time to wait after trigger before forcing *peak* value. Numeric expression [s]: 0 to 655.35 Resolution: 0.01
compliance	Specify the voltage or current compliance of the *port*. REMAIN macro (Note 5) or For voltage source: numeric expression [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) For current source: numeric expression [V]: –100 to 100 or –200 to 200 (HPSMU)

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 The *width* must be less than or equal to *period* – 0.002 s. (If your TIS revision is A.1.10 or the previous, the *width* must be less than 50% of the *period*.)

Note 3 If your TIS revision is A.1.10 or the previous, 0.0005 to 0.1000

Note 4 If your TIS revision is C.04.00 or the previous, 0.0050 to 1.0000

Note 5 REMAIN can be specified only if optimization level is 0 or 1. REMAIN means the previous compliance setting if source mode of SMU is same as the previous mode. Otherwise: If source mode was VS mode and is now IS mode, compliance is set to 20 V. If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

- *port*

  To specify the SMU, use either the *port address* of SMU, or the *pin number* of any measurement pin connected to the SMU port.

- *range*

  range parameter sets the current or voltage range of the SMU.

  If you specify 0.0, the SMU is set to auto-ranging mode. In auto-ranging mode, the SMU forces the current or voltage at the lowest range that includes both *base* or *peak*.

  If you specify a non-zero value:

  The voltage range is fixed to specified range.

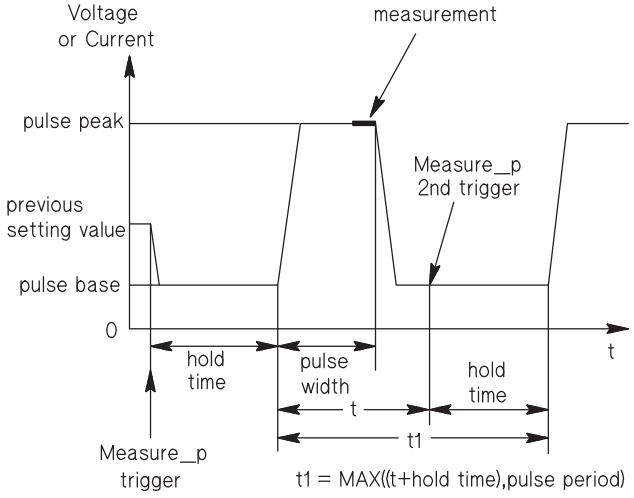
  The current range is determined by limited auto-ranging mode. The SMU forces the current at the specified range if the specified range can force the *base* and *peak*. If not, the SMU ranges up to range that can force the *base* and *peak*.

- *base*, *peak*, *width*, *period*, and *hold*

  If *mode* is set to I_SOURCE, *base* and *peak* must have same polarity.

  These determine the pulse conditions for pulsed spot measurement and staircase sweep with pulsed bias measurement as shown in following figures.

Figure 2-5 Pulsed Spot Measurements



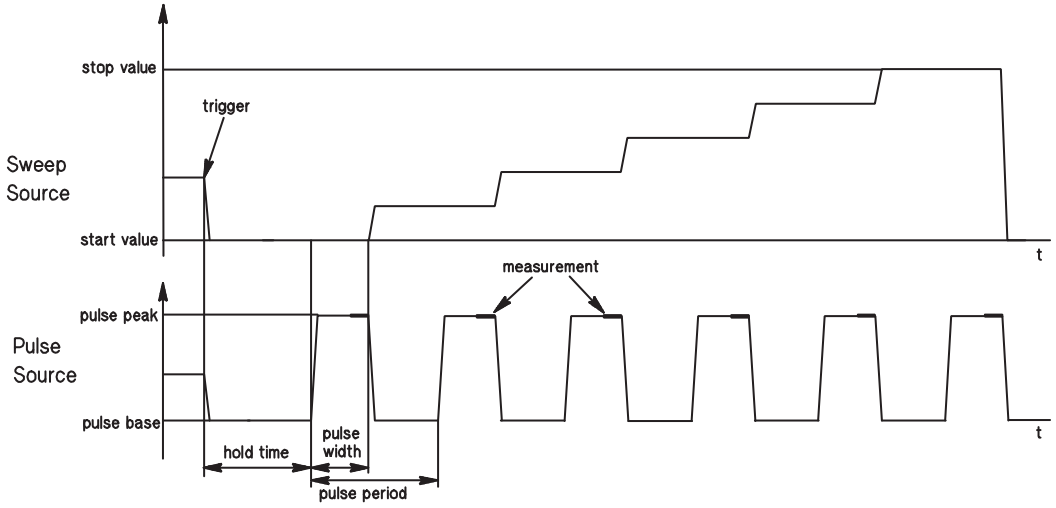
For pulsed spot measurements, execute measure_p function after you execute a set_pbias function.

A pulse is triggered by measure_p, and beginning of pulse is usually *hold* after the trigger statement. But the actual time from beginning of a pulse to the beginning of next pulse is *t1* as follows:

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

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

Figure 2-6 Staircase Sweep with Pulsed Bias Measurements



The sweep source is synchronized with the pulsed bias and ignores the hold time and delay time settings of set_iv.

For sweep measurements with pulsed bias, execute sweep_iv or rsweep_iv after you execute the set_pbias and set_iv functions.

If the following functions are executed for the specified unit after set_pbias executes, the set_pbias settings are lost. Accordingly, these functions must be executed before performing other sweep measurements after you perform a sweep measurement with pulsed bias.

```
disable_port,init_system,set_iv,set_piv
```

During pulsed measurement, DCS obtains measurement result after each A/D conversion, and SMU filters are set to OFF, ignoring set_adc settings.

Example

```
int number;
double vstart, vstop, hold_dmy, delay_dmy, comp_sweep;
double vbase, vpeak, width, period, hold, comp_bias;
double meas1[101], swp[101];
 :
 :
set_iv(24,LINEAR_V,20.0,vstart,vstop, number,hold_dmy,delay_dmy,
comp_sweep,0.0,COMP_STOP);
set_pbias(28,V_SOURCE,20.0,vbase,vpeak,width, period,hold,comp_bias);
sweep_iv(32,I_MEAS,0.0,meas1,swp,NULL);
```

See Also

- measure_p
- sweep_iv
- rsweep_iv

set_piv

Revision	A.01.00 or later
Control	SMU

This function sets the parameters for SMU pulsed sweep measurements. To execute, use the sweep_iv or rsweep_iv function.

Synopsis

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

Argument	Range Restrictions/Description	
port	SMU port address for pulsed sweep. You can specify a port address directly or a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
sweep_mode	Specify whether sweep is linear or logarithmic, voltage or current, and single or double. Use one of the following macros (or values): LINEAR_V (= 1) LINEAR_I (= 2) LINEAR_V_DBL (= 3) LINEAR_I_DBL (= 4) LOG_V (= –1) LOG_I (= –2) LOG_V_DBL (= –3) LOG_I_DBL (= –4)	

Argument	Range Restrictions/Description
range	Specify the voltage or current output range: 0 for auto-ranging mode or For voltage source: numeric expression [V]: –100 to 100 or –200 to 200 (HPSMU) For current source: numeric expression [A]: –0.1 to 0.1 or –1 to 1 (HPMSU)
base	Specify the pulse base voltage or current to which the *port* is reset after each sweep step. For voltage source: numeric expression [V]: –100 to 100 or –200 to 200 (HPSMU) For current source: numeric expression [A]: –0.1 to 0.1 or –1 to 1 (HPSMU)
start, *stop*	Specify sweep start and stop values, which determine the peak of each pulse step. For voltage source: numeric expression [V]: –100 to 100 or –200 to 200 (HPSMU) For current source: numeric expression [A] –0.1 to 0.1 or –1 to 1 (HPSMU)
number	Specify the number of sweep steps: 2 to 1001 For single sweep, there will be *number* steps from *start* to *stop*. For double sweep, there will be *number* steps from *start* to *stop*, then *number* steps from *stop* to *start*.
width (Note 2)	Pulse width: Numeric expression [s]: 0.0005 to 2.0000 (Note 3) Resolution: 1E4
period	Pulse period: Numeric expression [s]: 0.0050 to 5.0000 (Note 4) Resolution: 1E4 0: sets the pulse period to *width* + TIS execution overhead.
hold	Specify the time to wait after the trigger before forcing the first pulse step. Numeric expression [s]: 0 to 655.35 Resolution: 0.01

Argument	Range Restrictions/Description
compliance	Specify the voltage or current compliance of the *port*. REMAIN macro (Note 5) or For voltage source: numeric expression [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) For current source: numeric expression [V]: –100 to 100 or –200 to 200 (HPSMU)
stop_mode	Specify whether to continue or stop if compliance is reached. Choose one of the following macros: COMP_CONT (= 1) COMP_STOP (= 2)

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 The *width* must be less than or equal to *period* – 0.002 s. (If your TIS revision is A.01.10 or the previous, the *width* must be less than 50% of the *period*.)

Note 3 If your TIS revision is A.01.10 or the previous, 0.0005 to 0.1000

Note 4 If your TIS revision is A.01.10 or the previous, 0.0050 to 1.0000

Note 5 REMAIN can be specified only if optimization level is 0 or 1. REMAIN means the previous compliance setting if source mode of SMU is same as the previous mode. Otherwise: If source mode was VS mode and is now IS mode, compliance is set to 20 V. If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

- *base*, *start*, *stop*, *number*, *width*, *period*, and *hold*

  If SMU is a current source, pulse *base*, *start* value, and *stop* value must have same polarity.

  For log sweep mode, *start* and *stop* values must be same polarity and cannot be zero.

  These determine the pulsed sweep measurement conditions as shown in the following figure.

Figure 2-7 Single Pulsed Sweep

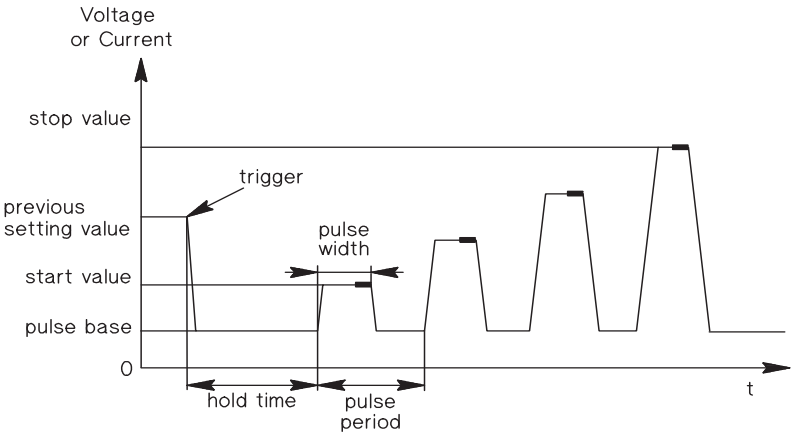
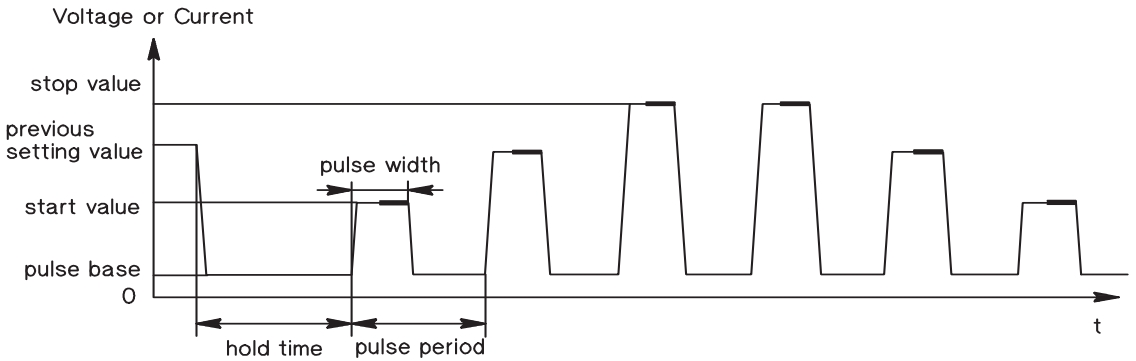


Figure 2-8 Double Pulsed Sweep



The start, stop, and step values are determined as follows:

Table 2-56 In the linear sweep mode

Item	Description
start value	resolution: depends on the set voltage or current range.
stop value (Note 1)	$\text{stop value} = \text{start value} + \text{step value} \times (\text{number of steps} - 1)$
step value	$\text{step value} = \frac{\text{stop value} - \text{start value}}{\text{number of steps} - 1}$ resolution: depends on the set voltage or current range maximum: stop value - start value

Note 1 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. Refer to force_v or force_i for setting resolution.

Table 2-57 In the log sweep mode

Item	Description
start value	resolution: depends on the set voltage or current range.
stop value (Note 1)	$\text{stop value} = \text{start value} \times 10^{\frac{\text{step value} \times (\text{number of steps} - 1)}{20}}$
step ratio	$\text{step ratio} = \frac{20 \times \log \frac{\text{stop value}}{\text{start value}}}{\text{number of steps} - 1}$ resolution: 0.02 dB/decade maximum: 20 dB/decade

Note 1 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. Refer to force_v or force_i for setting resolution.

- *port*

  To specify the SMU, use either the port address of SMU, or the pin number of any measurement pin connected to the SMU port.

- *sweep_mode*

  The *sweep_mode* determines whether the sweep is linear or logarithmic, double or single, and current or voltage:

  LINEAR_V (=1): linear single voltage sweep

  LINEAR_I (=2): linear single current sweep

  LINEAR_V_DBL (=3): linear double voltage sweep

  LINEAR_I_DBL (=4): linear double current sweep

  LOG_V (=–1): log single voltage sweep

LOG_I (=–2): log single current sweep

LOG_V_DBL (=–3): log double voltage sweep

LOG_I_DBL (=–4): log double current sweep

- *range*

  The voltage or current range is determined by the *range* value.

  If you specify *range* to 0.0, the SMU is set to auto-ranging mode. In auto-ranging mode, the SMU forces the current or voltage for each pulse sweep step and pulse base at the lowest range that includes output value.

  If you specify other than 0.0, the voltage range is fixed to specified range and the current rage is set to limited auto-ranging mode.

  In limited auto-ranging mode, the SMU forces the current at the specified range if the specified range can force *base*, *start*, and *stop*. If the specified range cannot force the *base*, *start*, and *stop*, SMU changes to the lowest output range that includes *base*, *start*, and *stop*.

- *stop_mode*

  If you set *stop_mode* to COMP_CONT, pulsed voltage or current is swept to the last specified step, even if an SMU reaches *compliance*.

  If you set *stop_mode* to COMP_STOP, the pulsed voltage or current sweep aborts and returns dummy data (9999999.99999) if an SMU reaches *compliance*.

Example

```
int number;
double vbase, vstart, vstop,width, period, hold, comp;
double meas1[101], swp[101];
 :
set_piv(24,LINEAR_V,20.0,vbase,vstart,vstop,
number,width,period,hold,comp,COMP_STOP);
sweep_iv(32,I_MEAS,0.0,meas1,swp1,NULL);
```

See Also

- sweep_iv
- rsweep_iv
- set_sync (for sweep measurements)

set_rangemode

Revision	B.03.00 or later
Control	HRSMU

This function sets the operation of the auto-ranging mode for current measurements by the HRSMU.

Synopsis

```
int set_rangemode (int port, int mode);
```

Argument	Range Restrictions/Description	
port	HRSMU port address. You can specify the port address directly or specify a pin number that is connected to the HRSMU port:	
	port address	You can specify the following macros (or values (Note 1)) SMU1 (20001), SMU2 (20002)
	pin number	1 to 49
mode	Determines operation of auto-ranging mode for current measurement. FULL_AUTORANGE_OFF (=0) FULL_AUTORANGE_ON (=1)	

Note 1 For program readability and future compatibility, it is recommended this value not be entered directly. Use the macro or PORT function instead.

- *port*

  To specify the HRSMU, use either the *port address* of the HRSMU, or the *pin number* of any measurement pin connected to the HRSMU port.

  If the specified port is not the HRSMU, this function does nothing.

- *mode*

  If FULL_AUTORANGE_ON is specified, auto-ranging mode is used for all ranges (including the 10 pA range).

If FULL_AUTORANGE_OFF is specified:

- The 1 nA limited auto-ranging mode is used for the current measurement even if you specify auto-ranging mode as a parameter of the TIS functions for current measurement, such as measure_i.
- You can specify 10 pA or 100 pA limited auto-ranging mode in the TIS functions for the current measurement and the current measurement will be executed in the specified limited auto-ranging mode.

set_skip

Revision	A.01.00 or later
Control	SMU

This function sets the parameters for step skipping method of a linear search measurement. The other parameters of the linear search measurement are set by set_lsearch function.

This function must be executed after set_lsearch function.

Synopsis

```
int set_skip (int skip, int skip_back, double sb_delay);
```

Argument	Range Restrictions/Description
skip	Determines size of large step. 1 to 20000
skip_back	Determines skip back size after coarsely finding the target value. 0 to 100
sb_delay	Delay time before starting fine search. 0 to 65.5350 [s] Resolution: 100 μs

Using step skipping method makes linear search faster. Step skipping performs the following:

1. Search unit starts to sweep with large steps until sense unit coarsely finds the target value.

   Size of large step is calculated from *skip* and *step*, which is specified by set_lsearch, as follows:

   skip × *step*
2. Search unit goes back the specified number of large steps (*skip_back*).
3. After waiting *sb_delay*, search unit starts to sweep with small steps (*step*) until sense unit accurately finds the target value.

Example

```
set_lsearch(base, collector, IS_NORM_ABOVE, start, stop, step, target,
sense_range, search_comp, delay);
set_skip(skip, skip_back, sb_delay);
search_iv(&ib, &stat, &ic);
```

See Also

- set_lsearch

set_smu

Revision	A.01.00 or later
Control	SMU

NOTE

4062 TIS function

This function is provided to execute the 4062 measurement program on the 4080. Do not use this function to make new 4080 measurement program.

This function makes all SMUs to use high-resolution A/D converter and sets measurement integration time for SMUs.

Synopsis

```
int set_smu (int integ_time, int filter, int samples);
```

Argument	Range Restrictions/Description
integ_time	Specify the measurement integration time. Specify a definite number of integrations by combining this argument with the *samples* argument or choose from one of the following macros: INTEG_MANUAL (= 0) INTEG_SHORT (= 1) INTEG_MEDIUM (= 2) INTEG_LONG (= 3)
filter	Turns the SMU filter on or off. Use one of the following macros: FILTER_ON (= 1) FILTER_OFF (= 0)
samples	Specify a number of samples to integrate the measurement value. 1 to 1023. This argument is used only when *integ_time* is set to INTEG_MANUAL.

This function makes all SMUs to use high-resolution A/D converter of 4080. So *number of samples* value for your 4062 measurement program is converted to the integration time for high-resolution A/D converter of 4080 as follows:

- If *integration time* is MANUAL

  Sets the integration time of high-resolution A/D converter to 300 μs × *number of samples*.

  If your power line frequency is 50 Hz and *number of samples* is larger than equal to 67, the integration time is rounded down by number of PLC.

  If your power line frequency is 60 Hz and *number of samples* is larger than equal to 56, the integration time is rounded down by number of PLC.

- If *integration time* is SHORT

  Sets the integration time of high-resolution A/D converter to 320 μs.

- If *integration time* is MEDIUM

  Sets the integration time of high-resolution A/D converter to 1 PLC.

- If *integration time* is LONG

  Sets the integration time of high-resolution A/D converter to 16 PLC.

set_smu_ch

Revision	A.01.00 or later
Control	SMU

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

Argument	Range Restrictions/Description	
port	SMU *measure* port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008). Specify 0 for all SMU ports.
	pin number	1 to 49
	0	All SMUs.
adc	A/D converter. You can use the following macros: PERCH_ADC (=0) REF_ADC (=1)	
filter	Filter setting. You can use the following macros: FILTER_OFF (=0) FILTER_ON (=1)	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port*

  To specify the SMU, use either the port address of SMU, or the pin number of any measurement pin connected to the SMU port.

  0 specifies for all SMU ports.

- *adc*

  Specifies whether the SMU uses the high-speed or the high-resolution A/D converter.

Macro	A/D Converter
PERCH_ADC (=0)	High-speed A/D Converter.
REF_ADC (=1)	High-resolution A/D Converter.

- *filter*

  This enables or disables the SMU output filter.

  The initial SMU 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(SMU1,1,FILTER_ON);
```

See Also

- set_adc

set_spa

Revision	D.05.16 or later
Control	SPA

This function specifies the Start and Stop frequencies, and the resolution bandwidth.

This function has a timeout function. If the SPA does not respond within 30 seconds, the function sends an error message to the standard out, and returns –1.

Synopsis

```
#include "/opt/hp4070/include/spa.h"
   int set_spa (double minf, double, maxf, double rbw);
```

Argument	Range Restrictions/Description
minf	Start frequency. 0 ≤ *minf*
maxf	Stop frequency. *maxf* ≤ 1.5E+9
rbw	Resolution bandwidth. 1E+3 ≤ *rbw* ≤ 5E+6

- *rbw*

  Reducing the resolution bandwidth lowers noise floor level. However, narrower resolution bandwidths cause longer sweep times.

  * Keysight N9000

  The resolution bandwidths above 3 MHz are 4 and 5 MHz. The bandwidths 1 KHz to 3 MHz are spaced at 10% spacing using the E24 series (24 per decade): (1.0, 1.1, 1.2, 1.3, 1.5, 1.6, 1.8, 2.0, 2.2, 2.4, 2.7, 3.0, 3.3, 3.6, 3.9, 4.3, 4.7, 5.1, 5.6, 6.2, 6.8, 7.5, 8.2, 9.1 in each decade.)

* Keysight E4411B

The resolution bandwidth is rounded to E4411B internal values as shown below.

Resolution bandwidth specified as the input parameter *rbw*	Actual bandwidth set by the Keysight E4411B
1E+3 < *rbw* ≤ 2E+3	1 kHz
2E+3 < *rbw* ≤ 6.5E+3	3 kHz
6.5E+3 < *rbw* ≤ 2E+4	10 kHz
2E+4 < *rbw* ≤ 6.5E+4	30 kHz
6.5E+4 < *rbw* ≤ 2E+5	100 kHz
2E+5 < *rbw* ≤ 6.5E+5	300 kHz
6.5E+5 < *rbw* ≤ 2E+6	1 MHz
2E+6 < *rbw* ≤ 3.6E+6	3 MHz
3.6E+6 < *rbw* ≤ 5E+6	5 MHz

NOTE

Throughput optimization tips

Broaden the resolution bandwidth. For instance, if the swept range is from 0 Hz to 500 MHz, set the resolution bandwidth to a value greater than or equal to 1 MHz.

Minimize the number of calls to set_spa(). In many cases, the same setting is used for multiple test structure. Call the set_spa() function to reset the SPA for measurements.

See Also

- measure_spa
- sweep_spa

set_sr_control

Revision	B.02.00 or later
Control	SWM

This function selects whether the control pulse for pulse switch is common for both PSC ports or independent for each PSC port.

Synopsis

```
int set_sr_control (int mode);
```

Argument	Range Restrictions/Description
mode	An integer value specifying pulse switch control mode: 0: Independent control 1: Synchronized control

- *mode* = 0

  The PSC1 port and PSC2 port are independent of each other.

  The control pulse to PSC1 can control the switch for the PSO1, PSO2, PSO3, and PSO4 ports. The control pulse to PSC2 can control the output for the PSO5, PSO6, and PSO7 ports.

- *mode* = 1

  The control pulse to PSC1 can control all switches for the pulse switch ports. The control pulse to PSC2 is ignored.

set_sr_mode

Revision	B.02.00 or later
Control	SWM

This function defines the switch control for the specified open/close pulse switch port using the control pulse.

Synopsis

```
int set_sr_mode (int port, int mode);
```

Argument	Range Restriction/Description
port	Specify open/close pulse switch port. You can specify the port address of PSO2(20262), PSO3(20263), PSO4(20264), PSO5(20265), PSC1(20268), or PSC2(20269) port.
mode	Specify control mode for open/close pulse switch. 0: Normally Close 1: Normally Open

This function defines the switch control for the specified on/off pulse switch port using the control pulse.

This function is effective for the port that is set to "Switching synchronized with output pulse" usage mode by set_sr_pg function.

- *port*

  If the specified port is PSC1, this function defines the control for PSO2, PSO3, and PSO4.

  If the specified port is PSC2, this function defines the control for PSO5.

- *mode*

  - *mode* = 0

    The switch is normally closed.

When the control pulse is at a low level, the switch opens.
When the control pulse is at a high level, the switch closes.

- *mode* = 1

  The switch is normally open.

  When the control pulse is at a low level, the switch closes.
  When the control pulse is at a high level, the switch opens.

Note that the init_system function does not reset this control mode settings.

set_sr_pg

Revision	B.02.00 or later
Control	SWM

This function specifies how switching of the pulse switch port will be controlled.

Synopsis

```
int set_sr_pg (int port, int usage, double width, double delay);
```

Argument	Range Restriction/Description	
port	Specify the pulse switch control port. You can specify the port address of PSC1(20268), PSC2(20269), or 0.	
usage	Integer value to select the switching method from the following modes: 0: Reset the corresponding switch 1: Independent switching 2: Switching synchronized with output pulse	
width	numeric expression: [s]	Specify the pulse width of control pulse. If SPGU, 5.0E–4 (Note 1) to *period* – 5E–8 (Note 2) If PGU (81150A), 5.0E–4 (Note 1) to 9.99E–1
delay	numeric expression: [s]	Specify the delay time of control pulse. If SPGU, 0 to *period* – 7.5E–8 (Note 2) If PGU (81150A), 0 to 9.99E–1

Note 1 A range of 1.0E–4 to 5.0E–4 can also be specified but functionality will not be guaranteed.

Note 2 Pulse period (*period*) is set by force_pg or prep_pg function.

- *port*

  If 0 is specified as *port*, PSC1 and PSC2 are specified.

- *usage*
  - *usage* = 0

    The state of the pulse switch is initialized as follows:

    Open/Close switch: open

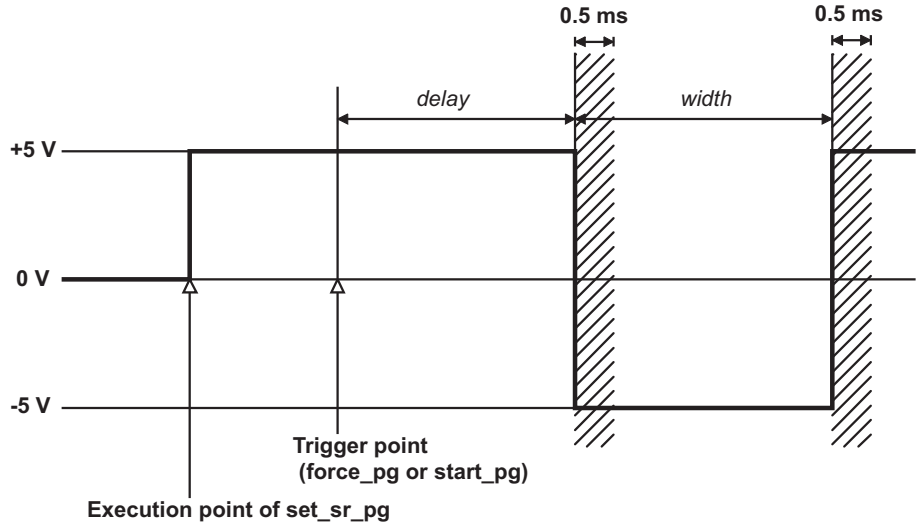
    Multiplexer switch: high impedance state
  - *usage* = 1

    The pulse switch is controlled by switch_sr function. *width* and *delay* are ignored.
  - *usage* = 2

    The pulse switch is controlled by the control pulse to the PSC port. When SPGU is used in ALWG mode, *pulse width* and *delay time* are ignored.

    The following figure shows the control pulse and relation between the pulse width(*width*), delay time(*delay*), and switching time:

Figure 2-9 Pulse Width and Delay Time



When set_sr_pg is executed, the control PG starts to force +5 V. After the trigger (force_pg or start_pg), the control PG changes voltage to –5 V and switching occurs after the delay time. Switching requires 0.5 ms for stabilization. After the pulse width time, the switches start to return the previous states, and become stable after 0.5 ms.

The following table shows the voltage of control pulse and the state of each switch:

Pulse Switch Port		State of Switch	
		Control Pulse +5 V	Control Pulse –5 V
Multiplexer Switch (PS1, PS6, PS7)		connect to PSIx1	connect to PSIx2
Open/close Switch (PS2, PS3, PS4, PS5)	Normally Close	close (enable)	open (disable)
	Normally Open	open (disable)	close (enable)

set_sync (for synchronous search measurements)

Revision	A.01.00 or later
Control	SMU

The set_sync function can be used *after* set_bsearch or set_lsearch to set up a synchronous search measurement. The set_bsearch or set_lsearch function sets up the search port, and set_sync sets up the synchronous search port.

The search is started by search_iv or rsearch_iv function.

You must set source mode (VS or IS) of synchronous search port to same source mode as search port set by set_bsearch or set_lsearch function by using force_i or force_v function.

For synchronous staircase *sweep* measurements, refer to next section: "set_sync (for synchronous sweep measurements)".

Synopsis

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

Argument	Range Restrictions/Description	
port	SMU port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
mode	Specify the polarity of the synchronization. You can choose from the following macros: POS_SYNC (= 0) NEG_SYNC (= 1)	

Argument	Range Restrictions/Description
offset	Specify offset used to calculate the synchronous values: Synchronous search port is SMU in VS mode [V]: –100 to 100 or –200 to 200 (HPSMU) Synchronous search port is SMU in IS mode [A]: –0.1 to 0.1 or –1 to 1 (HPSMU)
ratio, compliance, power_comp	Dummy parameters. Ignored for search measurements. Used only for sweep measurements.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port*

  To specify the synchronous search port (SMU), use either the port address of SMU, or the pin number of any measurement pin connected to the SMU port.

- *mode* and *offset*

  The *mode* and *offset* parameters set the output of the synchronous search port, as follows:

  - *Mode* = POS_SYNC:
    Synchronous search port output = *offset* + search port output
  - *Mode* = NEG_SYNC:
    Synchronous search port output = *offset* – search port output

  The output range of the synchronous search port is the same range as the search port.

  The specified *offset* must not cause S (see following) to be greater than ±100 V (±200 V) or ±0.1 A (±1 A), depending on the SMU type. And S determines the maximum compliance you can specify as follows.

  For Set_bsearch: S=MAX(|*min*±*offset*|, |*max*±*offset*|)

  For Set_lsearch: S=MAX(|*start*±*offset*|, |*stop*±*offset*|)

Table 2-58 In VS mode (MPSMU, HRSMU)

S (in V)	Maximum |Compliance| in mA
$0 < \|S\| \leq 20$	≤ 100
$20 < \|S\| \leq 40$	≤ 50
$40 < \|S\| \leq 100$	≤ 20

Table 2-59 In VS mode (HPSMU)

S (in V)	Maximum |Compliance| in mA
$0 < \|S\| \leq 14$	≤ 1000
$14 < \|S\| \leq 20$	≤ 700
$20 < \|S\| \leq 40$	≤ 350
$40 < \|S\| \leq 100$	≤ 125
$100 < \|S\| \leq 200$	≤ 50

Table 2-60 In IS mode (MPSMU, HRSMU)

S (in mA)	Maximum |Compliance| in V
$0 < \|S\| \leq 20$	≤ 100
$20 < \|S\| \leq 50$	≤ 40
$50 < \|S\| \leq 100$	≤ 20

Table 2-61 In IS mode (HPSMU)

S (in mA)	Maximum |Compliance| in V
$0 < \|S\| \leq 50$	≤ 200
$50 < \|S\| \leq 125$	≤ 100
$125 < \|S\| \leq 350$	≤ 40
$350 < \|S\| \leq 700$	≤ 20
$700 < \|S\| \leq 1000$	≤ 14

where compliance is the same as before this function executed.

Example

```
int drain, gate, substrate, stat;
double search;
 :
 :
if (force_v(drain, -2.0, -5, 1.15e-5) == -1) error_rep();
if (set_bsearch(gate,drain,VS_NORM_ACC,0.0,5.0, -1e-5,-1e-5,-2e-6,0,0.1) ==
-1) error_rep();
if set_sync(substrate,POS_SYNC,0.0,0.0,0.0,0.0) == -1) error_rep();
if (search_iv(&search, &stat, NULL) == -1) error_rep();
if (disable_port3(gate,drain,substrate) == -1) error_rep();
```

See Also

- set_bsearch
- set_lsearch
- search_iv
- search_iv2
- rsearch_iv

set_sync (for synchronous sweep measurements)

Revision	A.01.00 or later
Control	SMU

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

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

- Staircase sweep measurements (set_iv)
- Pulsed bias sweep 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, sweep_miv, rsweep_iv or rsweep_miv function. If set_piv is used, the synchronous sweep measurement is started by sweep_iv or rsweep_iv.

The output mode of the synchronous sweep port 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 port always performs a normal sweep even if the primary port performs pulsed sweep (set up by set_piv).

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

For synchronous *search* measurements, refer to the previous section: "set_sync (for synchronous search measurements)".

Synopsis

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

Argument	Range Restrictions/Description	
port	SMU synchronous sweep port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
mode	Specify the polarity of the synchronization. Choose from one of the following macros: POS_SYNC (= 0) NEG_SYNC (= 1)	
offset	Specify offset used to calculate the synchronous values. For voltage synchronous source: numeric expression [V]: –200 to 200 or –400 to 400 (HPSMU) For current synchronous source: numeric expression [A]: –0.2 to 0.2 or –2 to 2 (HPSMU)	
ratio	Specify the ratio used to calculate the synchronous values. If a negative ratio is desired, use NEG_SYNC for the *mode* argument. Numeric expression: 0.01 to 10 Resolution: 0.01	
compliance	Specify the voltage or current compliance on the synchronous port. REMAIN macro (Note 2) or For voltage synchronous source: numeric expression [A]: –0.1 to 0.1 or –1 to 1 (HPSMU) For current synchronous source: numeric expression [V]: –100 to 100 or –200 to 200 (HPSMU)	
power_compliance	Limits the power (voltage × current being forced and measured) applied to the *port*. Numeric expression [W]: 0.001 to 2 or 0.001 to 14 (HPSMU) Resolution: 0.001 0 means no power compliance. See description in set_iv function.	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

Note 2 REMAIN can be specified only if optimization level is 0 or 1.
REMAIN means the previous compliance setting if source mode of synchronous SMU is same as the previous mode. Otherwise:
If source mode was VS mode and is now IS mode, compliance is set to 20 V.
If source mode was IS mode and is now VS mode, compliance is set to 100 μA.

- *port*

  To specify the SMU, use either the port address of SMU, or the pin number of any measurement pin connected to the SMU port.

- *mode*, *offset*, and *ratio*

  These parameters are used to calculate the start and stop values of the synchronous sweep port as follows:

  - *Mode* = POS_SYNC:

    sync_start = *offset* + *ratio* × *primary_start*
    sync_stop = *offset* + *ratio* × *primary_stop*

  - *Mode* = NEG_SYNC:

    sync_start = *offset* − *ratio* × *primary_start*
    sync_stop = *offset* − *ratio* × *primary_stop*

  Where *primary_start* and *primary_stop* are the *start* and *stop* values specified in set_iv or set_piv.

  The specified *offset* and *ratio* must not cause *sync_start* or *sync_stop* to be greater than ±100 V (±200 V) or ±0.1 A (±1 A) depending on the SMU type.

  The number of steps for the synchronous sweep port is same as specified by *number* in set_iv or set_piv.

- *compliance*

  The output range used and the allowable voltage *compliance* or current *compliance* settings for the synchronous output port depend on the calculated *sync_start* and *sync_stop* as follows:

Table 2-62 Allowable current *compliance* Settings (MPSMU and HRSMU)

Start and Stop Values [V]	Allowable Compliance
0 ≤ |*sync_start* and *sync_stop*| ≤ 20 V	0 to ±100 mA
20 < |*sync_start* and *sync_stop*| ≤ 40 V	0 to ±50 mA
40 < |*sync_start* and *sync_stop*| ≤ 100 V	0 to ±20 mA

Table 2-63 Allowable current *compliance* Settings (HPSMU)

Start and Stop Values [V]	Allowable Compliance
0 ≤ |*sync_start* and *sync_stop*| ≤ 14 V	0 to ±1 A
14 < |*sync_start* and *sync_stop*| ≤ 20 V	0 to ±700 mA
20 < |*sync_start* and *sync_stop*| ≤ 40 V	0 to ±350 mA
40 < |*sync_start* and *sync_stop*| ≤ 100 V	0 to ±125 mA
100 < |*sync_start* and *sync_stop*| ≤ 200 V	0 to ±50 mA

Table 2-64 Allowable voltage *compliance* Settings (MPSMU and HRSMU)

Start and Stop Values	Allowable Compliance
0 ≤ |*sync_start* and *sync_stop*| ≤ 20 mA	0 to ±100 V
20 mA < |*sync_start* and *sync_stop*| ≤ 50 mA	0 to ±40 V
50 mA < |*sync_start* and *sync_stop*| ≤ 100 mA	0 to ±20 V

Table 2-65 Allowable voltage *compliance* Settings (HPSMU)

Start and Stop Values	Allowable Compliance
0 ≤ |*sync_start* and *sync_stop*| ≤ 50 mA	0 to ±200 V
50 mA < |*sync_start* and *sync_stop*| ≤ 125 mA	0 to ±100 V
125 mA < |*sync_start* and *sync_stop*| ≤ 350 mA	0 to ±40 V
350 mA < |*sync_start* and *sync_stop*| ≤ 700 mA	0 to ±20 V
700 mA < |*sync_start* and *sync_stop*| ≤ 1 A	0 to ±14 V

- *power_compliance*

  Refer to set_iv function.

Example

```
double offset, ratio, comp2;
 :
 :
set_sync(32, POS_SYNC, offset, ratio, comp2, 0.0);
sweep_iv(32, I_MEAS, 0.0, meas1, swp1, swp2);
```

See Also

- set_iv
- sweep_iv
- sweep_miv
- set_piv
- set_pbias
- rsweep_iv
- rsweep_miv

set_time_pg, set_time_pg1

Revision	B.02.00 or later
Control	PGU, SPGU

This function sets the pulse timing parameters (such as pulse width, delay time, and transition time) for the pulse that will be forced from the specified pulse generator (PG) port.

If SPGU is used in the ALWG mode, executing this function will cause an error.

Synopsis

```
int set_time_pg (int port, double width1, double delay1, double
            leading_edge1, double trailing_edge1, double width2, double
            delay2, double leading_edge2, double trailing_edge2);

int set_time_pg1 (int port, double width1, double delay1, double
            leading_edge1, double trailing_edge1);
```

Argument	Range Restrictions/Description	
port	Specify the port connected to the PG. You can specify the HF, THF, AUX, or PSI port address or pin number that currently connects to the port.	
	port address	You can specify the following values (Note 1):
		20201 to 20206, 20211 to 20213, 20251 to 20260, 20101 to 20108
	pin number	1 to 48 (Note 2)
width1, *width2*	Pulse width:	
	Numeric expression [s]:	If SPGU, 5.0E–8 to *period* – 5.0E–8 (Note 3) If PGU (81150A), 1.0E–8 to 9.99E–1 (Note 4)
delay1, *delay2*	Delay time:	
	Numeric expression [s]:	If SPGU, 0.0 to *period* – 7.5E–8 (Note 3) If PGU (81150A), 0.0 to 9.98E–1

Argument	Range Restrictions/Description	
leading_edge1, *leading_edge2*	Leading-edge transition time:	
	Numeric expression [s]:	If SPGU, 2.00E–8 to 4.00E–1 If PGU (81150A), 7.5E–9 to 2.00E–1
trailing_edge1, *trailing_edge2*	Trailing-edge transition time:	
	Numeric expression [s]:	If SPGU, 2.00E–8 to 4.00E–1 If PGU (81150A), 7.5E–9 to 2.00E–1

Resolution of the pulse timing parameters is 3 digits (10 ps minimum) for PGU (81150A). For SPGU, see Table 2-66.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use PORT function or MACRO instead.

Note 2 A specified measurement pin must be connected to an HF, THF, or AUX port.

Note 3 Pulse period (*period*) is set by force_pg or prep_pg.

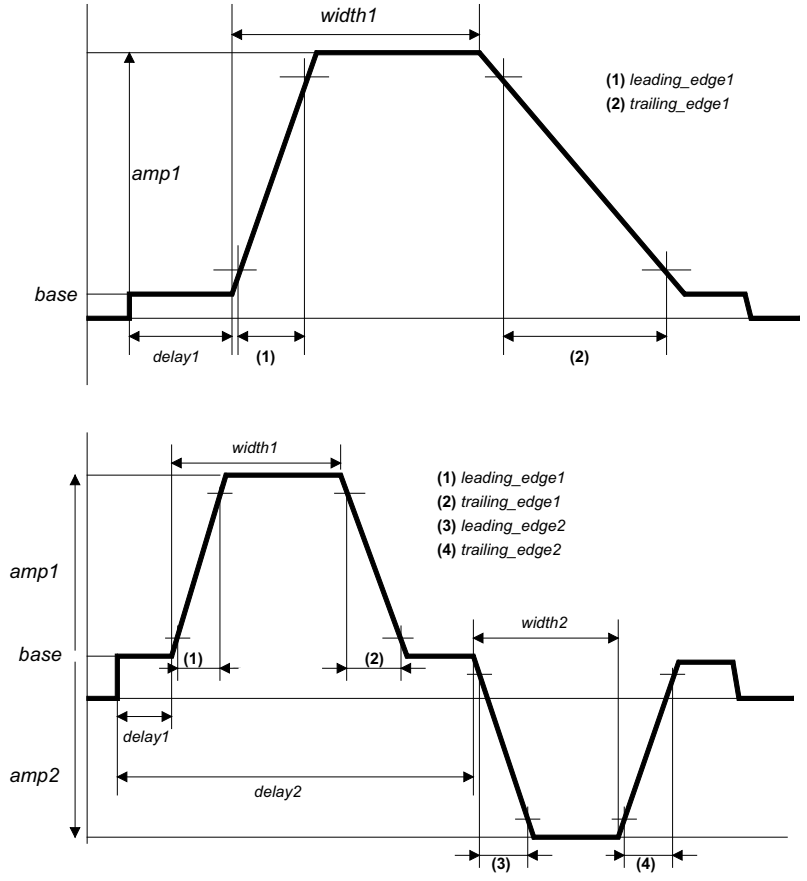
Note 4 For PGU (81150A), pulse width must satisfy the following formulas.
$width1 \geq leading\_edge1 \times 1.25$
$width2 \geq leading\_edge2 \times 1.25$

This function specifies the pulse timing parameters for the pulse generator port specified by *port address* or *pin number*. The force_pg or start_pg function sends these timing parameters to the PGs before starting outputs. And then, the system checks if applicable values are specified.

See Figure 2-10 for the definitions of pulse setup parameters. For the 3-level pulses, the definitions shown in Figure 2-11 are also effective.

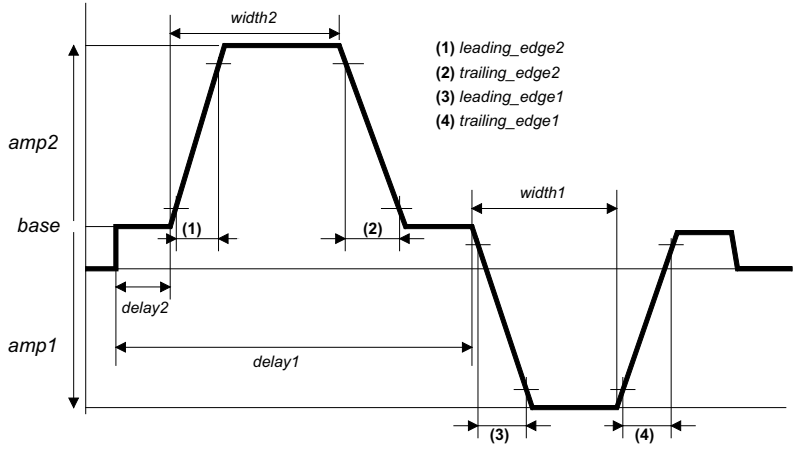
For setting pulse generator, see “set_type_pg”.

Figure 2-10 Pulse Setup Parameters



Pulse width	Time from the leading-edge to the trailing-edge.
Delay time	Time from the beginning of pulse period to the leading-edge.
Leading-edge transition time	Time taken for the transition from 10 % to 90 % of the pulse amplitude.
Trailing-edge transition time	Time taken for the transition from 90 % to 10 % of the pulse amplitude.

Figure 2-11 **Case of *delay1* ≥ *delay2* + *width2***



NOTE

PGU (81150A) 3-level pulse output

PGU can apply the 3-level pulses by using two output channels. The pulses are output from the PG11 channel. Then the PG12 channel cannot be used. The amplitude of the 3-level pulse cannot exceed 20 V.

Table 2-66 **Resolution of Pulse Timing Parameters for SPGU**

Parameter	$T_{max} \leq 8.0\ \mu s$	$T_{max} > 8.0\ \mu s$
width, *delay*	2.5 ns	10 ns
leading_edge, *trailing_edge*	2.0 ns	8.0 ns

Where, T_{max}=MAX(*leading-edge*, *trailing-edge*)

NOTE

4062F Compatibility

For 4062F TIS compatibility, this function can accept a unit address for the 4062F PGU, and automatically convert it to a unit address for 4080 by using a port mapping table.

set_timestamp

Revision	A.01.00 or later
Control	n.a.

This function can enable or disable the return of time stamp data by the following measurement-related functions:

- force_meas_t
- measure_cpgt
- measure_csrst
- measure_it
- measure_m
- measure_vt
- measure_ztrt
- port_status_t
- rsearch_iv
- rsweep_ivt
- rsweep_mivt
- search_iv2
- status_cvf
- status_miv

Synopsis

```
int set_timestamp (int on);
```

Argument	Range Restrictions/Description
on	0: disable Other than 0: enable

If *on* is set to enable, the measurement related functions return time stamp data with each measurement data.

If *on* is set to disable, the measurement related functions return 0 as time stamp data.

set_type_pg

Revision	B.02.00 or later
Control	PGU, SPGU

This function sets the pulse output mode for the pulse generator (PG) connected to the specified port.

The pulse generator can force two types of pulse wave, which are 2-level pulse output and 3-level pulse output. For the image of output pulses, see Figure 2-10 on page 460.

The actual output pulse to be forced to the measurement pin depends on the specified port and pulse generator configuration.

If SPGU is used in the ALWG mode, executing this function will cause an error.

Synopsis

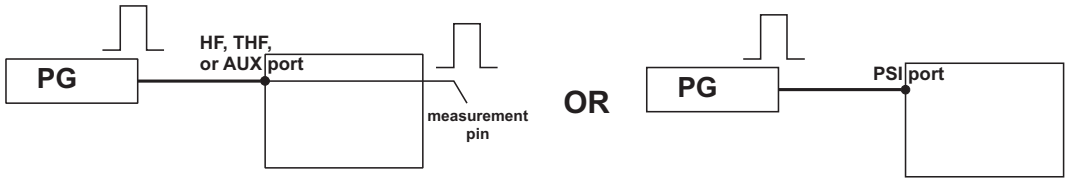
```
int set_type_pg (int port, int mode);
```

Argument	Range Restrictions/Description	
port	Specify the port connected to the PG. You can specify the HF, THF, AUX, or PSI port address or pin number that currently connects to the port.	
	port address	You can specify the following values (Note 1):
		20201 to 20206, 20211 to 20213, 20251 to 20260, 20101 to 20108
	pin number	1 to 48
mode	Specify the output mode of the PG. 1: "2-level pulse" output mode 2: "3-level pulse" output mode	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use PORT function instead.

To output "2-level pulse"

Specify an HF, THF, or AUX port connected to a PG, and set the *mode* to "2-level pulse".



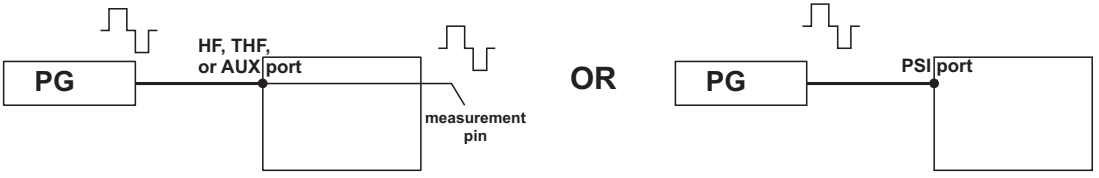
To set the pulse output level, use the set_level_pg function. The setup parameters *amp1* (pulse amplitude1) and *base* (pulse base) determine the pulse voltage. *amp2* is ignored.

To set the pulse timing parameters, use the set_time_pg1 function. All the timing parameters determine the timing of the output pulse.

If the specified port connects to the pulse switch port, use the set_sr_pg and switch_sr functions to connect the PG to the specified port.

To output "3-level pulse"

Specify an HF, THF, or AUX port connected to a PG, and set the *mode* to "3-level pulse".



To set the pulse output level, use the set_level_pg function. The setup parameters *amp1* (pulse amplitude1), *amp2* (pulse amplitude2), and *base* (pulse base) determine the pulse voltage.

To set the pulse timing parameters, use the set_time_pg function. All the timing parameters determine the timing of the output pulse.

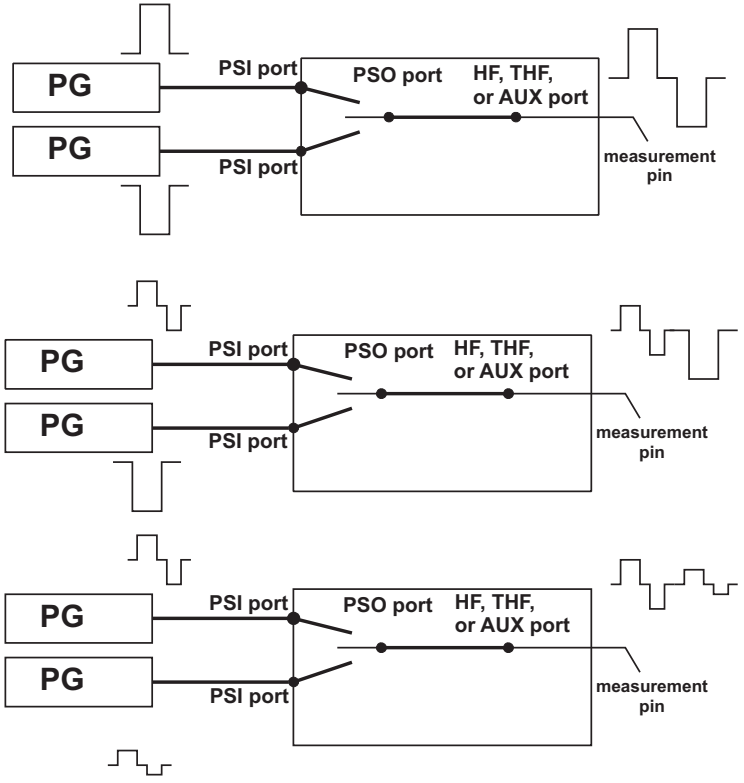
If the specified port connects to the pulse switch port, use the set_sr_pg and switch_sr functions to connect the PG to the specified port.

PGU (81150A) can output 3-level pulses by using two output channels. The pulses are output from the PG11 channel. Then the PG12 channel cannot be used. The amplitude of the “3-level pulse” cannot exceed 20 V.

With SPGU, *each* SPGU channel can output 3-level pulses.

To use multiplexer

Pulse switch has multiplexers which have two inputs and one output and are used to switch the input to connect to the output. Multiplexers can be used to generate multilevel pulse signals as shown below.



Connect pulse generators (PG) to the multiplexer input ports (PSI*x*1 and PSI*x*2, *x*: 1, 6, or 7) and connect the multiplexer output port (PSO*x*) to an HF, THF, or AUX port.

To set the output pulse, execute the set_type_pg, set_level_pg, and set_time_pg, set_time_pg1 functions for each PG. Then specify the PSI port connected to the PG as *port*.

To control the multiplexer, use the set_sr_pg and switch_sr (or switch_sr_wait) functions. For set_sr_pg, set 0 to *port* and set 1 to *usage*.

For using SPGU:

When using SPGU with more than two channels, one channel can be used to control the multiplexer. This makes possible to output multilevel pulses synchronized with the output pulses from the control PG.

To control multiplexer:

- Connect the control PG to the multiplexer PSC port.
- Use the set_sr_control function to specify the PSC port.
- Use the set_sr_pg function to set the output pulses from the control PG.

  Then specify the PSC port to *port* and set 2 to *usage*. And set the timing parameters of the control pulse to *width* and *delay*.
  It is no need to set the pulse level. It is set automatically.

set_vth

Revision	A.01.00 or later
Control	SMU

This function sets the parameters for an FET gate-source threshold voltage measurement that uses linear search with step skipping and least squares method. The measure_vth function starts the threshold voltage measurement.

Synopsis

```
int set_vth (int gate_port, int drain_port, double vg_start, double vg_stop,
             double vg_step, double id_start, double id_measrange, int vg_skip,
             int vg_stepback, double sb_delay, double gate_comp, double
             delay, int chan_type);
```

Argument	Range Restrictions/Description	
gate_port	Gate SMU port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
drain_port	Drain SMU port address. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
vg_start	Gate voltage to start searching *id_start*. –100 to 100 [V] or –200 to 200 [V] (HPSMU)	

Argument	Range Restrictions/Description
vg_stop	Maximum gate voltage of search for *id_start*. –100 to 100 [V] or –200 to 200 [V] (HPSMU)
vg_step	Step size of gate voltage. 0 < |*vg_step*|≤ 100 0 < |*vg_step*|≤ 200 (HPSMU)
id_start	Drain current at which to start fine search for maximum gradient point. –0.1 to 0.1 [A] or –1 to 1 [A] (HPSMU)
id_measrange	Current measurement range of drain SMU. –0.1 to 0.1 [A] or –1 to 1 [A] (HPSMU)
vg_skip	Factor used to determine large step size. 1 to 20000
vg_skipback	Number of large steps to go back after finding *id_start*. 0 to 100
sb_delay	Delay time between skip back and start of fine search for maximum gradient point. 0 to 65.5350 [s] with 0.0001 resolution
gate_comp	Gate current compliance. –0.1 to 0.1 [A]
delay	Delay time for measurement at each step. 0 to 65.5350 [s] with 0.0001 resolution
chan_type	FET channel type. N type: 0 or positive integer. P type: negative integer.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

After set_vth sets the search parameters, measure_vth performs the threshold voltage measurement as follows:

1. Gate SMU starts to sweep with large steps from *vg_start* until drain SMU finds *Id_start*.

Size of large step is calculated from *vg_skip* as follows:

vg_skip × *vg_step*

id_start must be drain current before FET reaches threshold. To make measurement faster, you need to specify *id_start* close to the threshold, but before the threshold.

2. Gate SMU goes back the specified number of large steps (*vg_skipback*)
3. After waiting *sb_delay*, gate SMU starts to sweep with small steps (*vg_step*) until the gradient of Vgs-Ids or Vgs-√{Ids} curve is maximum.
4. The X-intercept of line tangent to this maximum gradient point is returned to measure_vth as the threshold voltage. Refer to measure_vth.

Example

```
connect_pin(drain, drain_pin);
connect_pin(gate, gate_pin);
connect_pin(source, source_pin);
connect_pin(substrate, substrate_pin);
force_v(drain, vd, vd_range, d_comp);
force_v(source, vs, vs_range, s_comp);
force_v(substrate, vsb, vsbrange, sb_comp);
set_vth(gate, drain, vg_start, vg_stop, vg_step, id_start, meas_range, 3,
20, 0, 0.1, 0, 0);
measure_vth(VTH_SAT, &vth, &status, &gm, &id, num_samples, vd);
```

See Also

- measure_vth
- set_lsearch
- set_skip

set_wait_time

Revision	A.01.00 or later
Control	SMU

This function changes the system default wait time for forcing voltage or current.

The 4080 has predefined system default wait times (which are optimized for output range and other settings) for forcing voltage or current. This is time to wait for settling of the output before forcing next value or making measurement.

Synopsis

```
int set_wait_time (double next_force_wait, double meas_wait);
```

Argument	Range Restrictions/Description
next_force_wait	Wait time factor between forcing an output value and forcing next value. 0 to 10 with 0.1 resolution
meas_wait	Wait time factor between an output value and performing measurement. 0 to 10 with 0.1 resolution

- *next_force_wait*

  The wait time between forcing an output value and forcing next value is calculated as *next_force_wait* × *system predefined default*.

- *meas_wait*

  The wait time between forcing an output value and performing measurement is calculated as *meas_wait* × *system predefined default*.

Example

```
set_wait_time(2,1);
```

set_zf

Revision	C.04.00 or later
Control	HSCMU

This function sets the sweep parameters for Z/θ-f measurements by the HSCMU.

To start the sweep measurements, use the sweep_zf function.

To set the test signal level, use the set_cmu function.

To set the DC bias, use the force_v function.

Synopsis

```
int set_zf (double start, double stop, double hold, double delay, int
            stop_mode);
```

Argument	Range Restrictions/Description
start, *stop*	Specify the start and stop measurement frequencies. 1 kHz to 2 MHz
hold	Specify time to wait after the *start* measurement frequency is set. If a non-zero *delay* is set, an additional *delay* time follows after the *hold* time. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms
delay	Specify the time to wait after each measurement frequency change before making measurement. Numeric expression [s]: 0.0 to 650.0 Resolution = 1 ms
stop_mode	Specify whether to continue or stop if HSCMU reports an error. Choose from one of the following macros: COMP_CONT (= 1) COMP_STOP (= 2)

- *start* and *stop*

  The HSCMU executes the impedance and phase measurement at all measurement frequencies between *start* and *stop* that can be set by HSCMU.

To know how many frequency steps are actually set, use the get_freq_step function.

- *hold*

  The HSCMU waits *hold* after setting the first measurement frequency (*start)*.

- *Fdelay*

  For each measurement frequency, HSCMU waits *delay*, then measures.

- *stop mode*

  If you set *stop mode* to COMP_CONT, the frequency sweep *continues* to the last specified step, even if an HSCMU reports an error.

  If you set *stop mode* to COMP_STOP, the frequency sweep *aborts* and returns dummy data (9999999.99999) if an HSCMU reports an error.

short_cal

Revision	A.01.10 or later
Control	SMU, HSCMU, SPGU

This function sets all system hardware resources to their initial status. It also starts the self-calibration function of the SMUs, HSCMU, and SPGUs.

Synopsis

```
int short_cal (void);
```

This function is exactly the same as the long_cal function.

You should perform calibration after your system has fully warmed up to correct for offsets caused by temperature drifts.

For best measurement results, perform calibration when the ambient temperature changes more than 3 °C (6 °F).

Example

```
err=short_cal();
```

See Also

- init_system
- long_cal

short_corr84

Revision	A.01.00 or later
Control	CMU, (HSCMU)

NOTE

CMU Control TIS function

This function can be used to execute 4080 measurement program for using CMU on the 4080 with HSCMU.

This function performs nothing for the HSCMU without an error.

Do not use this function to make new measurement program for using HSCMU.

This function performs the SHORT compensation measurement for the CMU.

Before executing this function, you must disconnect the CMU connection cables at testhead end, and connect the cables to each other.

Synopsis

```
int short_corr84 (void);
```

This function triggers the SHORT compensation function in the CMU, and compensation data is stored in the internal non-volatile memory of the CMU.

Refer to "CMU Error Compensation" in the *User's Guide* for more information about the error compensation theory for the CMU.

Example

```
short_corr84();
```

See Also

- open_corr84

short_corr_th

Revision	A.01.10 or later
Control	HSCMU, CMU

This function performs the SHORT compensation measurement for the HSCMU on the two measurement pins which are connected to CMH and CML and enables error compensation beyond measurement pins, for example, up to the probe needles.

Before executing this function:

- Connect CMH and CML to the desired measurement pins.
- Shorten the probe needles connected to the specified measurement pins.

Synopsis

```
int short_corr_th (void);
```

This function measures SHORT error factors beyond two capacitance measurement pins, for example, up to the probe needles.

For measuring OPEN error factors, use open_corr_th function.

After executing this function, the 4080 automatically compensates capacitance measurement values by using measured error factors until ending the test session.

You can execute this function for any two measurement pins (but not chuck connection pin).

By repeating this function to the different combination of measurement pins, you can measure SHORT error factors for up to 500 different combination of measurement pins.

You can store measured compensation data into the file by using store_corr_data function and re-load the data for error-compensation later or in another test session by using load_corr_data function.

NOTE

You must re-measure if the conditions change

When the conditions change, re-measure the error compensation data. If you don't re-measure, the 4080 performs error compensation incorrectly.

For example, if the probe card is changed, you must re-measure the error compensation data.

Example

```
short_corr_th();
```

See Also

- open_corr_th
- disable_corr
- status_corr

spgu_alwg_force

Revision	D.05.10 or later
Control	SPGU

This function repeatedly forces the waveform data loaded by the spgu_alwg_load function for a specified number of times.

Synopsis

```
int spgu_alwg_force (double repeat);
```

Argument	Range Restrictions/Description
repeat	Specify the number of repetition. 0 to 1000000000

This function repeatedly forces the waveform data (one period data) for a specified number of times. To repeat infinitely, specify 0 as *repeat*.

If *repeat* is a non-zero value, this function instructs the SPGU to force the specified number of periods and then returns program execution to the program that called this function. You cannot abort the waveform data output when the data are being forced from the measurement program.

If *repeat* is 0, this function makes the SPGU start to force the waveform data continuously, and immediately returns the program execution to the program. The waveform data will continue to be forced until the next stop_pg, connect_pin, connect_pin2, connect_pin3, connect_th, init_system, or tap_port function execution.

If *repeat* is a non-integer, it will be rounded down. For example, if you specify 13.7, the data will be forced 13 times.

See Also

- spgu_alwg_load

spgu_alwg_load

Revision	D.05.10 or later
Control	SPGU

This function loads the ALWG setting data (arbitrary linear waveform data) from a file.

Synopsis

```
int spgu_alwg_load (char *filename);
```

Argument	Range Restrictions/Description
filename	Specify the name of the file that stores the ALWG setting data.

This function loads the ALWG setting data (arbitrary linear waveform data) from the specified file. Use the spgu_alwg_force function to actually force the waveform for a specified number of times.

For information about ALWG setting data, see the *Keysight 4080 User's Guide*.

Example

```
spgu_mode(1);
spgu_alwg_load("spgu-dat");
```

See Also

- spgu_alwg_force

spgu_alwg_r

Revision	D.05.10 or later
Control	SPGU

This function specifies the impedance value of the device under test (DUT) that is used in the ALWG mode.

Synopsis

```
int spgu_alwg_r (int port, double impedance);
```

Argument	Range Restrictions/Description	
port	The port that connects to the SPGU. Specify the HF, THF, AUX, or PSI port address or pin number that currently connects to the port.	
	port address:	Specify the following values (Note 1) (or macros): HF1 (20201) to HF6 (20206), THF1 (20211) to (THF3) 20213, PSI11 (20251) to PSI72 (20260), AUX1 (20101) to AUX8 (20108), 20281 to 20290
	pin number:	1 to 48
impedance	Load impedance [Ω]. 0.0 for default setting (Note 2) or 0.1 to 1E6	

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

Note 2 The priority of default setting is determined in the order: last value specified as *impedance* > value measured by spgu_measure_r > 1MΩ.

The *impedance* is the impedance value of the device under test (DUT), used to adjust the output voltage from SPGU. For more details, see the description of *impedance* for set_level_pg.

See Also

- set_level_pg

spgu_measure_r

Revision	D.05.10 or later
Control	SPGU

This function performs voltage measurement by using the specified SPGU and returns the measured voltage and load resistance values.

Synopsis

```
int spgu_measure_r (int port, double vs, double rs, double *vm, double
                    *rm, double delay);
```

Argument	Range Restrictions/Description	
port	The port that connects to the SPGU. Specify the HF, THF, AUX, or PSI port address or pin number that currently connects to the port.	
	port address:	Specify the following values (Note 1) (or macros): HF1 (20201) to HF6 (20206), THF1 (20211) to (THF3) 20213, PSI11 (20251) to PSI72 (20260), AUX1 (20101) to AUX8 (20108), 20281 to 20290
	pin number:	1 to 48
vs	Voltage value to force [V]. (–40 to 40 @ *rs*=1E6, –20 to 20 @ *rs*=50) (Note 2) $\|vs\| \geq 0.1$ V	
rs	Expected resistance value [Ω]. (0.1 to 1E6)	
vm	Pointer to the double variable which is used to return the measured voltage value.	
rm	Pointer to the double variable which is used to return the measured resistance value.	
delay	Delay time between voltage force and measurement [s]. (0 to 9.9)	

Note 1 For program readability and future compatibility, we recommend that you avoid using this value directly and use macro or the PORT function instead.

Note 2 $\pm 40 \times rs$ / (50 + *rs*), max current = 400 mA

This function should be called before other functions that sets up SPGU such as set_level_pg or spgu_alwg_load. Executing spgu_measure_r sets the SPGU to the initial state.

The specified port must be connected to the device before measurement.

The *rs* parameter can be a rough value but is used to force voltage to the device as strictly as possible.

The resistance value measured by this function can be used for the *impedance* parameter of the set_level_pg function.

This function performs averaging. The measurement interval is 10.0 μs (50 Hz power line frequency) or 8.33 μs (60 Hz power line frequency), and the number of averaging is 2000. The measurement is performed for one PLC (power line cycle).

NOTE

Recommended load impedance range for accurate measurement

SPGU performs a simplified measurement to measure the load resistance. The measurement accuracy will decline as resistance increases. It is recommended to perform measurement under the following conditions:

Force voltage: ≥ 1 V
Minimum load resistance: 40 Ω
Maximum load resistance: 500 Ω at 1 V, 2 kΩ at 5 V, 5 kΩ at 10 V

Example

```
connect_pin(PORT(2,1),10);
spgu_measure_r(10,vs,rs,&vm,&rm,0.0);
 :
 :
set_level_pg(10,amp1,amp2,b_level,0.0);
```

See Also

- set_level_pg

spgu_mode

Revision	D.05.10 or later
Control	SPGU

This function sets the operation mode of SPGU.

Synopsis

```
int spgu_mode (int mode);
```

Argument	Range Restrictions/Description
mode	Specify SPGU operation mode: 0: pulse generation (PG) 1: arbitrary linear waveform generation (ALWG)

Specify SPGU operation mode as *mode*. (Initial setting is 0 (PG).)

Executing this function initializes SPGU. If you change the SPGU operation mode, you will need to reconfigure all the necessary settings for SPGU. (Previous settings are not restored by this function.)

Example

```
spgu_mode(1);
```

See Also

- spgu_mode_q

spgu_mode_q

Revision	D.05.10 or later
Control	SPGU

This function returns the current operation mode of SPGU.

Synopsis

```
int spgu_mode_q (int *mode);
```

Argument	Range Restrictions/Description
mode	Specify the pointer which is used to return the SPGU operation mode: 0: pulse generation (PG) 1: arbitrary linear waveform generation (ALWG)

See Also

- spgu_mode

stand_by_port, stand_by_port2, stand_by_port3

Revision	A.01.00 or later
Control	SMU, RFU

These functions set SMU to Standby mode. If an SMU is in Standby mode, the SMU stays in Active state (forcing output) even if a connection-type function or tap_port function is executed for another SMU, HSCMU, or CMU port.

If an SMU is *not* in Standby mode, the SMU changes from Active state to Zero Output state when a connection-type function or tap_port function is executed for another SMU, HSCMU, or CMU port. Then, when next force_i, force_v, or force_meas function is executed, and if SMU is still connected to a measurement pin or chuck, the SMU is returned to its previous Active state.

Synopsis

```
int stand_by_port (int port1);
int stand_by_port2 (int port1, int port2);
int stand_by_port3 (int port1, int port2, int port3);
```

Argument	Range Restrictions/Description	
port1, *port2*, *port3*	SMU port addresses. You can specify the port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008), RFU11 (20301), RFU12 (20302)
	pin number	1 to 49 for SMU, 101 to 105 for RFU11, 201 to 205 for RFU12
	0	If you specify *port1* as 0, this function disables Standby mode for all SMU/RFU ports. If you specify *port2* or *port3* as 0, the following parameters are ignored.

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

See Also

- disable_stand_by_port

start_pg

Revision	B.02.00 or later
Control	PGU, SPGU

This function forces pulses from all pulse generators (PG) that are ready. If SPGU is used in the ALWG mode, executing this function will cause an error.

Synopsis

```
int start_pg (double count);
```

Argument	Range Restrictions/Description
count	Specify how many times the PGs force the period of pulse. 0.0 to 1000000000.0 (Note 1)

Note 1 The fractional part is truncated.

This function forces pulses from the following PGs:

- All PGs that connect to the HF, THF, PS, or AUX ports that are connected to the measurement pins using connect_pin, connect_th, or connect_chuck functions.
- All PGs that do not connect to the HF, THF, PS, or AUX ports and are output-enabled by ch_sw_on function.

You must set up these PGs using the set_level_pg, set_time_pg, prep_pg, or load_pg function, before executing this function.

- *count*

  If the specified *count* is 0, PGs will force the pulses continuously until the next stop_pg, connect_pin, connect_th, init_system, or tap_port execution.

See Also

- prep_pg
- stop_pg

status_corr

Revision	A.01.10 or later
Control	HSCMU, CMU

This function returns if there is OPEN and/or SHORT error compensation data, which is measured by open_corr_th or short_corr_th function, for capacitance measurement beyond specified two measurement pins.

Synopsis

```
int status_corr (int h_pin, int l_pin, int *open_status, int *short_status);
```

Argument	Range Restrictions/Description
h_pin	Specify SWM pin number to connect to CMH port. 1 through 48
l_pin	Specify SWM pin number to connect to CML port. 1 through 48
open_status	Pointer to integer variable in which to store the returned OPEN compensation data status.
short_status	Pointer to integer variable in which to store the returned SHORT compensation data status.

The status_corr function returns if there is OPEN and/or SHORT error compensation data, which is measured by open_corr_th or short_corr_th function, for capacitance measurement beyond specified two measurement pins.

- *open_status*, *short_status*

  If the value other than 0 is returned to *open_status* and/or *short_status*, there is OPEN and/or SHORT error compensation data and the 4080 automatically performs error compensation for capacitance measurement on the *h_pin* and *l_pin* measurement pins.

Example

```
status_corr(1, 2, &open_stat, &short_stat);
```

See Also

- open_corr_th
- short_corr_th
- disable_corr

status_cvf

Revision	C.04.00 or later
Control	HSCMU

This function gets the status of the data measured by sweep_cvf.

Synopsis

```
int status_cvf (double freq[], double bias[], int status[], double
                timestamp[]);
```

Argument	Range Restrictions/Description
freq	Pointer to double array in which to store the measurement frequencies set by set_cvf.
bias	Pointer to double array in which to store the values of DC bias steps set by set_cvf.
status	Pointer to integer array in which to store the returned measurement status.
timestamp	Pointer to double array in which to store the returned time stamp of each measurement data. If you do not need this data, specify NULL.

- *freq*

  The measurement frequencies set by set_cvf are returned to this array.

  The size of this array must be equal to or larger than the number of measurement frequencies set by the set_cvf function.

- *bias*

  The values of DC bias sweep steps are returned to this array.

  The size of this array must be equal to or larger than the sweep steps set in the set_cvf function.

- *status*, *timestamp*

  The status of measurement data are returned to this array.

  The meaning of the status value is as follows:

  0: Normal

  1: Analog bridge is unbalanced

  2: DC bias source is overloaded

  3: A/D converter is not working

  4: Sweep measurement has been aborted

  5: Measurement overflow

  If two or more conditions occur during a measurement, the priority of the returned numerical code is 1>2>5>3>4.

  The time stamp of measurement data are returned to this array.

  The size of the array must be equal to Nf × Nv, where Nf is the number of measurement frequencies and Nv is the number of DC bias steps.

  The data is stored as (F0, V0), ... , (F0, Vn), (F1, V0), ... , (Fn, Vn), where (Fn, Vn) is the data for nth measurement frequency and nth DC bias step.

status_miv

Revision	A.01.00 or later
Control	SMU

Returns the status of the data measured by sweep_iv, sweep_miv, or rsweep_miv.

Synopsis

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

Argument	Range Restrictions/Description
n	Specify the number of ports: 1 to 8.
ports	Specify a pointer to an integer array that contains the SMU port addresses or pin numbers connected to the ports.
statuses	Specify a pointer to an integer array in which to return the measurement statuses.
times	Specify a pointer to a double array in which to return the time stamp values.

- *ports* array

  To specify the measurement ports. You can specify port addresses of SMUs or pin numbers of any measurement pin connected to the SMU ports.

- *statuses* array and *times* array

  Measurement status and time stamp values are returned to elements of these arrays that correspond to ports specified in *ports* array. For status and time stamp, refer to the port_status_t function.

See also

- port_status_t
- sweep_iv
- sweep_miv
- rsweep_miv

stoh

Revision	D.05.10 or later
Control	n.a.

This function transforms S parameters into H parameters.

The following equation is used for the S-to-H transformation:

$$\begin{bmatrix} h11 & h12 \\ h21 & h22 \end{bmatrix} = \begin{bmatrix} \frac{(1+S11)(1+S22)-S12\times S21}{(1-S11)(1+S22)+S12\times S21} & \frac{2\times S12}{(1-S11)(1+S22)+S12\times S21} \\ \frac{-2\times S21}{(1-S11)(1+S22)+S12\times S21} & \frac{(1-S22)(1-S11)-S12\times S21}{(1-S11)(1+S22)+S12\times S21} \end{bmatrix}$$

$$\begin{bmatrix} H11 & H12 \\ H21 & H22 \end{bmatrix} = \begin{bmatrix} Z0\times h11 & h12 \\ h21 & \frac{h22}{Z0} \end{bmatrix}$$

Synopsis

```
void stoh (int num, COMPLEX s11[], COMPLEX s21[], COMPLEX s12[], COMPLEX
           s22[], COMPLEX h11[], COMPLEX h21[], COMPLEX h12[], COMPLEX
           h22[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
s11	Array containing the S11 parameters.
s21	Array containing the S21 parameters.
s12	Array containing the S12 parameters.
s22	Array containing the S22 parameters.

Argument	Range Restrictions/Description
h11	Return array for H11 parameters transformed from S parameters.
h21	Return array for H21 parameters transformed from S parameters.
h12	Return array for H12 parameters transformed from S parameters.
h22	Return array for H22 parameters transformed from S parameters.
z0	Characteristic impedance Z0.

If the absolute value of the denominator in the above equation $[(1-S11)\times(1+S22)+S12\times S21\ ]$ is equal to 0 (zero) or if Z0 is equal to 0 (zero), $1\times10^{+99}$ + $(1\times10^{+99})$j is returned to the H parameters.

See Also rfs_z0_r

stop_pg

Revision	B.02.00 or later
Control	PGU, SPGU

This function stops the continuous pulses from the pulse generators (PG) that are triggered by start_pg, force_pg, or spgu_alwg_force.

Synopsis

```
int stop_pg (void);
```

See Also

- force_pg
- start_pg
- spgu_alwg_force

store_corr_data

Revision	A.01.10 or later
Control	HSCMU, CMU

This function stores compensation data for capacitance measurement, which is measured by open_corr_th and short_corr_th, into the specified file.

Synopsis

```
int store_corr_data (char *file_specifier);
```

Argument	Range Restrictions/Description
file_specifier	A character string that specifies the name of a file where the compensation data is stored and may include an absolute path name. Maximum length: 1023 characters.

This function stores OPEN and SHORT compensation data, which is measured by open_corr_th and short_corr_th functions, beyond switching matrix measurement pins, for example up to the probe needles, into the specified file, so that you can re-load the data for error-compensation later or in another test session by using load_corr_data function.

Example

```
store_corr_data("c-corr");
```

See Also

- load_corr_data
- open_corr_th
- short_corr_th

stoy

Revision	D.05.10 or later
Control	n.a.

This function transforms S parameters into Y parameters.

The following equation is used for the S-to-Y transformation:

$$\begin{bmatrix} y11 & y12 \\ y21 & y22 \end{bmatrix} = \begin{bmatrix} \dfrac{(1+S22)(1-S11)+S12\times S21}{(1+S11)(1+S22)-S12\times S21} & \dfrac{-2\times S12}{(1+S11)(1+S22)-S12\times S21} \\ \dfrac{-2\times S21}{(1+S11)(1+S22)-S12\times S21} & \dfrac{(1+S11)(1-S22)+S12\times S21}{(1+S11)(1+S22)-S12\times S21} \end{bmatrix}$$

$$\begin{bmatrix} Y11 & Y12 \\ Y21 & Y22 \end{bmatrix} = \frac{1}{Z0} \times \begin{bmatrix} y11 & y12 \\ y21 & y22 \end{bmatrix}$$

Synopsis

```
void stoy (int num, COMPLEX s11[], COMPLEX s21[], COMPLEX s12[], COMPLEX
           s22[], COMPLEX y11[], COMPLEX y21[], COMPLEX y12[], COMPLEX
           y22[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
s11	Array containing the S11 parameters.
s21	Array containing the S21 parameters.
s12	Array containing the S12 parameters.
s22	Array containing the S22 parameters.

Argument	Range Restrictions/Description
y11	Return array for Y11 parameters transformed from S parameters.
y21	Return array for Y21 parameters transformed from S parameters.
y12	Return array for Y12 parameters transformed from S parameters.
y22	Return array for Y22 parameters transformed from S parameters.
z0	Characteristic impedance Z0.

If the absolute value of the denominator in the above equation [(1+S11)×(1+S22)–S12×S21] is equal to 0 (zero) or if Z0 is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the Y parameters.

See Also rfs_z0_r

stoy1

Revision	D.05.10 or later
Control	n.a.

This function transforms one-port S parameters into one-port Y parameters.

The following equation is used for the S-to-Y transformation:

$$Y = \frac{1}{Z0} \times \frac{1 - S}{1 + S}$$

Synopsis

```
void stoy1 (int num, COMPLEX s[], COMPLEX y[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
s	Name of array containing one-port S parameters.
y	Return array for one-port Y parameters transformed from the S parameters.
z0	Characteristic impedance Z0.

If S is equal to –1 or Z0 is equal to 0 (zero), $1 \times 10^{+99} + (1 \times 10^{+99})j$ is returned to the Y parameters.

See Also rfs_z0_r

stoz

Revision	D.05.10 or later
Control	n.a.

This function transforms S parameters into Z parameters.

The following equation is used for the S-to-Z transformation:

$$\begin{bmatrix} z11 & z12 \\ z21 & z22 \end{bmatrix} = \begin{bmatrix} \frac{(1+S11)(1-S22)+S12\times S21}{(1-S11)(1-S22)-S12\times S21} & \frac{2\times S12}{(1-S11)(1-S22)-S12\times S21} \\ \frac{2\times S21}{(1-S11)(1-S22)-S12\times S21} & \frac{(1-S11)(1+S22)+S12\times S21}{(1-S11)(1-S22)-S12\times S21} \end{bmatrix}$$

$$\begin{bmatrix} Z11 & Z12 \\ Z21 & Z22 \end{bmatrix} = Z0\times \begin{bmatrix} z11 & z12 \\ z21 & z22 \end{bmatrix}$$

Synopsis

```
void stoz (int num, COMPLEX s11[], COMPLEX s21[], COMPLEX s12[], COMPLEX
           s22[], COMPLEX z11[], COMPLEX z21[], COMPLEX z12[], COMPLEX
           z22[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
s11	Array containing the S11 parameters.
s21	Array containing the S21 parameters.
s12	Array containing the S12 parameters.
s22	Array containing the S22 parameters.

Argument	Range Restrictions/Description
z11	Return array for Z11 parameters transformed from S parameters.
z21	Return array for Z21 parameters transformed from S parameters.
z12	Return array for Z12 parameters transformed from S parameters.
z22	Return array for Z22 parameters transformed from S parameters.
z0	Characteristic impedance Z0.

If the absolute value of the denominator in the above equation [(1–S11)×(1–S22)–S12×S21] is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the Z parameters.

See Also rfs_z0_r

stoz1

Revision	D.05.10 or later
Control	n.a.

This function transforms one-port S parameters into one-port Z parameters.

The following equation is used for the S-to-Z transformation:

$$Z = Z0 \times \frac{1+S}{1-S}$$

Synopsis

```
void stoz1 (int num, COMPLEX s[], COMPLEX z[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
s	Name of array containing the one-port S parameters.
z	Return array for one-port Z parameters transformed from the S parameters.
z0	Characteristic impedance Z0.

If S is equal to 1, $1\times10^{+99}$ + $(1\times10^{+99})$j is returned to the Z parameters.

See Also rfs_z0_r

sweep_cv

Revision	C.04.00 or later
Control	HSCMU

This function performs the C/G-V measurement by HSCMU according to the sweep (bias source) conditions set by set_cv function, and returns the error corrected measurement values, bias source values, and status and timestamp for each measurement value.

Synopsis

```
int sweep_cv (double range, double capacitance[], double conductance[],
              double bias[]);
```

Argument	Range Restrictions/Description
range	Specify the measurement range by using a capacitance value. 0 for auto-ranging mode or Numeric expression [F]: – (Note 1)
capacitance	Specify a pointer to a double array in which to return the measured capacitance values. If not needed, specify NULL.
conductance	Specify a pointer to a double array in which to return the measured conductance values. If not needed, specify NULL.
bias	Specify a pointer to a double array in which to return the bias source sweep voltage values. If not needed, specify NULL.

Note 1 See the description of *range* parameter in measure_cpg function for details.

- *range*

  If *range* is set to 0, Auto ranging mode is used.

  If *range* is set to a non-zero value, fixed ranging is used. For details, refer to measure_cpg function.

- *capacitance*

  The compensated capacitance data is returned to this array. When the measurement status is abnormal, 9.99999E+9 is returned as the capacitance value.

- *conductance*

  The compensated conductance data is returned to this array, which must be the same size as the *capacitance* array.

  If the measurement status is abnormal, 9.99999E+9 is returned as the conductance value.

- *bias*

  The bias source sweep voltages are returned to this array, which must be the same size as the *capacitance* array.

To get the measurement status and timestamp for the measurement data, use the status_cvf function. If the port_status function is performed after executing the sweep_cv function, the returned value reflects only the results of last sweep point.

sweep_cvf

Revision	C.04.00 or later
Control	HSCMU

This function performs the C/G-V-f measurement by HSCMU according to the sweep (bias source and measurement frequency) conditions set by set_cvf function, and returns the error corrected measurement values.

Synopsis

```
int sweep_cvf (double range, double capacitance[], double conductance[]);
```

Argument	Range Restrictions/Description
range	Specify the capacitance measurement range. (Note 1) Specify 0.0 for Auto range mode or a Numeric expression [F]: real value
capacitance	Specify a pointer to a double array in which to return the measured capacitance values. If not needed, specify NULL.
conductance	Specify a pointer to a double array in which to return the measured conductance values. If not needed, specify NULL.

Note 1 If the revision of your system software is older than C.04.01 with patch P.04.01.2004.0131, the measurement range is a dummy parameter. HSCMU always uses Auto ranging mode in C/G-V-f measurement.

- *range*

  The HSCMU actually performs the impedance measurement and then converts the measured impedance to the capacitance and conductance values.

  If the *range* (capacitance range) other than 0 is specified, the impedance range value (Ir) is calculated from the specified range (Cr) and measurement frequency (fm) as follows:

$$Ir = 1/(2 \times \pi \times |fm| \times |Cr|)$$

And an actual impedance range for impedance measurement is determined from the calculated impedance range value (Ir), test signal level (Vosc) and measurement frequency (fm) as shown in following table:

Calculated *Ir* value	Measurement Frequencies (fm), Test Signal level (Vosc), and Corresponding Impedance Ranges			
	Vosc = 30, 50, 100mV_{rms}		Vosc = 10 mV_{rms}	
	fm ≤ 100 kHz	120 kHz ≤ fm	fm ≤ 100 kHz	120 kHz ≤ fm
0 ≤ *Ir* ≤ 300 Ω	100 Ω	100 Ω	100 Ω	100 Ω
300 Ω < *Ir* ≤ 1 kΩ	300 Ω	300 Ω	100 Ω	100 Ω
1 kΩ < *Ir* ≤ 3 kΩ	1 kΩ	1 kΩ	1 kΩ	1 kΩ
3 kΩ < *Ir* ≤ 10 kΩ	3 kΩ	3 kΩ	1 kΩ	1 kΩ
10 kΩ < *Ir* ≤ 30 kΩ	10 kΩ	10 kΩ	10 kΩ	10 kΩ
30 kΩ < *Ir* ≤ 100 kΩ	30 kΩ	30 kΩ	10 kΩ	10 kΩ
100 kΩ < *Ir* ≤ 300 kΩ	100 kΩ	30 kΩ	100 kΩ	10 kΩ
300 kΩ < *Ir*	300 kΩ	30 kΩ	100 kΩ	10 kΩ

If you specify 0.0 for the *range*, auto-ranging is used, which performs the measurement in range that gives the best resolution.

- *capacitance*, *conductance*

  The compensated capacitance data is returned to the *capacitance* array. When the measurement status is abnormal, 9.99999E+9 is returned as the capacitance value.

  The compensated conductance data is returned to the *conductance* array.

  The size of these array must be equal to Nf × Nv, where Nf is the number of measurement frequencies and Nv is the number of DC bias steps.

  The data is stored as (F0, V0), ... , (F0, Vn), (F1, V0), ... , (Fn, Vn), where (Fn, Vn) is the data for nth measurement frequency and nth DC bias step.

To get the measurement status and timestamp for the measurement data, use the status_cvf function. If the port_status function is performed after executing the sweep_cv function, the returned value reflects only the results of last sweep point.

sweep_cv84

Revision	A.01.00 or later
Control	CMU, (HSCMU)

NOTE

CMU Control TIS function

This function can be used to execute 4080 measurement program for using CMU on the 4080 with HSCMU.

Do not use this function to make new measurement program for using HSCMU. (Use sweep_cv.)

This function performs the C/G-V measurement by HSCMU or CMU according to the sweep (bias source) conditions set by set_cv84 function, and returns the error corrected measurement values and bias source values.

Synopsis

```
int sweep_cv84 (double range, double capacitance[], double conductance[],
                double bias[]);
```

Arguments for HSCMU

Argument	Range Restrictions/Description
range	Specify the measurement range by using a capacitance value. 0 for auto-ranging mode or Numeric expression [F]: See range parameter in measure_cmu84 function.
capacitance	Specify a pointer to a double array in which to return the measured capacitance values. If not needed, specify NULL.
conductance	Specify a pointer to a double array in which to return the measured conductance values. If not needed, specify NULL.
bias	Specify a pointer to a double array in which to return the bias source sweep voltage values. If not needed, specify NULL.

- ***range***

If *range* is set to 0, Auto ranging mode is used.

If *range* is set to a non-zero value, fixed ranging is used. For details, refer to measure_cmu84 function.

- *capacitance*

  The compensated capacitance data is returned to this array. When the measurement status is abnormal, 9.99999E+9 is returned as the capacitance value.

- *conductance*

  The compensated conductance data is returned to this array, which must be the same size as the *capacitance* array.

  If the measurement status is abnormal, 9.99999E+9 is returned as the conductance value.

- *bias*

  The bias source sweep voltages are returned to this array, which must be the same size as the *capacitance* array.

Arguments for CMU

The measurement resolution depends on the *integ_time* set in the set_cmu84 function.

Argument	Range Restrictions/Description
range	Specify the measurement range by using a capacitance value. 0 for auto-ranging mode or Numeric expression [F]: See range parameter in measure_cmu84 function.
capacitance	Specify a pointer to a double array in which to return the measured capacitance values. If not needed, specify NULL.
conductance	Specify a pointer to a double array in which to return the measured conductance values. If not needed, specify NULL.
bias	Specify a pointer to a double array in which to return the bias source sweep voltage values. If not needed, specify NULL.

- *range*

  If *range* is set to 0, Auto ranging mode is used.

  If *range* is set to a non-zero value, limited auto ranging is used. For details, refer to measure_cmu84 function.

- *capacitance*

  The compensated capacitance data is returned to this array. When the measurement status is abnormal, -9999999.99999 is returned as the capacitance value.

- *conductance*

  The compensated conductance data is returned to this array, which must be the same size as the *capacitance* array.

  If the measurement status is abnormal, -9999999.99999 is returned as the conductance value.

- *bias*

  The bias source sweep voltages are returned to this array, which must be the same size as the *capacitance* array.

C/G-V measurement status: If the port_status function is performed after executing the sweep_cv84 function, the returned value reflects only the results of last sweep point.

sweep_iv

Revision	A.01.00 or later
Control	SMU

This function can perform the following sweep type measurements:

- staircase sweep
- synchronous staircase sweep
- pulsed sweep
- staircase sweep with pulsed bias measurements

The sweep_iv function performs staircase sweep, pulsed sweep, *or* staircase sweep with pulsed bias measurements according to measurement parameters set in the following functions:

Measurement	Function Name
Staircase sweep measurements	set_iv
Pulsed sweep measurements	set_piv
Staircase sweep with pulsed bias measurements	set_iv, set_pbias

If set_sync was specified, sweep_iv performs *synchronous* staircase sweep measurements using the primary sweep port (specified by set_iv or set_piv), the synchronous sweep port (specified by set_sync), and pulse bias port (if set_pbias was specified).

For each sweep step, measurements are made at the measurement port specified by sweep_iv. After the sweep is completed, measurement values, primary source values, and synchronous source values for each measurement are returned to arrays.

The rsweep_iv function performs similar sweeps, but in *real-time*, which means a measurement value can be returned after each sweep step.

The sweep_iv function returns all measurement values after the sweep is completed, not after each sweep step.

Synopsis

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

Argument	Range Restrictions/Description	
port	SMU *measure* port address. You can specify a port address directly or specify a pin number that is connected to the port. You can use one of the following macros (or values):	
	port address	You can specify the following macros (or values (Note 1)): SMU1 (20001), SMU2 (20002), SMU3 (20003), SMU4 (20004), SMU5 (20005), SMU6 (20006), SMU7 (20007), SMU8 (20008).
	pin number	1 to 49
mode	Specify whether *port* will perform a voltage or current measurement. Use one of the following macros: V_MEAS (= 1) I_MEAS (= 2)	
range	Specify the measurement range. 0 for auto-ranging mode or –0.1 to 0.1 or –1 to 1 (HPSMU) for current measurement or –100 to 100 or –200 to 200 (HPSMU) for voltage measurement	
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 these values are not necessary, specify NULL. If set_sync function was *not* specified, this argument must be NULL.	

Note 1 For program readability and future compatibility, we recommend not to use this value directly but use macro or PORT function instead.

- *port*

  To specify the SMU measurement port for sweep_iv, use either the *port address* of SMU or *pin number* of any SWM pin connected to the SMU port.

- *mode* and *range*

  If *mode* is V_MEAS (voltage measurement), refer to the measure_v function for *range*.

  If *mode* is I_MEAS (current measurement), refer to the measure_i function for *range*.

- *measure*

  The present measurement value is returned to *measure*.

- *source* array and *sync* array

  The sweep source values determined by set_iv or set_piv are returned to *source* array.

  The synchronous sweep source values determined by set_sync are returned to *sync* array.

Use the status_miv function to get the statuses of each measurement points. The port_status function returns the status of the last measurement point only.

Example

```
int err;
double offset, ratio, comp2;
 :
err=set_sync(32,POS_SYNC,offset,ratio,comp2,0.0);
err=sweep_iv(32,I_MEAS,0.0,meas1,swp1,swp2);
```

See Also

- rsweep_iv
- rsweep_miv
- set_iv
- set_piv
- set_pbias
- set_sync (for sweep measurements)
- sweep_miv
- status_miv

sweep_miv

Revision	A.01.00 or later
Control	SMU

This function can perform the following *multichannel* sweep type measurements:

- staircase sweep
- synchronous staircase sweep

This function *cannot* set the measurement mode, so you should use force_i or force_v functions to set desired measurement mode for each port before sweep_miv function.

Note that this function cannot be used for pulsed sweep or staircase sweep with pulsed bias measurements.

- For Staircase Sweep Measurements

  This function performs *multichannel* staircase sweep measurements according to the measurement parameters set in the set_iv function.

- For Synchronous Staircase Sweep Measurements

  This function performs *multichannel synchronous* staircase sweep measurements according to the measurement parameters set in the set_iv and set_sync functions.

Synopsis

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

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

Argument	Range Restrictions/Description
n	Specify how many ports (1 to 8) will perform measurement. This value determines the size of the following arrays: *ports*, *ranges*, and the primary index of *meas_vals*.
ports	Specify a pointer to an integer array that determines the measurement ports. This array must have *n* elements. The array should contain the SMU port addresses or SWM pin numbers that are connected to ports.
ranges	Specify a pointer to double array that contains the measurement range for each port. This array must have *n* elements. For Auto range mode, specify 0.
meas_vals	Specify a pointer to a double array in which to return the measurement values.
source	Specify a pointer to a double array in which to return the sweep source data (which is determined by set_iv). If you don't need this data, 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). If you don't need this data, specify NULL. If you don't use the set_sync function, this argument must be NULL.

- *ports* array

  To specify the measurement ports for sweep_miv, you can specify port addresses of SMUs or pin numbers of any measurement pin connected to the SMU ports. This array should have *n* elements.

- *ranges* array

  You should assign range values to *ranges* array elements that correspond to each *ports* array element. This array should have *n* elements.

  For voltage measurement, refer to the measure_v function for the measurement range.

  For current measurement, refer to the measure_i function for the measurement range.

- *meas_vals* array

  The measurement values for each measurement port are returned to the *meas_vals* array. So, the size of *meas_vals* array should be *n*×*number* for a single sweep, or *n*×*2*×*number* for a double sweep. Where, *number* is the number of steps specified in set_iv function.

- *source* and *sync* arrays

  The source values output by the sweep source (set by set_iv) are returned to the *source* array.

  Also, for synchronous sweep measurement, the values output by the synchronous source (set by set_sync) are returned to the *sync* array.

  The size of *source* and *sync* arrays should be *number* for a single sweep, or *2*×*number* for a double sweep. Where, *number* is specified in set_iv function.

Use the status_miv function to get the measurement statuses.

Offline Mode In offline mode, this function returns values as follows:

- Measurement values are calculated and returned to *meas_vals* array elements as follows:

$$\text{nth measurement value} = \frac{basevalue}{\text{number of steps} - 1} \times (n-1)$$

  Where: number of *steps* is set by the set_iv function, and *basevalue* is determined as follows:

Table 2-67 For current measurements

Meas. Range	*basevalue*	Polarity
0	I compliance	Same as SMU *output value*
≠ 0	Range value	

Table 2-68 **For voltage measurements**

Meas. Range	*basevalue*	Polarity
0	V compliance	Same as SMU *output value*
≠ 0	Range value	

- *source* array

  Sweep source values determined by set_iv are returned to the *sources* array.

- *sync* array

  Synchronous sweep source values determined by set_sync are returned to the *sync* array.

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 (for sweep measurements)
- sweep_iv
- rsweep_iv
- rsweep_miv
- status_miv

sweep_spa

Revision	D.05.16 or later
Control	SPA

This function performs the measurement defined by set_spa and returns the amplitude vs. frequency characteristics to arrays.

Synopsis

```
#include "/opt/hp4070/include/spa.h"
   int sweep_spa (double amp[], double freq[]);
```

Argument	Range Restrictions/Description
amp	Array of size 401 which is used to return the amplitude values.
freq	Array of size 401 which is used to return the frequency values.

- *amp*

  Return array for amplitude values of 401 points.

- *freq*

  Return array for frequency values of 401 points.

See Also

- set_spa

sweep_zf

Revision	C.04.00 or later
Control	HSCMU

This function performs the Z/θ--f measurement by HSCMU according to the sweep (measurement frequencies) conditions set by set_zf function, and returns the error corrected measurement values, measurement frequencies, and measurement status for each measurement value.

Synopsis

```
int sweep_zf (double range, double impedance[], double phase[], double
              freq[], double status[]);
```

Argument	Range Restrictions/Description
range	Specify the measurement range by using a impedance value. 0 for auto-ranging mode or Numeric expression [Ω]: – (Note 1)
impedance	Specify a pointer to a double array in which to return the measured impedance values.
phase	Specify a pointer to a double array in which to return the measured phase values. If not needed, specify NULL.
freq	Specify a pointer to a double array in which to store the measurement frequencies. If not needed, specify NULL.
status	Specify a pointer to a double array in which to store the measurement status. If not needed, specify NULL.

Note 1 See the description of *range* parameter in measure_ztr function for details.

- *range*

  If *range* is set to 0, Auto ranging mode is used.

If *range* is set to a non-zero value, fixed ranging is used. For details, refer to measure_ztr function.

- *impedance*

  The compensated impedance data is returned to this array. When the measurement status is abnormal, 9.99999E+9 is returned as the impedance value.

  The size of this array must be equal to or larger than the number of measurement frequencies set by the set_zf function. You can use the get_freq_step function to know the actual number of measurement frequencies set by the set_zf function.

- *phase*

  The compensated phase data is returned to this array, which must be the same size as the *impedance* array.

  If the measurement status is abnormal, 9.99999E+9 is returned as the phase value.

- *freq*

  The measurement frequencies set by the set_zf function are returned to this array, which must be the same size as the *impedance* array.

- *status*

  The status of measurement data are returned to this array.

  The meaning of the status value is as follows:

  0: Normal

  1: Analog bridge is unbalanced

  2: DC bias source is overloaded

  3: A/D converter is not working

  4: Sweep measurement has been aborted

  5: Measurement overflow

  If two or more conditions occur during a measurement, the priority of the returned numerical code is 1>2>5>3>4.

switch_sr

Revision	B.02.00 or later
Control	SWM, PGU, SPGU

This function controls the open/close switch or multiplexer switch of the pulse switch port.

Synopsis

```
int switch_sr (int port, int mode);
```

Argument	Range Restrictions/Description
port	Port address of PSO or PSC port. 20261 to 20269
mode	Integer value to select the connection mode of open/close switch or multiplexer switch. 0 to 2

This function controls the switch of pulse switch port and the output switch of the corresponding pulse generator (PG).

Before this function, the set_sr_pg function must set the switching method to "independent switching" mode.

mode depends on type of specified port as follows:

- If the specified port is open/close switch port, which are PSO2(20262), PSO3(20263), PSO4(20264), or PSO5(20265):

mode	Pulse Switch
0	Open
1	Close

- If the specified port is the multiplexer switch port, which are PSO1(20261), PSO6(20266), or PSO7(20267):

mode	Pulse Switch
0	Both PSI ports do not connect to the specified port.
1	PSIx1 port connects to the specified port.
2	PSIx2 port connects to the specified port.

If the specified port is PSC1(20268), this function controls the switch of PSO1, PSO2, PSO3, and PSO4.

If the specified port is PSC2(20269), this function controls the switch of PSO5, PSO6, and PSO7.

switch_sr_wait

Revision	C.04.00 or later
Control	SWM, PGU, SPGU

This function controls the open/close switch or multiplexer switch of the pulse switch port.

Synopsis

```
int switch_sr_wait (int port, int mode, double wait);
```

Argument	Range Restrictions/Description
port	Port address of PSO or PSC port. 20261 to 20269
mode	Integer value to select the connection mode of open/close switch or multiplexer switch. 0 to 2
wait	Wait time between turning on the output switch of the pulse generator and the open/close operation of pulse switch. 0.01 to 2.0 [s], resolution: 10 ms

This function controls the switch of pulse switch port and the output switch of the corresponding pulse generator (PG) same as switch_sr. By using this function, you can specify the wait time between turning on the output switch of the pulse generator and the open or close operation of pulse switch.

Before this function, the set_sr_pg function must set the switching method to "independent switching" mode.

mode depends on type of specified port as follows:

- If the specified port is open/close switch port, which are PSO2(20262), PSO3(20263), PSO4(20264), or PSO5(20265):

mode	Pulse Switch
0	Open
1	Close

- If the specified port is the multiplexer switch port, which are PSO1(20261), PSO6(20266), or PSO7(20267):

mode	Pulse Switch
0	Both PSI ports do not connect to the specified port.
1	PSIx1 port connects to the specified port.
2	PSIx2 port connects to the specified port.

If the specified port is PSC1(20268), this function controls the switch of PSO1, PSO2, PSO3, and PSO4.

If the specified port is PSC2(20269), this function controls the switch of PSO5, PSO6, and PSO7.

sync_th

Revision	A.01.00 or later
Control	SWM

The system controller sends testhead control statements (for SWM and SMUs) to a FIFO buffer, and the statements in FIFO are executed by testhead CPU.

So, a synchronization problem can occur if your measurement program has statements for testhead control *and* for external equipment you installed except CMU and DVM.

The sync_th function pauses program execution until all previous statements (that were sent to testhead CPU) are completed by testhead CPU.

If this function is not executed, the system controller may start to execute the next statements of the measurement program before the testhead control statements are executed by testhead CPU.

So, this function is necessary to keep synchronization between testhead CPU execution and measurement program execution.

Synopsis

```
int sync_th (double wait_time);
```

Argument	Range Restrictions/Description
wait_time	0 to 655.53 [s]

- *wait time*

  If *wait_time* other than 0 is specified, this function waits the additional *wait_time* after testhead CPU has executed all testhead commands.

T

This section describes the C functions that begin with T.

tap_port

Revision	A.01.00 or later
Control	SWM

This function connects or disconnects SMU1 port to AUX1 port, or SMU2 port to AUX2 port.

If there is an Active unit when the tap_port function executes, all units (except SMUs that are set to Standby mode) are set to Zero Output state.

Active unit means a unit that is currently forcing output. For Standby mode, refer to stand_by_port.

If a unit is set to Zero Output state by the tap_port function and is still connected to measurement pins or chuck, the unit is returned to its previous Active state when next force_i, force_v, or force_meas function is executed.

Synopsis

```
int tap_port (int on, int port);
```

Argument	Range Restrictions/Description	
on	Connect or disconnect the specified SMU and AUX port. You can use one of the following macros (or values):	
	Connect	ON (=1)
	Disconnect	OFF (=0)

Argument	Range Restrictions/Description	
port	Specify port address of SMU1 or AUX1, or SMU2 or AUX2. You can specify a port address directly or specify a pin number that is connected to the port:	
	port address	You can specify the following macros (or values): SMU1 (20001) SMU2 (20002) 20101: this is AUX1 port 20102: this is AUX2 port
	pin number	1 to 49

- *on*

  Specify whether you want to connect or disconnect SMU and AUX determined by *port*.

- *port*

  If you specify SMU1 or AUX1 for *port*, SMU1 is connected to (or disconnected from) AUX1.
  or
  If you specify SMU2 or AUX2 for *port*, SMU2 is connected to (or disconnected from) AUX2.

Example

```
tap_port(ON,SMU1);
```

W

This section describes the C functions that begin with W.

wait_event

Revision	A.01.00 or later
Control	n.a.

This function pauses the measurement program, and waits for an event to occur or for an event to finish.

Synopsis

```
int wait_event (int *event_on, int *event_off, double timeout);
```

Argument	Range Restrictions/Description
event_on, *event_off*	*event_on* specifies the event to wait to occur. *event_off* specifies the event to wait to complete. To specify a event, you can use the following macros (or decimal values): EVENT_HIGH_VOLTAGE (=1) EVENT_PWF (=2) EVENT_OVER_IV (=4) EVENT_FIXOPEN (=8) EVENT_INTLOPEN (=16)
timeout	Dummy parameter.

- *event_on*

  The wait_event function waits until the specified event occurs, then goes to next statement.

- *event_off*

  The wait_event function waits until the specified event is finished, then goes to next statement.

  If the specified *event_off* is not occurring when this function is executed, this function immediately returns 0 to *event_off* and goes to next statement.

You can specify the following events:

Event	Event Value	Description
EVENT_HIGH_VOLTAGE	1	SMU outputs voltage 40 V or more.
EVENT_PWF	2	Testhead power failed.
EVENT_OVER_IV	4	Output voltage or current of SMU is over its limit.
EVENT_FIXOPEN	8	Fixture is open.
EVENT_INTLOPEN	16	Interlock is open.

NOTE

Multiple events

You can specify multiple events by adding the event values. For example, to specify high voltage or power fail, specify 3.

For multiple events specified by *event_on*, if one of the events occurs, the event value of that event is returned to *event_on*.

For multiple events specified by *event_off*, when that event is finished, the event value of that event is returned to *event_off*.

Example

```
int event_on; event_on=EVENT_HIGH_VOLTAGE; ...
wait_event(&event_on,NULL,0);
```

wait_th

Revision	A.01.00 or later
Control	n.a.

This function sends a wait command to testhead CPU.

This causes the testhead CPU to wait the specified *wait_time* after previous command, then execute next command.

Synopsis

```
int wait_th (double wait_time);
```

Argument	Range Restrictions/Description
wait_time	0 to 655.35 [s] Resolution: 100 μs

Example

```
wait_th(1);
```

Y

This section describes the C functions that begin with Y.

ytoh

Revision	D.05.10 or later
Control	n.a.

This function transforms Y parameters into H parameters.

The following equation is used for the Y-to-H transformation:

$$\begin{bmatrix} H11 & H12 \\ H21 & H22 \end{bmatrix} = \begin{bmatrix} \frac{1}{Y11} & \frac{-Y12}{Y11} \\ \frac{Y21}{Y11} & \frac{Y11 \times Y22 - Y12 \times Y21}{-Y11} \end{bmatrix}$$

Synopsis

```
void ytoh (int num, COMPLEX y11[], COMPLEX y21[], COMPLEX y12[], COMPLEX
           y22[], COMPLEX h11[], COMPLEX h21[], COMPLEX h12[], COMPLEX
           h22[]);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
y11	Array containing the Y11 parameters.
y21	Array containing the Y21 parameters.
y12	Array containing the Y12 parameters.
y22	Array containing the Y22 parameters.
h11	Return array for H11 parameters transformed from Y parameters.

Argument	Range Restrictions/Description
h21	Return array for H21 parameters transformed from Y parameters.
h12	Return array for H12 parameters transformed from Y parameters.
h22	Return array for H22 parameters transformed from Y parameters.

If the absolute value of the denominator in the above equation [Y11] is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the H parameters.

ytos

Revision	D.05.10 or later
Control	n.a.

This function transforms Y parameters into S parameters.

The following equation is used for the Y-to-S transformation:

$$\begin{bmatrix} y11 & y12 \\ y21 & y22 \end{bmatrix} = Z0 \times \begin{bmatrix} Y11 & Y12 \\ Y21 & Y22 \end{bmatrix}$$

$$\begin{bmatrix} S11 & S12 \\ S21 & S22 \end{bmatrix} = \begin{bmatrix} \frac{(1-y11)(1+y22)+y12\times y21}{(1+y11)(1+y22)-y12\times y21} & \frac{-2\times y12}{(1+y11)(1+y22)-y12\times y21} \\ \frac{-2\times y21}{(1+y11)(1+y22)-y12\times y21} & \frac{(1+y11)(1-y22)+y12\times y21}{(1+y11)(1+y22)-y12\times y21} \end{bmatrix}$$

Synopsis

```
void ytos (int num, COMPLEX y11[], COMPLEX y21[], COMPLEX y12[], COMPLEX
           y22[], COMPLEX s11[], COMPLEX s21[], COMPLEX s12[], COMPLEX
           s22[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
y11	Array containing the Y11 parameters.
y21	Array containing the Y21 parameters.
y12	Array containing the Y12 parameters.
y22	Array containing the Y22 parameters.
s11	Return array for S11 parameters transformed from Y parameters.
s21	Return array for S21 parameters transformed from Y parameters.

Argument	Range Restrictions/Description
s12	Return array for S12 parameters transformed from Y parameters.
s22	Return array for S22 parameters transformed from Y parameters.
z0	Characteristic impedance Z0.

If the absolute value of the denominator in the above equation [(1+y11)×(1+y22)–y12×y21] is equal to 0 (zero), $1\times10^{+99}$ + $(1\times10^{+99})$j is returned to the S parameters.

See Also rfs_z0_r

ytos1

Revision	D.05.10 or later
Control	n.a.

This function transforms one-port Y parameters into one-port S parameters.

The following equation is used for the Y-to-S transformation:

$$S = \frac{1 - Z0 \times Y}{1 + Z0 \times Y}$$

Synopsis

```
void ytos1 (int num, COMPLEX y[], COMPLEX s[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
y	Name of array containing the one-port Y parameters.
s	Return array for one-port S parameters transformed from the Y parameters.
z0	Characteristic impedance Z0.

If Z0×Y is equal to –1, $1 \times 10^{+99} + (1 \times 10^{+99})j$ is returned to the S parameters.

See Also rfs_z0_r

ytoz

Revision	D.05.10 or later
Control	n.a.

This function transforms Y parameters into Z parameters.

The following equation is used for the Y-to-Z transformation:

$$\begin{bmatrix} Z11 & Z12 \\ Z21 & Z22 \end{bmatrix} = \begin{bmatrix} \frac{Y22}{Y11\times Y22 - Y12\times Y21} & \frac{-Y12}{Y11\times Y22 - Y12\times Y21} \\ \frac{-Y21}{Y11\times Y22 - Y12\times Y21} & \frac{-Y11}{Y11\times Y22 - Y12\times Y21} \end{bmatrix}$$

Synopsis

```
void ytoz (int num, COMPLEX y11[], COMPLEX y21[], COMPLEX y12[], COMPLEX
           y22[], COMPLEX z11[], COMPLEX z21[], COMPLEX z12[], COMPLEX
           z22[]);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
y11	Array containing the Y11 parameters.
y21	Array containing the Y21 parameters.
y12	Array containing the Y12 parameters.
y22	Array containing the Y22 parameters.
z11	Return array for Z11 parameters transformed from Y parameters.
z21	Return array for Z21 parameters transformed from Y parameters.
z12	Return array for Z12 parameters transformed from Y parameters.
z22	Return array for Z22 parameters transformed from Y parameters.

If the absolute value of the denominator in the above equation [Y11×Y22–Y12×Y21] is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to Z parameters.

ytoz1

Revision	D.05.10 or later
Control	n.a.

This function transforms one-port Y parameters into one-port Z parameters.

The following equation is used for the Y-to-Z transformation:

$$Z = \frac{1}{Y}$$

Synopsis

```
void ytoz1 (int num, COMPLEX y[], COMPLEX z[]);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
y	Name of array containing the one-port Y parameters.
z	Return array for one-port Z parameters transformed from the Y parameters.

If Y is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the Z parameters.

Z

This section describes the C functions that begin with Z.

ztoh

Revision	D.05.10 or later
Control	n.a.

This function transforms Z parameters into H parameters.

The following equation is used for the Z-to-H transformation:

$$\begin{bmatrix} H11 & H12 \\ H21 & H22 \end{bmatrix} = \begin{bmatrix} \frac{Z11 \times Z22 - Z12 \times Z21}{Z22} & \frac{Z12}{Z22} \\ \frac{-Z21}{Z22} & \frac{1}{Z22} \end{bmatrix}$$

Synopsis

```
void ztoh (int num, COMPLEX z11[], COMPLEX z21[], COMPLEX z12[], COMPLEX
           z22[], COMPLEX h11[], COMPLEX h21[], COMPLEX h12[], COMPLEX
           h22[]);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
z11	Array containing the Z11 parameters.
z21	Array containing the Z21 parameters.
z12	Array containing the Z12 parameters.
z22	Array containing the Z22 parameters.

Argument	Range Restrictions/Description
h11	Return array for H11 parameters transformed from Z parameters.
h21	Return array for H21 parameters transformed from Z parameters.
h12	Return array for H12 parameters transformed from Z parameters.
h22	Return array for H22 parameters transformed from Z parameters.

If the absolute value of the denominator in the above equation [Z22] is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the H parameters.

ztos

Revision	D.05.10 or later
Control	n.a.

This function transforms Z parameters into S parameters.

The following equation is used for the Z-to-S transformation:

$$\begin{bmatrix} z11 & z12 \\ z21 & z22 \end{bmatrix} = \frac{1}{Z0} \times \begin{bmatrix} Z11 & Z12 \\ Z21 & Z22 \end{bmatrix}$$

$$\begin{bmatrix} S11 & S12 \\ S21 & S22 \end{bmatrix} = \begin{bmatrix} \frac{(z11-1)(z22+1)-z12\times z21}{(z11+1)(z22+1)-z12\times z21} & \frac{2\times z12}{(z11+1)(z22+1)-z12\times z21} \\ \frac{2\times z21}{(z11+1)(z22+1)-z12\times z21} & \frac{(z11-1)(z22+1)-z12\times z21}{(z11+1)(z22+1)-z12\times z21} \end{bmatrix}$$

Synopsis

```
void ztos (int num, COMPLEX z11[], COMPLEX z21[], COMPLEX z12[], COMPLEX
           z22[], COMPLEX s11[], COMPLEX s21[], COMPLEX s12[], COMPLEX
           s22[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
z11	Array containing the Z11 parameters.
z21	Array containing the Z21 parameters.
z12	Array containing the Z12 parameters.
z22	Array containing the Z22 parameters.
s11	Return array for S11 parameters transformed from Z parameters.
s21	Return array for S21 parameters transformed from Z parameters.

Argument	Range Restrictions/Description
s12	Return array for S12 parameters transformed from Z parameters.
s22	Return array for S22 parameters transformed from Z parameters.
z0	Characteristic impedance Z0.

If the absolute value of the denominator in the above equation [(z11+1)(z22+1)–z12×z21] is equal to 0 (zero) or if z0 is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the S parameters.

See Also rfs_z0_r

ztos1

Revision	D.05.10 or later
Control	n.a.

This function transforms one-port Z parameters into one-port S parameters.

The following equation is used for the Z-to-S transformation:

$$S = \frac{Z - Z0}{Z + Z0}$$

Synopsis

```
void ztos1 (int num, COMPLEX z[], COMPLEX s[], double z0);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
z	Name of array containing the one-port Z parameters.
s	Return array for one-port S parameters transformed from the Z parameters.
z0	Characteristic impedance Z0.

If (Z+Z0) is equal to 0 (zero), $1 \times 10^{+99} + (1 \times 10^{+99})j$ is returned to the S parameters.

See Also rfs_z0_r

ztoy

Revision	D.05.10 or later
Control	n.a.

This function transforms Z parameters into Y parameters.

The following equation is used for the Z-to-Y transformation:

$$\begin{bmatrix} Y11 & Y12 \\ Y21 & Y22 \end{bmatrix} = \begin{bmatrix} \frac{Z22}{Z11\times Z22 - Z12\times Z21} & \frac{-Z12}{Z11\times Z22 - Z12\times Z21} \\ \frac{-Z21}{Z11\times Z22 - Z12\times Z21} & \frac{-Z11}{Z11\times Z22 - Z12\times Z21} \end{bmatrix}$$

Synopsis

```
void ztoy (int num, COMPLEX z11[], COMPLEX z21[], COMPLEX z12[], COMPLEX
           z22[], COMPLEX y11[], COMPLEX y21[], COMPLEX y12[], COMPLEX
           y22[]);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
z11	Array containing the Z11 parameters.
z21	Array containing the Z21 parameters.
z12	Array containing the Z12 parameters.
z22	Array containing the Z22 parameters.
y11	Return array for Y11 parameters transformed from Z parameters.
y21	Return array for Y21 parameters transformed from Z parameters.
y12	Return array for Y12 parameters transformed from Z parameters.
y22	Return array for Y22 parameters transformed from Z parameters.

If the absolute value of the denominator in the above equation [Z11×Z22–Z12×Z21] is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the Y parameters.

ztoy1

Revision	D.05.10 or later
Control	n.a.

This function transforms one-port Z parameters into one-port Y parameters.

The following equation is used for the Z-to-Y transformation:

$$Y = \frac{1}{Z}$$

Synopsis

```
void ztoy1 (int num, COMPLEX z[], COMPLEX y[]);
```

Argument	Range Restrictions/Description
num	Number of valid data for the array.
z	Name of array containing the one-port Z parameters.
y	Return array for one-port Y parameters transformed from the Z parameters.

If Z is equal to 0 (zero), $1\times10^{+99}$ + ($1\times10^{+99}$)j is returned to the Y parameters.

3 Error Messages

This chapter explains how to read error codes, if you encounter some problem.

1-xxxxx

- **1-10001 System software inconsistency discovered.**
  1-10002 System software inconsistency discovered in session login mechanism.
  1-10003 System software inconsistency discovered in session logout mechanism.

  Unknown error occurred during the 4080 system software execution, or login/logout session. Cannot continue the session. Execute hp4070 -logout, then execute hp4070 -login again.

- **1-10004 Cannot allocate memory.**

  Could not allocate enough memory. Execute hp4070 -logout, then execute hp4070 -login again.

- **1-10006 Cannot set socket buffer size.**
  1-10007 Failed to call socket().
  1-10008 Failed to call bind().
  1-10009 Failed to call listen().
  1-10010 Failed to call symlink().
  1-10011 Failed to call accept().
  1-10012 Failed to call connect().
  1-10013 Failed to call shmget().
  1-10014 Failed to call shmat().

  Unknown error occurred in the 4080 system software. Execute hp4070 -logout, then execute hp4070 -login again. Or reboot the controller.

- **1-10015 Cannot open hardware configuration file(in /etc/opt/hp4070/config/).**
  1-10016 Cannot fstat() hardware configuration file(in /etc/opt/hp4070/config/).
  1-10017 Cannot mmap() hardware configuration file(in /etc/opt/hp4070/config/).
  1-10018 Cannot read() hardware configuration file(in /etc/opt/hp4070/config/).
  1-10019 Cannot unmmap() hardware configuration file(in /etc/opt/hp4070/config/).

Could not open the configuration file (/etc/opt/hp4070/config/1). Check the configuration file. If problem is not found in the data, permission and so on, re-install the 4080 system software.

- **1-10020 Cannot allocate event handler table.**

  Too many test programs are running. Or too many events are registered by wait_event function. Stop unnecessary test programs.

- **1-10021 Cannot read Optical Interface Special Data Register 0.**
  1-10022 Cannot ioctl() Optical Interface Card.
  1-10023 Cannot ioctl() Optical Interface Card.
  1-10024 Cannot ioctl() Optical Interface Card.
  1-10025 Cannot read Optical Interface Special Data Register N.
  1-10029 Cannot ioctl() Optical Interface Card.

  Error occurred when accessing the optical interface card. Execute hp4070 -logout, then execute hp4070 -login again. Or the optical interface card may be defective.

- **1-10026 Power Fail occurred. TIS Daemon process is going to die.**

  Testhead was turned off. Turn the testhead on. And execute hp4070 -login.

- **1-10027 Another online TIS Daemon is running.**

  Another online test session is already running. Only one online TIS Daemon can be running.

- **1-10028 Cannot open optical interface device file.**

  Could not open the optical interface device file. Check the following device files. If problem is not found in the data, permission and so on, re-install the 4080 system software. The permission of the file must be crw-rw-rw-.

  - /dev/hp4070/opt_sN
  - /dev/hp4070/opt_sNf0
  - /dev/hp4070/opt_sNf1
  - /dev/hp4070/opt_sNf2
  - /dev/hp4070/opt_sNs0
  - /dev/hp4070/opt_sNs1
  - /dev/hp4070/opt_sNs2

where, *N* is the ISA slot number (1, 2, 3, or 4) where the optical interface card is installed.

- **1-10031 Cannot fork expmon4070 (errno=*N*).**

  Could not execute expmon4070. Check /opt/hp4070/lbin/expmon4070. The permission must be -r-xr-xr-x. If too many processes are running, kill unnecessary processes.

  where, *N* is the error number returned by the kernel or system call.

- **1-10032 OptCard selftest(register r/w test) error.**
  1-10033 OptCard selftest(FIFO memory r/w test) error.
  1-10034 OptCard selftest(communication test) error.

  Failed on the self-test of the optical interface card installed in the system controller. Replace the optical interface card.

- **1-10035 Test Head is not in normal mode. Cannot continue TIS Daemon.**

  Unknown problem occurred in the testhead. Turn off the power of the testhead and on again. Then restart the TIS server again.

- **1-20001 Client Table full. Please quit some applications that use TIS library.**

  Too many test programs or test algorithms are running. Stop unnecessary programs or algorithms.

- **1-20003 Error in iclear() on accessing GPIB device.**
  1-20004 Error in itimeout() on accessing GPIB device.
  1-20005 Error in igpibllo() on accessing GPIB device.
  1-20006 Error in iprintf() on accessing GPIB device.
  1-20007 Error in iopen() on accessing GPIB device.
  1-20008 Error in iwrite() on accessing GPIB device.
  1-20009 Error in iread() on accessing GPIB device.
  1-20022 Error in igpibbusstatus().

  Error occurred in accessing GPIB interface. Check the GPIB interface card or the system instruments.

- **1-20010 Error in ioctl() on accessing optical interface.**
  1-20011 Error in write() on accessing optical interface.
  1-20012 Error in read() on accessing optical interface.
  1-20014 Error in select() on accessing optical interface (errno=*N*).
  1-20015 Error in select() on accessing optical interface.

  Error occurred in accessing the optical interface. Check the optical interface card.

where, *N* is the error number returned by the kernel or system call.

- **1-20013 Optical Interface Timeout.**

  Time out occurred when accessing the testhead. Check the test program. Or turn off the testhead.

- **1-20020 Cannot load C compensation data file.**

  Could not read the capacitance compensation data. Check the data and permission of the /etc/opt/hp4070/ccdata file and /etc/opt/hp4070/ccdata0 file.

- **1-20021 C compensation data is not loaded.**

  TIS function needs the capacitance compensation data. Related to error No. 1-20020. Solve error No. 1-20020.

- **1-20085 CMU GPIB error occurred.**

  GPIB error occurred during communication with the CMU. Check the GPIB interface card, GPIB cable, and CMU.

- **1-20086 DVM GPIB error occurred.**

  GPIB error occurred during communication with the DVM. Check the GPIB interface card, GPIB cable, and DVM.

- **1-20088 VNA GPIB error occurred.**

  GPIB error occurred during communication with the VNA. Check the GPIB interface card, GPIB cable, and VNA.

- **1-20089 VNA-DCBias GPIB error occurred.**

  GPIB error occurred during communication with the DC Bias Source for the VNA. Check the GPIB interface card, GPIB cable, and DC Bias Source.

- **1-20090 SPG GPIB error occurred.**

  GPIB error occurred during communication with the SPGU. Check the GPIB interface card, GPIB cable, and SPGU.

- **1-20092 SPGU detected emergency status while forcing pulse.**

  SPGU detected its emergency status while forcing pulses.

- **1-20093 SPGU detected over current, over voltage, or high temperature status while forcing pulse.**

  SPGU detected its overcurrent status, overvoltage status, or high temperature condition for operation while forcing pulses.

- **1-30001 Cannot make connection because no session is logged in now.**

  No session is logged in now. Execute hp4070 -login.

- **1-30002 Cannot make connection because of wrong session id.**

  Wrong session ID was specified. Use correct session ID, or log out of unnecessary session.

- **1-30003 Request was refused because the TIS Daemon is not in the normal state. Current state is *N*.**

  where, *N* (=1 to 6) means the following status.

  - 1 : Testhead power fail occurred.
  - 2 : Testhead emergency (Over Current or Over voltage) occurred.
  - 3 : Test fixture is open.
  - 4 : Interlock is open.
  - 5 : Testhead firmware is abnormal. Replace Testhead CPU board.
  - 6 : Another client program (test algorithm, etc.) is in abort process.

  This error message appears because of the above status. Solve the problem corresponding to the status.

- **1-30004 Optical Interface timeout.**

  Time out occurred during communication with the testhead. Check the optical interface card and the optical fiber cables.

- **1-30005 GPIB timeout.**

  Time out occurred during communication with the system instruments. Check the GPIB interface card, GPIB cable, and the system instruments.

- **1-30006 Hard ware Exception occurred and cannot accept any request.**

  Sub-message for error No. 1-30003.

- **1-30007 A client is in aborting sequence.**

  Sub-message for error No. 1-30003.

- **1-30008 SICL error occurred: *N*.**

  Error occurred during communication with the system instruments. Check the GPIB interface card, GPIB cable, and the system instruments.

- **1-30009 Cannot make connection. Only one Debugging Panel may run at a time.**

  Tried to open a second Interactive Debugging Panel in same session. You can open only one debugging panel at a time in same session.

- **1-30010 Cannot make connection. Another Diagnostics is running.**

  Only one Diagnostics program can run at a time. Use the Diagnostics program that is already running. Or stop the program that is already running to newly run the Diagnostics program.

- **1-30011 Cannot make connection. Another Performance Verification is running.**

  Only one Performance Verification (PV) program can run at a time. Use the program that is already running. Or stop the program that is already running to newly run the PV program.

- **1-30012 Request was refused because Diagnostics or PV is running.**

  Test program using TIS cannot be executed when the Diagnostics program or the Performance Verification (PV) program is running. Stop the Diagnostics program or the PV program to run the test program using TIS.

- **1-30013 TIS command cannot be executed because the testhead is in power failure.**

  TIS function cannot be executed because a power failure was detected in the testhead. Turn on the power to the testhead again, then restart the TIS server.

- **1-30014 TIS command cannot be executed because over voltage or current is forced to SMUs.**

  The TIS function cannot be executed because over voltage or over current is being forced to the SMUs from external instruments or SMUs.

- **1-30015 TIS command cannot be executed because the fixture is open.**

  The TIS function cannot be executed because the test fixture lid is open. Close the lid of test fixture.

- **1-30016 TIS command cannot be executed because the interlock is open.**

  The TIS function cannot be executed because the interlock circuit is open. Close the interlock circuit.

- **1-30017 TIS command cannot be executed because the testhead cannot work properly.**

  The TIS function cannot be executed because the testhead cannot work proerly. There is something wrong with the optical interface communication, or something wrrong with the testhead. Contact your nearest Keysight Technologies sales office if the system cannot be restarted.

- **1-30018 TIS command cannot be executed because the system is running an internal process.**

  The TIS function cannot be executed because the system is running an internal process.

- **1-30019 TIS command cannot be executed because an abnormal condition exists.**

  The TIS function cannot be executed because an abnormal condition in the system. Remove the abnormal condition before executing the TIS function again.

2-xxxxx

- **2-10002 Cannot open file. filename:** *File*

  Could not open the file specified by *File*. Confirm the file name and permission of the file.

- **2-10004 An illegal format data exists. filename:** *File*, **data:** *Data*

  Tried to open file with invalid data format. Change the data format of the data *Data* in the file *File*.

- **2-10008 An improper option is specified. option:** *Option*

  Improper value (*Option*) was specified for the option. Specify the option properly.

- **2-10009 An improper argument is specified. option:** *Option*, **argument:** *Argument*

  Argument value (*Argument*) invalid for the option (*Option*) was specified. Specify the argument properly.

- **2-10014 Cannot execute command. command path:** *Path*

  Could not execute the command path *Path*. Re-install the 4080 system software.

- **2-10019 Cannot get character input.**

  Could not read the character input. Maximum 1024 characters.

- **2-11001 Failed to write the test selection file. Check permission.**

  Un-specifiable error occurred in write operation of the test selection file. Check the permission of the file.

- **2-13002 An error occurred in TIS.**

  Error occurred in TIS. Refer to the additional messages.

- **2-14001 Cannot open Optical Interface Card device file.**

  Could not open the device file for the optical interface card. Check the configuration file (/etc/opt/hp4070/config/1) and specify the proper slot number for the optical interface card. Or kill the unnecessary process that opened the optical interface card device file.

- **2-14003 An error occurred in Optical Interface Card Loopback test.**

  Optical interface card failed the loopback test. Check the optical fiber cables or the optical interface card.

- **2-21001 No test is selected. Please select at least one test.**

  No test is selected in the "Test Selection" field. Select at least one test.

- **2-21002 No SMU is selected. Please select at least one SMU.**

  No SMU is selected in the "Test Selection" field. Select at least one SMU.

- **2-21003 No Pin is selected. Please select at least one pin, chuck or input board.**

  No pin is selected in the "Test Selection" field. Select at least one pin, chuck pin, or input board.

- **2-21004 Only a numeric value is allowed here.**

  Invalid character (non-numeric) is entered in the input field for numeric character. Enter a numeric value.

- **2-21005 Only an integer value is allowed here.**

  Invalid character (non-integer) is entered in the input field for integer. Enter an integer value.

- **2-21006 Invalid repeat count. Must be 1 to 999999 (integer).**

  Wrong value is specified for how many times to repeat the test. The value must be 1 to 999999.

- **2-21007 Invalid margin ratio. Must be 0.0 to 1.0 (float).**

  Wrong value is specified for the marginal ratio used to define the marginal test limit. The value must be 0.0 to 1.0.

- **2-21008 Can't read the file. The file does not exist or no read permission.**

  The file could not be read. The file does not exist or does not have read permission. Check if the file exists or check the permission of the file.

- **2-21009 Can't write the file. The file or directory has no write permission, or the directory does not exist.**

  The file could not be written. The file or directory for the file does not have write permission. Or the directory does not exist. Check that the directory exists, and check the permission of the directory and the file.

- **2-21012 Failed to start diagnostics (errno = *N*).**

  Kernel error No. *N* occurred in starting diagnostics. Check if the /opt/hp4070/lbin/diagBody4070 program exists and check the permission of the program. If the program does not exist, re-install the 4080 system software. Or check the number of processes in progress, and kill unnecessary process.

- **2-21013 Can't preserve the result data file (errno = *N*). Check permission or maybe diagnostics was not executed yet.**

  Kernel error No. *N* occurred in preserving the result data file by File: Preserve data... menu on the Diagnostics window. Execute diagnostics once at least. Or check the permission of the file and /var/opt/hp4070/diag directory.

- **2-21014 Failed to write the result data (errno = *N*). Check permission.**

  Kernel error No. *N* occurred in writing the result data. Check the permission of the file and /var/opt/hp4070/diag directory.

- **2-21015 Failed to read result data file (errno = *N*). Check permission.**

  Kernel error No. *N* occurred in reading the result data. Check the permission of the file and /var/opt/hp4070/diag directory.

- **2-21016 Failed to write to result data file (errno = *N*). Check permission or free disk space.**

  Kernel error No. *N* occurred in writing the result data file. Check the permission of the file and /var/opt/hp4070/diag directory. Or check the free disk space. File system full may cause this error.

- **2-23001 High-resolution ADC Board is defective or not installed.**

  The high-resolution ADC board is not installed or may be defective. Install or replace the board.

- **2-23002 GNDU Board is defective or not installed.**

  The GNDU board is not installed or may be defective. Install or replace the board.

- **2-23003 Chuck Connection Pin Board is defective or not installed.**

  The chuck connection pin board is not installed or may be defective. Install or replace the board.

- **2-23004 Low Current Input Board (Port** *Port_No.***) is defective or not installed.**

  The low current input board is not installed or may be defective. Install or replace the board.

- **2-23005 Kelvin Input Board is defective or not installed.**

  The kelvin input board is not installed or may be defective. Install or replace the board.

- **2-23006 CMU Input Board is defective or not installed.**

  The CMU input board is not installed or may be defective. Install or replace the board.

- **2-23007 Relay Test Board (Block** *Block_No.***) is defective or not installed.**

  The relay test board is not installed or may be defective. Install or replace the board.

- **2-23011 ADC Board attenuation factor measurement failed. port:** *Port***, expected:** *Data1***, measured:** *Data2*

  High-resolution ADC board failed the attenuation factor measurement test. If all ports failed this test, replace the high-resolution ADC board, otherwise follow the diagnostics execution results.

- **2-23012 ADC Board guard amp offset measurement failed. port:** *Port***, measured:** *Data*

  High-resolution ADC board failed the guard amplifier offset measurement test. Follow the diagnostics execution results. If all ports failed this test, replace the high-resolution ADC board.

- **2-23021 Status error occurred in SMU. port:** *Port***, status:** *Status*

  Status error occurred in the SMU specified by *Port*. Check the SMU if the 4080 failed the diagnostics.

- **2-23022 Status error occurred in CMU. status:** *Status*

  Status error occurred in the CMU. Check the CMU if the 4080 failed the diagnostics.

- **2-23023 Status error occurred in DVM. status:** *Status*

  Status error occurred in the DVM. Check the DVM if the 4080 failed the diagnostics.

- **2-23031 HF Matrix Board N is defective or not installed.**

  Reference configuration file (/etc/opt/hp4070/config/refconfig1) shows the existence of HF matrix board N, but it is not detected by the testhead firmware. HF matrix board may not be installed or may be defective. Install or replace the board.

- **2-23032 HF Matrix Addressing Board is defective or not installed.**

  Reference configuration file (/etc/opt/hp4070/config/refconfig1refconfig1) shows the existence of HF matrix addressing board, but it is not detected by the testhead firmware. HF matrix addressing board is not installed or may be defective. Install or replace the board.

- **2-23033 Pulse Switch N Board is defective or not installed.**

  Reference configuration file (/etc/opt/hp4070/config/refconfig1) shows the existence of pulse switch board N, but it is not detected by the testhead firmware. Pulse switch board N is not installed or may be defective. Install or replace the board.

- **2-23041 SMU (port:** *N***) is defective or not installed.**

  The SMU board is not installed in port *N* or is defective. Confirm that the SMU configuration. hp4070 -login command displays the 4080 system configuration.

- **2-31001 File already exists. Do you want to replace it?**

  If you want to replace the data, select **OK** button. Or else, select **Cancel** button.

- **2-31002 The test(s) requested on non-existent resources will be ignored.**

  Invalid test item, SMU, or pin number is selected. The selection is ignored.

- **2-33001 Cannot identify Pin Board. pin:** *Pin*

  Actual pin board configuration did not match the configuration file (/etc/opt/hp4070/config/1). Check the configuration file and define the actual configuration.

- **2-33002 Cannot identify SMU Board. port:** *Port*

  Actual SMU configuration did not match the configuration file (/etc/opt/hp4070/config/1). Check the configuration file and define the actual configuration.

- **2-33003 Cannot identify CMU.**

  Actual CMU configuration did not match the configuration file (/etc/opt/hp4070/config/1). Check the configuration file and define the actual configuration.

- **2-33004 Cannot identify DVM.**

  Actual DVM configuration did not match the configuration file (/etc/opt/hp4070/config/1). Check the configuration file and define the actual configuration.

- **2-33005 Testhead Power Line Cycle is mismatching. reference:** *Ref* **[Hz], actual:** *Act* **[Hz]**

  Actual line frequency did not match the configuration file (/etc/opt/hp4070/config/1). Check the configuration file and define the actual frequency.

- **2-33006 Cannot identify HF Matrix.**

  The command attempts to control the HF matrix, even though the reference configuration file (/etc/opt/hp4070/config/refconfig1) has the information of no MF matrix installed. Check the following items:

  - If the command that you execute is correct.
  - If the reference configuration file includes the correct system configuration information.

- **2-33007 Cannot identify Pulse Switch.**

  The command attempts to control the pulse switch, even though the reference configuration file (/etc/opt/hp4070/config/refconfig1) has the information of no pulse switch installed. Check the following items:

  - If the command that you execute is correct.
  - If the reference configuration file includes the correct system configuration information.

- **2-33008 Cannot identify PG*n*.**

  The command attempts to control the PG*n*, even though the reference configuration file (/etc/opt/hp4070/config/refconfig1) has the information of no PG*n* installed. Check the following items:

  - If the command that you execute is correct.

- If the reference configuration file includes the correct system configuration information.

- **2-33009 PG*n* type or GPIB address mismatch.**

  The PGn definition in the reference configuration file (/etc/opt/hp4070/config/refconfig1) is not correct. Check if the reference configuration file includes the correct system configuration information.

- **2-33011 Cannot find pin for connection test. Port:** *Port*.

  Cannot execute the connection test. At least one pin board must be installed in:

  - Slot 1 to 24 for the HF ports 1, 2, and 3.
  - Slot 25 to 48 for the HF ports 4, 5, and 6.
  - Slot 1 to 48 for the AUX ports 1 to 8.

- **2-41001 Margin detection disabled.**

  Detection mode for the marginal limit was disabled.

- **2-44343 Low Current Input Board** *N* **inner port test was skipped.**

  The low current input board relay test was skipped because of an SMU error. Repair the SMU connected to the low current input board *N*.

- **2-45001 Relay contact check failed.**
  description: *Test case*
  measured: *Data1*, **offset:** *Offset*, **Rmeas:** *Data2*,
  Rlimit: *Limit*

  Failed in the contact check of the HF Matrix Relay Test. Check the connection of the short adapter. Or do troubleshooting and replace defective part.

- **2-45002 Relay stuck check failed. description:** *Test case*
  board: *board*, **relay:** *relay*, **measured:** *Data1*, **expected:** *Data2*

  Failed in the open check of the HF Matrix Relay Test. Do troubleshooting and replace defective part.

- **2-45101 PG selftest failed. description: PG*n* Status code:** *Code*
  Message: *Message from PG*

  Failed in the selftest of the PG. Do troubleshooting and replace defective part.

- **2-45201 PG connection test failed. description: PG***n* **port:** *port* **expected:** *Data1* **measured:** *Data2*

  Failed in the PG connection test. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45202 PG connection test marginally passed. description: PG***n* **port:** *port* **expected:** *Data1* **measured:** *Data2*

  Passed in the PG connection test but the result is nearly test limit. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45301 Pulse switch test failed.**
  description: PG*n* **port:** *port* **control: TH**
  expected: *Data1* **measured:** *Data2*

  Failed in the testhead controlled pulse switch test. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45302 Pulse switch test marginally passed.**
  description: PG*n* **port:** *port* **control: TH**
  expected: *Data1* **measured:** *Data2*

  Passed in the testhead controlled pulse switch test but the result is nearly test limit. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45303 Pulse switch test failed.**
  description: PG*n* **port:** *port* **control:** *N* **switch:** *switch*
  expected: *Data1* **measured:** *Data2*

  Failed in the PG controlled pulse switch test (multiplexer switch). Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45304 Pulse switch test marginally passed.**
  description: PG*n* **port:** *port* **control:** *N* **switch:** *switch*
  expected: *Data1* **measured:** *Data2*

  Passed in the PG controlled pulse switch test (multiplexer switch) but the result is nearly test limit. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45305 Pulse switch test failed.**
  description: PG*n* **port:** *port* **control:** *N* **switch:** *switch*
  expected: *Data1* **measured:** *Data2*

  Failed in the PG controlled pulse switch test (open/close switch). Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45306 Pulse switch test marginally passed.**
  description: PG*n* **port:** *port* **control:** *N* **switch:** *switch*
  expected: *Data1* **measured:** *Data2*

  Passed in the PG controlled pulse switch test (open/close switch) but the result is nearly test limit. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45401 PG output level test failed.** **description: PG***n* **port:** *port*
  expected: *Data1* **measured:** *Data2*

  Failed in the PG output level test. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45402 PG output level test marginally passed.**
  description: PG*n* **port:** *port* **expected:** *Data1* **measured:** *Data2*

  Passed in the PG output level test but the result is nearly test limit. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45501 PG trigger test failed.** **description: PG***n* **port:** *port*
  expected: *Data1* **measured:** *Data2*

  Failed in the PG trigger test. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

- **2-45502 PG trigger test marginally passed.**
  description: PG*n* **port:** *port* **expected:** *Data1* **measured:** *Data2*

  Passed in the PG trigger test but the result is nearly test limit. Check the cable connection and connect properly. Or do troubleshooting and replace defective part.

3-xxxxx

- **3-10001 System software inconsistency discovered.**

  Unknown error occurred during the 4080 system software execution. Cannot continue the session.

- **3-10002 Cannot open file. filename:** *File.*

  Could not open the file specified by *File.* Check the file name and permission of the file.

- **3-10004 An illegal format data exists. filename:** *File.***data:** *Data*

  Tried to open file with invalid data format. Change the data format of the data *Data* in the *FIle* or remake datafile.

- **3-10006 Cannot write data to file. filename:** *Filename*

  Could not write data to *Filename*. Check the permission of file or directory. Or check the free disk space. File system full may cause this error.

- **3-10015 Cannot execute command. command path:** *Path*

  Check the *Path* command. The command may not have execute permission. If problem is not found, re-install 4080 system software.

- **3-10023 Failed to open Standard Set calibration data directory.**

  Could not open /etc/opt/hp4070/diag/caldata directory. Check the data directory. If problem is not found in the directory, permission and so on, re-install the 4080 system software.

- **3-10024 Cannot rename file. filename:** *Filename*

  Could not rename the file. Check the permission of file or directory. If problem is not found, re-install the 4080 system software.

- **3-11001 Failed to write the test selection file.**

  Un-specifiable error occurred during write operation of the test selection file. Check the permission of file.

- **3-11002 Failed to open temporary PVheader file.**

  Could not open /var/opt/hp4070/diag/PVheader file. Check the free disk space. File system full may cause this error. If problem is not found, re-install the 4080 system software.

- **3-11003 pvBody4070 process is alreay running. Something is wrong.**

  The pvBody4070 process has been started. Check the pvBody process.

- **3-11004 Failed to open pipe (errno = *N*).**

  Kernel error No. *N* occurred during communication with diagnostics program. Exit from the 4080 system software environment, then log into it again. Then execute diagnostics program.

- **3-11005 Failed to invoke diagBody4070 program (errno = *N*).**

  Kernel error No. *N* occurred when starting diagnostics. Check if the diagBody4070 program exists and check the permission of the program. If the program does not exist, re-install the 4080 system software. Or check the number of processes in progress, and kill unnecessary process.

- **3-13002 An error occurred in TIS.**

  Error occurred in TIS. Refer to the additional messages.

- **3-16001 Memory exhausted.**

  The measurement data could not be loaded because of small free memory space. Stop unnecessary applications and close windows to increase free memory.

- **3-16010 Failed to open the file. Check permission or diskette.**

  The file was not loaded/saved because error occurred during load/save of file. Check the permission of the file. Or check the diskette.

- **3-16011 Failed to read the file. Check permission or diskette.**

  The file was not loaded because error occurred during load of file. Check the permission of the file. Or check the diskette.

- **3-16012 Failed to write the file. Check permission or diskette.**

  The file was not saved because error occurred during save of file. Check the permission of the file. Or check the diskette.

- **3-16013 Failed to end printing (errno = *N*).**

  Error No. *N* occurred at end of printing. Cannot close the pipe for lp command.

- **3-20001 Failed to read Standard Set calibration data.**

  Could not read the data file in /etc/opt/hp4070/diag/caldata directory. Check the permission of the data file and directory.

- **3-20002 Failed to write Standard Set calibration data.**

  Could not write the data file in /etc/opt/hp4070/diag/caldata directory. Check the permission of the data file and directory.

- **3-20003 Failed to remove Standard Set calibration data.**

  Could not remove the data file in /etc/opt/hp4070/diag/caldata directory. Check the permission of the data file and directory.

- **3-21001 No test is selected. Please select at least one test.**

  No test is selected in the "Test Selection" field. Select at least one test.

- **3-21002 No SMU is selected. Please select at least one SMU.**

  No SMU is selected in the "SMU Selection" field. Select at least one SMU.

- **3-21003 No Pin is selected. Please select at least one pin, chuck or input board.**

  No pin is selected in the "Pin Selection" field. Select at least one pin or chuck pin.

- **3-21004 Failed to open the profile data file.**

  Could not open the profile data file. Check if the file exists and permission of file or directory. If this message is displayed when you open the system default file, check the /etc/opt/hp4070/diag/PVprofile.

- **3-21005 Failed to read the profile data file.**

  Could not read the profile data file. Check if the file exists and permission of file or directory.

- **3-21006 Failed to write to the profile data file.**

  Could not write the profile data file. Check if the file exists and permission of file or directory. If this message is displayed when you write the system default file, check the /etc/opt/hp4070/diag/PVprofile.

- **3-21008 Can't read the file. The file does not exist or no read permission, or the directory does not exist.**

  The file could not be read. The file or directory for the file does not have read permission. Or the directory does not exist. Check that the directory exists, and check the permission of the directory and the file.

- **3-21009 Can't write the file. The file or directory does not have write permission, or the directory does not exist.**

  The file could not be written. The file or directory for the file does not have write permission. Or the directory does not exist. Check that the directory exists, and check the permission of the directory and the file.

- **3-21010 The DMM address for PV is not specified. Specify GPIB address in Configuration Window.**

- **3-21011 Standard C/R set to be used is not specified. Specify their serial numbers in Configuration Window.**

- **3-21012 Specified value for** *Equipment_Name* **exceeds the ±5% range limit against the nominal value. Set the correct value.**

- **3-21013 GPIB address must be integer value. Specify an integer value for DMM address.**

  A non-integer value is specified for GPIB address. Specify an integer value.

- **3-21022 The Oscilloscope address for PV is not specified. Specify GPIB address in Configuration Window.**

  No GPIB address for the oscilloscope is specified in the Standard Selection window. Specify the GPIB address of the oscilloscope.

- **3-21023 Specified Oscilloscope GPIB address is an integer. Specify an integer value.**

  Specified GPIB address value for the oscilloscope is not integer. Specify a proper integer value for the GPIB address of the oscilloscope.

- **3-23001 High-resolution ADC Board is defective or not installed.**

  The high-resolution ADC board is not installed or may be defective. Install or replace the board.

- **3-23002 GNDU Board is defective or not installed.**

  The GNDU board is not installed or may be defective. Install or replace the board.

- **3-23003 Chuck Connection Pin Board is defective or not installed.**

  The chuck connection pin board is not installed or may be defective. Install or replace the board.

- **3-23004 Low Current Input Board (Port** *Port_No.***) is defective or not installed.**

  The low current input board is not installed or may be defective. Install or replace the board.

- **3-23005 Kelvin Input Board is defective or not installed.**

  The Kelvin input board is not installed or may be defective. Install or replace the board.

- **3-23006 CMU Input Board is defective or not installed.**

  The CMU input board is not installed or may be defective. Install or replace the board.

- **3-23007 Relay Test Board (Block** *Block_No.***) is defective or not installed.**

  The relay test board is not installed or may be defective. Install or replace the board.

- **3-23011 ADC Board attenuation factor measurement failed. port:** *Port***, expected:** *Data1***, measured:** *Data2*

  The high-resolution ADC board failed the attenuation factor measurement test. Follow the performance verification execution results. If all ports failed this test, replace the high-resolution ADC board.

- **3-23012 ADC Board guard amp offset measurement failed. port:** *port* **, measured:** *Data*

  The high-resolution ADC board failed the guard amplifier offset measurement test. Follow the diagnostics execution results. If all ports failed this test, replace the high-resolution ADC board.

- **3-23021 Status error occurred in SMU. port:** *port***, status:** *Data*

  Status error occurred in the SMU specified by *Port*. Check the SMU if the 4080 failed the performance verification.

- **3-23022 Status error occurred in CMU. status:** *Status*

  Status error occurred in the CMU. Check the CMU if the 4080 failed the performance verification.

- **3-23023 Status error occurred in DVM. status:** *Status*

  Status error occurred in the DVM. Check the DVM if the 4080 failed the performance verification.

- **3-23030 Error in iopen() on GPIB device (DMM for PV).**

  An error in iopen() occurred when the PV4070 program tries to communicate the DMM (Digital Multimeter). Confirm if the specified logical unit of the DMM is correct or if the SICL configuration has the specified logical unit entry.

- **3-23031 DMM for PV is not found.**

  The DMM for PV is not responsed by the serial polling on the specified logical unit+GPIB address. Check the following items:

  - If the GPIB address setting of the DMM is correct.
  - If another GPIB device other than the DMM is on the specified GPIB address.
  - If the GPIB cable is not broken.

- **3-23032 The DMM address must specify both logical unit and GPIB address.**

  Logical unit or GPIB address of the DMM is not specified in the Standard Selection window. Specify both the logical unit and GPIB address of the DMM.

- **3-23034 Error in iread()/iwrite() on GPIB device (DMM for PV).**

  An error occurred during the communication between the computer and the DMM. Check the following items:

  - If the GPIB cable is not broken.
  - If the GPIB interface of the computer is not broken.
  - If the GPIB interface of the DMM is not broken.

- **3-23036 Each test pin must be connected to different connector pin.**

  Multiple test pins are connected to the same connector. Change the pin assignment.

- **3-23051 HF Matrix Board *N* is defective or not installed.**

  Reference configuration file "/etc/opt/hp4070/config/refconfig1" shows the existence of HF matrix board *N*, but it is not detected by the testhead firmware. HF matrix board may not be installed or may be defective. Install or replace the board.

- **3-23052 HF Matrix Addressing Board is defective or not installed.**

  Reference configuration file "/etc/opt/hp4070/config/refconfig1" shows the existence of HF matrix addressing board, but it is not detected by the testhead firmware. HF matrix addressing board is not installed or may be defective. Install or replace the board.

- **3-23053 Pulse Switch *N* Board is defective or not installed.**

  Reference configuration file "/etc/opt/hp4070/config/refconfig1" shows the existence of pulse switch board *N*, but it is not detected by the testhead firmware. Pulse switch board *N* is not installed or may be defective. Install or replace the board.

- **3-23054 Error in iopen() on GPIB device (Oscilloscope for PV).**

  An error in iopen() occurred when the PV4070 program tried to communicate the oscilloscope. Confirm if the specified logical unit of the oscilloscope is correct or if the SICL configuration has the specified logical unit entry.

- **3-23055 Oscilloscope for PV is not found.**

  The oscilloscope for PV is not responsed by the serial polling on the specified logical unit+GPIB address. Check the following items:

  - If the GPIB address setting of the oscilloscope is correct.
  - If another GPIB device other than the oscilloscope is on the specified GPIB address.
  - If the GPIB cable is not broken.

- **3-23056 The Oscilloscope address must specify both logical unit and GPIB address.**

  Logical unit or GPIB address of the oscilloscope is not specified in the Standard Selection window. Specify both the logical unit and GPIB address of the oscilloscope.

- **3-23057 Error in iread()/iwrite() on GPIB device (Oscilloscope for PV).**

  An error occurred during the communication between the computer and the oscilloscope. Check the following items:

  - If the GPIB cable is not broken.
  - If the GPIB interface of the computer is not broken.
  - If the GPIB interface of the oscilloscope is not broken.

- **3-23070 Error in itimeout() on GPIB device (DMM for PV).**
  3-23071 Error in iclear() on GPIB device (DMM for PV).
  3-23072 Error in itermchr() on GPIB device (DMM for PV).

  An error occurred when the computer tried to initialize the DMM (Digital Multimeter). Check the following items:

  - If the GPIB cable is not broken.
  - If the GPIB interface of the computer is not broken.
  - If the GPIB interface of the DMM is not broken.
  - If the DMM is working correctly.

- **3-23080 Error in itimeout() on GPIB device (Oscilloscope for PV).**
  3-23081 Error in iclear() on GPIB device (Oscilloscope for PV).
  3-23082 Error in itermchr() on GPIB device (Oscilloscope for PV).

  An error occurred when the computer tried to initialize the oscilloscope. Check the following items:

  - If the GPIB cable is not broken.
  - If the GPIB interface of the computer is not broken.
  - If the oscilloscope is working correctly.
  - If the GPIB interface of the oscilloscope is not broken.

- **3-24302 Cannot find available SMU. Please check using DIAG4070.**

  Could not find available SMU for calibrating the relay test board. Check the testhead configuration by using DIAG4070.

- **3-24303 Failed to calibrate a relay test board. Please check using DIAG4070.**

  Could not calibrate a relay test board and could not check of measurement pin position is correct. Check the relay test board by using DIAG4070.

- **3-24501 Each test pin must be connected to different connector pin.**

  Multiple test pins are connected to the same connector. Change the pin assignment.

- **3-24801 Cannot specify chuck connection pin for test pin.**

  Chuck connection pin was specified to be CMU bias test pin. Set CMU bias test pin to other pin.

- **3-24901 One of 1/2/3/4/48 pin should be available for SMU Output Resistance Test.**

  Check the pin board 1, 2, 3, 4 and 48 by using DIAG4070.

- **3-24902 Pin *N* is defective or not installed. SMU Output Resistance Test needs this pin board.**

  The pin *N* is not installed or may be defective. Check the pin *N* by using DIAG4070.

- **3-24903 The SMU Output Resistance Test requires at least two SMUs.**

  SMU is malfunctioning or at least two SMUs are not installed. Check the SMU function and configuration by using DIAG4070.

- **3-24951 Each test pin must be in different block.**

  Pulse level test and pulse parameter test require that at least one pin board is installed in each matrix block of the testhead. One pin board must be installed in the slot numbers 1 to 24, and another pin board must be in the numbers 25 to 48.

- **3-24952 Each test pin must be a multiple of 4.**

  Test pins specified for the pulse parameter test must be multiples of 4. That is, they are 4, 8, 12, 16, ... , 48. Check the pin configuration.

- **3-25001 Calibration Bus(block: *N*) does not have enough performance. Please check using DIAG4070**.

  The calibration is discontinued because calibration bus does not have enough performance. Check the calibration bus by using DIAG4070 and calibrate again.

- **3-25002 SMU *N* does not have enough performance. Please check using DIAG4070. (*V* [V]: *I* [A], *V* [V]: *I* [A])**

  The calibration is discontinued because SMU *N* does not have enough performance. Check the SMU *N* by using DIAG4070 and calibrate again.

- **3-25003 Failed to measure Reference Resistor.**
  (measured: *Measured_R* **[ohm], expected:** *Expected_R* **[ohm])**

  3-25004 Failed to measure Reference Resistor.
  (expected: *R1* **[ohm], min:** *R2* **[ohm], max:** *R3* **[ohm])**

  Could not measure the reference resistor. Check the cable connections, DMM for PV, and high-resolution ADC board.

- **3-25011 ADC Reference Volt measurement failed. (measured:** *Measured_V* **[V], expected:** *Expected_V* **[V]).**

  3-25012 ADC B-COM Volt measurement failed.(B-COM: *V* **[V])**

  Check the high-resolution ADC board by using DIAG4070. Or check the cable connections.

- **3-25021 ADC R Ref Calibration must be executed before SMU R Calibration.**

  The R Ref Calibration was not executed before SMU Ref Calibration. Re-calibrate SMU reference by using **CAL SMU**.

- **3-25031 Vtop/Vbase measurement for skew calibration was failed. Check PG*n*-HF*m* connection.**

  Failed to measure Vtop or Vbase value for the skew calibration. Check the cable connections between the PG*n* and HF*m*. If the skew calibration still fails with this error, check the cable connections between the PV pulse fixture and signal input connector of the oscilloscope.

- **3-25032 PG*n* delay time measurement for skew calibration was failed.**

  Failed to measure delay times for the skew calibration. Check the cable connections between the PG*n* and HF*m*. If the skew calibration still fails with this error, check the cable connections between the PV pulse fixture and signal input connector of the oscilloscope.

- **3-26001 Can't read the file. The file does not exist or no read permission or illegal path name is specified. Check the path name.**

- **3-26002 Can't write the file. The file or directory has no write permission, or an illegal path name is specified. Check the path name.**

- **3-26003 Failed to initiate spooler command. Check the specified command.**

- **3-26004 Failed to print (errno =** *Error***).**

  Could not send the data to spooler command. Check the spooler setting.

- **3-26006 Failed to open diskette directory. Check the following:**
  DOS format diskette is inserted into the disk drive.
  The specified disk drive path name is correct.
  The scsifloppy device driver is present in the HP-UX kernel.
  The DOS-UTIL fileset is installed.
- **3-26010 File format error. Specified field delimiter is not found. Check the file format and delimiter specification.**
- **3-26011 File format error. Specified string quotation mark is not found. Check the file format and quotation specification.**
- **3-26012 File format error. Numerical value is not found in a numeric field.**
- **3-26013 File format error. No valid data.**
- **3-27001 An error occurred in Testhead. status:** *Status*

  Testhead timeout occurred when ADC board was calibrated. Check the ADC board, CPU board and so on.
- **3-31001 File already exists. Do you want to overwrite it?**

  If you want to overwrite the data, select **OK** button. If not, select **Cancel** button.
- **3-31002 The test(s) requested on non-existent resources will be ignored.**

  Invalid test item, SMU, or pin number is selected. The selection is ignored.
- **3-33001 Cannot identify Pin Board. pin:** *Pin*

  Actual pin board configuration did not match the configuration file (/etc/opt/hp4070/config/refconfig1). Check the configuration file and define the actual configuration.
- **3-33002 Cannot identify SMU Board. port:** *Port*

  Actual SMU configuration did not match the configuration file (/etc/opt/hp4070/config/refconfig1). Check the configuration file and define the actual configuration.
- **3-33003 Cannot identify CMU.**

  Actual CMU configuration did not match the configuration file (/etc/opt/hp4070/config/refconfig1). Check the configuration file and define the actual configuration.

- **3-33004 Cannot identify DVM.**

  Actual DVM configuration did not match the configuration file (/etc/opt/hp4070/config/refconfig1). Check the configuration file and define the actual configuration.

- **3-33005 Testhead Power Line Cycle in config file does not match actual line frequency. reference:** *Ref* **[Hz], actual:** *Act* **[Hz]**

  Actual line frequency did not match the configuration file (/etc/opt/hp4070/config/refconfig1). Check the configuration file and define the actual frequency.

- **3-33006 Cannot identify HF Matrix.**

  The command attempts to control the HF matrix, even though the reference configuration file "/etc/opt/hp4070/config/refconfig1" has the information of no MF matrix installed. Check the following items:

  - If the command that you execute is correct.
  - If the reference configuration file includes the correct system configuration information.

- **3-33007 Cannot identify Pulse Switch.**

  The command attempts to control the pulse switch, even though the reference configuration file "/etc/opt/hp4070/config/refconfig1" has the information of no pulse switch installed. Check the following items:

  - If the command that you execute is correct.
  - If the reference configuration file includes the correct system configuration information.

- **3-33008 Cannot identify PG***n*.

  The command attempts to control the PG*n*, even though the reference configuration file "/etc/opt/hp4070/config/refconfig1" has the information of no PG*n* installed. Check the following items:

  - If the command that you execute is correct.
  - If the reference configuration file includes the correct system configuration information.

- **3-36001 File already exists. Do you want to overwrite it?**

  If you want to overwrite the data, select **OK** button. If not, select **Cancel** button.

- **3-41001 Is it ok to remove the Standard Set calibration data? Note that this operation is not recoverable.**

  If you want to remove the data, select **OK** button. If not, select **Cancel** button.

- **3-41002 Do you want to discard all unsaved changes? Note that this operation is not recoverable.**

  If you want to discard all unsaved changes, select **OK** button. If not, select **Cancel** button.

- **3-41003 The currently modified data will be lost. Do you want to save them before loading the new Standard Set data?**

  If you want to save the data, select **OK** button. If not, select **Cancel** button.

- **3-45701 Relay contact check failed. description:** *Test case* **measured:** *Data1*, **offset:** *Offset*, **Rmeas:** *Data2*, **Rlimit:** *Limit*

  Failed in the contact check of the HF Matrix Relay Test. Check the connection of the short adapter. Or do troubleshooting and replace defective part.

- **3-45702 Relay stuck check failed. description:** *Test case* **board:** *board*, **relay:** *relay*, **measured:** *Data1*, **expected:** *Data2*

  Failed in the open check of the HF Matrix Relay Test. Do troubleshooting and replace defective part.

- **3-45801 Pulse switch test failed.**
  description: *Description* **measured:** *Data*

  Failed in the pulse switch test (on/off test). Check the test lead connections and connect properly. Or do troubleshooting and replace defective part.

- **3-45901 Pulse level test failed.**
  description: *Test case* **measured:** *Data*

  Failed in the pulse level test. Check the connection from PGU and HF port to the signal input port of the oscilloscope. And connect properly. Or do troubleshooting and replace defective part.

- **3-45921 Pulse parameter test failed. description:** *Description* **measured:** *Data*

  Failed in the pulse parameter test (except for overshooting). Check the connection from PGU and HF port to the signal input port of the oscilloscope. And connect properly. Or do troubleshooting and replace defective part.

- **3-45922 Pulse parameter test failed due to overshoot. description:** *Description* **measured:** *Data*

  Failed in the pulse parameter test due to overshoot. Check the connection from PGU, HF port to the oscilloscope. And connect properly. Or do troubleshooting and replace defective part.

4-xxxxx

- **4-110001 Cannot make connection because no session is logged in now.**

  Before starting up the IDP, you need to log in to the 4080 operating environment. Execute hp4070 -login command before executing idp4070 command.

- **4-110004 Break connection because TIS Daemon process was down.**

  TIS daemon was down during IDP operation. Log in to the 4080 again, and execute idp4070 command.

- **4-110005 Cannot send event data to idp4070.**

  IDP was down abnormally. Exit IDP by choosing File: Exit menu on the IDP window.

- **4-120001 Cannot allocate memory for reading last measurement data.**

  The measurement data could not be loaded when choosing File: Open Last Test menu on the Graph window of IDP because of small free memory space. Stop unnecessary applications and close windows to increase free memory size.

- **4-202004 Allocation size is too large. Ignored.**

  Tried to load data file that was edited or broken. Internal error. Number of vector data is larger than the allocated size. Excess data is not set.

- **4-202005 Data size is larger than allocated size. The data is ignored.**

  Tried to load data file that was edited or broken. Internal error. Tried to set number of vector data that was negative number. No action.

- **4-202006 Invalid block number is selected. Ignored.**

  Internal error. Block number of the data block to be cleared is a negative number or over the maximum number. No action.

- **4-202007 Selected block size is too large. Last selected block is ignored.**

  Tried to load data file that was edited or broken. Internal error. Too large block size was specified for block selection. The block is not selected.

- **4-202009 Measurement vector number is invalid. The vector is ignored.**

  Tried to load data file that was edited or broken. Internal error. Tried to set number of the vector data that is negative number, zero, or over the number of registered data. The vector data is not set.

- **4-205011 Warning: Illegal data format in data file. Ignored. Too long line. Line =** *Line_No.*.

  Data file to be loaded was edited or broken. Number of characters in line *Line_No.* of data file to be loaded was over 128 characters. The line is ignored. Delete the line or edit the line so that it is 128 or less characters.

- **4-205012 Warning: Too many data in one data block: <= 2002 data per data block. Used only the first 2002 data.**

  Data file to be loaded was edited or broken. Number of data in a vector data is over 2002. Only the first 2002 data are loaded. Delete the excess data so that number of data is 2002 or less.

- **4-205013 Warning: Bad data Tag found in data file. Ignored. Line =** *Line_No.*.

  Data file to be loaded was edited or broken. Invalid specifier was used in line *Line_No.* of the data file. The line is ignored. Delete the line. Or edit the data file. Use ": (colon)", for example, "BX:", "BY:" and so on, to indicate the specifiers.

- **4-205014 Warning: Cannot use Attributes Tag in data block portion. Ignored. Use this Tag in header portion of data file. Line =** *Line_No.*.

  Data file to be loaded was edited or broken. Attribute tag is inserted in line *Line_No.* of the data file to be loaded. The line is ignored. Delete the line.

- **4-205015 Warning: Illegal data format in data file. Ignored. Line =** *Line_No.*.

  Data file to be loaded was edited or broken. Non-numeric data is included in line *Line_No.*. The line is ignored. Delete the non-numeric data, or exchange it with the numeric data.

- **4-205016 Warning: Illegal range format in data file for** *Data***. Ignored. Line =** *Line_No.*.

  Data file to be loaded was edited or broken. Could not get the data from the lines #XI, #X and #Y for graph in the data file. The line *Line_No.* is ignored. Delete the line. Or edit the file and insert the proper value.

- **4-205018 Warning: Illegal Character in data file for** *Data*. **Used NULL String. Line =** *Line_No.*.

  Data file to be loaded was edited or broken. Invalid character was used in line *Line_No.*. The line is ignored. Edit the file. Numeric characters, and '–' (minus), '+' (plus), '_' (underscore), ' ' (space), '.' (period) and ',' (comma) are allowed for the data.

- **4-205019 Warning: Illegal string description in data file. Ignored. Line =** *Line_No.*.

  Data file to be loaded was edited or broken. Strings are not enclosed by double quotations ('). The line *Line_No.* is ignored. Edit the file and properly enclose the strings by double quotations.

- **4-205101 Warning: Bad data range for Log scale;***Axis*. **Turned off log scale mode.**

  Tried to set minimum value or maximum value of X axis or Y axis in log scale to 0 or negative value. Graph scale is set to linear scale automatically. The data must be positive value for log scale.

- **4-205102 Warning: Bad data for Log scale. Turned off log scale mode.**

  Tried to set X-axis data or Y-axis data in log scale to negative value. Graph scale is set to linear scale automatically. The data must be positive value for log scale.

- **4-205103 Warning: Bad range for scale. Set previous value.**

  Difference between maximum value and minimum value of the graph is almost 0. Minimum value and maximum value are set to the previous values automatically. Specify a bigger difference.

- **4-205112 Warning: No Y data. Cannot plot any data.**

  Data file to be loaded was edited or broken. Internal error. Y vector data does not exist although X vector data exists. Graph does not plot any data. Create one Y vector data at least.

- **4-211000 Can't open file.**

  Error occurred during loading/saving the IDP data file. The data file is not loaded/saved. Confirm that the directory for the data file exists, and the permission of the directory and the data file.

- **4-211001 Can't write file.**

  Error occurred while saving the IDP data file. The data file is not saved. Confirm that there is sufficient free disk space. Prepare approx. 10 KByte for one IDP data file.

- **4-211002 Can't read file.**

  Size of the data file is less than the size of IDP data file. The data file is not loaded. Specify the IDP data file name properly. If the file was edited, the file cannot be loaded.

- **4-211003 Saving to a file is not allowed when some resource fields are modified.**

  Tried to save the IDP data file before setup was completed (clicking **Set** button) on the IDP window. The data file is not saved. Close the Save Dialog and click **Set** button. Then save the data file.

- **4-211004 File format revision mismatch. Can't read the file.**

  Tried to load data file, but did not match the IDP data format. The data file is not loaded. Specify the file name of the IDP data. If the IDP data file was edited and changes made, the data file cannot be loaded.

- **4-211005 Can't close the resource display when some resource fields are modified.**

  Tried to delete a resource from the Resource Selection dialog before setup for the resource was completed (clicking **Set** button) on the IDP window. The resource is not deleted from the Resource Selection dialog. Close the Resource Selection dialog, click **Set** button. Then delete the resource from the Resource Selection dialog.

- **4-211006 Can't open last measurement data.** *Test_name* **doesn't create array data.**
  4-211007 Can't open last measurement data. No measurement has been performed.
  4-211008 Can't open last measurement data. There is no measurement data.

  The measurement might have been executed on offline system.

  Data on the Graph window does not change. Before choosing File: Open Last Test menu, you must execute measurement that makes array data.

  File: Open Last Test menu on the Graph window is used to display array data that was measured in the On-line mode.

- **4-212001 Cannot create measurement blocks.**

  Number of measurement data exceeds the maximum number. Measurement block is not created. Clear the data loaded on the Graph.

- **4-212002 Cannot create vector data blocks.**

  Tried to load data file that was edited or broken. Internal error. Number of data for vector data is zero, negative number, or exceeds the maximum number. No action.

- **4-212003 Cannot allocate data.**

  Tried to load data file that was edited or broken. Internal error. Tried to create data with negative number or over the maximum number.

- **4-215100 Error: Data block index value must be 1 to 7.**

  Tried to load data file that was edited or broken. Internal error. Data block index value must be 1 to 7.

- **4-215150 Error: Bad file name: May be too long file name or no suffix;***File_name*.

  Name of file to load is too long or no suffix. The file is not loaded. Enter the proper name for the file on the Load dialog, and try to load the file again.

- **4-215151 Error: Bad file suffix.**

  Suffix of the file name entered is not '.dpg' or '.dps'. The file is not loaded. Enter the proper name for the file in the Load Dialog, and try to load the file again.

- **4-215152 Error: Bad file type; not an ordinary file.**

  File is not ASCII file. The file is not loaded. Enter the proper name for the file in the Load Dialog, and try to load the file again.

- **4-215155 Error: Too long file path name. Maximum 1018 characters.**

  File name over 1018 characters was entered. The file is not saved. Use file name with 1018 or less characters and try to save the file again.

- **4-215156 Error: Specified path name is too long. Only the leading 1023 characters are used.**

  Path name over 1023 characters was entered. The file is not saved. Use path name with 1023 or less characters and try to save the file again.

- **4-215301 Error: Syntax error in dump command;** *Command*.

  Syntax error occurred in dump command. Enter the proper command for dump/print.

- **4-215303 Error: Cannot generate PostScript code.**

  Temporary PostScript file could not be created. Secure enough free disk space for temporary file.

- **4-225001 System Error: Cannot open data file;** *File_name*.

  Specified data file does not exist, improper directory is specified, or temporary file for dump cannot be created. Check the name of data file, or permission of the file and directory. Or secure enough free disk space for temporary file.

5-xxxxx

- **5-10001 Cannot make connection with TIS Daemon. No session or environment variable PCS_SESSION_ID is not properly set.**

  Execute hp4070 -login command. Or confirm the setting of the environmental variable PCS_SESSION_ID.

- **5-10002 Cannot allocate memory. Please quit and try again.**

  To allocate memory, stop the action and try it again.

- **5-10003 Environmental variable: PCS_SESSION_ID is not set.**

  Quit from the 4080 operating environment, set PCS_SESSION_ID, and then log in again.

6-xxxxx

- **6-20001 Cannot translate 4062 port address into 4070 port address.**

  Specified port address for the 4062 could not be translated to the 4080 port address. Check the port address, and specify the correct port address.

- **6-20002 Invalid port is assigned.**

  Specified port address is not correct. Confirm the port address.

- **6-20003 Invalid pin is assigned.**

  Specified pin number is not correct. Confirm the pin number.

- **6-20004 [F_ROM] Pin number must be 0 to 48.**

  Pin number must be 0 to 48. Enter a valid value.

- **6-20005 Specified pin board is defective or not installed.**

  Pin board for the specified pin number is not installed or defective. Confirm the pin board configuration. hp4070 -login command displays the 4080 system configuration.

- **6-20006 From_pin is out of range.**

  Pin number assigned for the *first pin* parameter of connect_th function is not correct. Confirm the pin number.

- **6-20007 To_pin is out of range.**

  Pin number assigned for the *last pin* parameter of connect_th function is not correct. Confirm the pin number.

- **6-20008 1st parameter must be 1 (connect) or 0 (disconnect).**

  Wrong value was specified for the first parameter of tap_port function. The value must be 0 (disconnect) or 1 (connect). Check the value and specify the proper value.

- **6-20009 2nd parameter must be port address of SMU/AUX port 1 or 2.**

  Wrong value was specified for the second parameter of tap_port function. The value must be the port address of SMU port 1 or 2, or AUX port 1 or 2. Check the value and specify the proper value.

- **6-20010 3nd parameter must be port address of SMU/AUX port 1 or 2.**

  Wrong value was specified for the third parameter of tap_port function. The value must be the port address of SMU port 1 or 2, or AUX port 1 or 2. Check the value and specify the proper value.

- **6-20011 Improper SMU voltage range specified.**

  Voltage range setting for SMU is not correct. Check the value of the output voltage and the voltage range, and specify the proper value.

- **6-20012 Improper SMU voltage source, voltage output range, or compliance specified.**

  Output voltage value, voltage output range, or current compliance value are not correct. Verify the values of output voltage, voltage output range, and current compliance and then specify the correct value.

- **6-20013 Improper SMU current range specified.**

  Current range setting of SMU is not correct. Check the value of the output current and the current range, and specify the proper value.

- **6-20014 Improper SMU current source or compliance specified.**

  Value of output current or voltage compliance is not correct. Check the value of output current and voltage compliance, and specify the proper value.

- **6-20015 Specified SMU is defective or not installed.**

  Specified SMU board is not installed or defective. Confirm the SMU configuration. hp4070 -login command displays the 4080 system configuration.

- **6-20016 Source pin must be connected to an SMU.**

  Specified pin board is not connected to SMU. Check the measurement program. The program must have a program line that connects the SMU to the pin board *before* the program line that forces the output of the SMU specified by the pin number.

- **6-20017 Source port must be an SMU or a pin connected to an SMU.**

  Source port address used in the force_meas function is not correct. Source port address must be SMU port address or the pin number of the pin board connected to the SMU. Check the port address and specify the proper value.

- **6-20018 Measure pin must be connected to an SMU.**

  Specified pin board is not connected to SMU. Check the measurement program. The program must have a program line that connects the SMU to the pin board *before* the program line that performs measurement using the SMU specified by the pin number.

- **6-20019 Measure port must be an SMU or a pin connected to an SMU.**

  Measurement port address is not correct. Measurement port address must be SMU port address or the pin number of the pin board connected to the SMU. Check the port address and specify the proper value.

- **6-20020 3rd parameter must be 1 (voltage source) or 2 (current source).**

  Wrong value was specified for the third parameter of force_meas function. The value must be 1 (voltage source) or 2 (current source). Check the value and specify the proper value.

- **6-20021 Wait time must be 0 to 655.35 sec.**

  Specified value for the wait time is out of range. The value must be 0 to 655.35 seconds. Check the value and specify the proper value.

- **6-20022 Hold time must be 0 to 655.35 sec.**

  Specified value for the hold time is out of range. Available value for the hold time is 0 to 655.35 sec. Check the value and specify the proper value.

- **6-20023 Port must be CMH when forcing voltage on CMU.**

  Invalid port was specified for force_v. When setting the dc bias of CMU, the CMH port address must be specified. Check the port address and specify the correct port.

- **6-20024 Wait time must be 0 to 655.35 sec.**

  Wait time value is not correct. Available value for the wait time is 0 to 655.35 sec. Check the value and specify the proper value.

- **6-20025 Size of port array must be greater than or equal to 1.**

  No port is defined for the array used to specify the multiple ports. Check the value and size of the array, and define the array properly.

- **6-20026 1st parameter must be an SMU port address or a pin connected to an SMU.**

  Wrong value was specified for the first parameter of stand_by_port function. The value must be the SMU port address or the pin number of the pin board connected to the SMU. Check the value and specify the value properly.

- **6-20027 2nd parameter must be an SMU port address or a pin connected to an SMU.**

  Wrong value was specified for the second parameter of stand_by_port function. The value must be the SMU port address or the pin number of the pin board connected to the SMU. Check the value and specify the value properly.

- **6-20028 3rd parameter must be an SMU port address or a pin connected to an SMU.**

  Wrong value was specified for the third parameter of stand_by_port function. The value must be the SMU port address or the pin number of the pin board connected to the SMU. Check the value and specify the value properly.

- **6-20029 *N*th parameter must be an SMU port address or a pin connected to an SMU.**

  Wrong value was specified for the *N*th parameter of stand_by_port function. The value must be the SMU port address or the pin number of the pin board connected to the SMU. Check the value and specify the value properly.

- **6-20030 Cannot connect Extended I/O port to the Chuck terminal.**

  Cannot specify Extended Path port for connect_chuck function. Specify an SMU, GNDU, or AUX port address for the connect_chuck function.

- **6-20031 Ports must be SMU or CMU.**

  Invalid port was specified for disable_port function. Must specify an SMU or CMU port address, or the pin number of a pin board connected to the SMU or CMU. Check the port value and specify the port properly.

- **6-20032 Adc must be 0 (ADC_PERCH) or 1 (ADC_REF).**

  Wrong value was specified for the parameter to select ADC. The value must be 0 (high-speed ADC) or 1 (high-resolution ADC). Check the value and specify the value properly.

- **6-20033 Filter must be 0 (FILTER_OFF) or 1 (FILTER_ON).**

Wrong value was specified for the parameter to select SMU filter status. The value must be 0 (filter off) or 1 (filter on). Check the value and specify the value properly.

- **6-20034 Wait time ratio must be 0.0 to 10.0.**

  Wrong value was defined for the wait time factor. The value must be 0 to 10. Check the value and specify the value properly.

- **6-20035 Integration time mode setting must be 0, 1, 2, or 3.**

  Wrong value was specified for the mode parameter of set_smu or set_adc function. The value must be 0 (manual), 1 (short), 2 (medium) or 3 (long). Check the value and specify the proper value.

- **6-20036 Averaging or integration time is out of range.**

  Wrong value was specified for the integration time or averaging parameter of set_adc or set_smu function, or the integration time parameter of the set_adc3458 function. Check the value and specify the proper value.

- **6-20037 Autozero must be 0 (AUTOZERO_OFF) or 1 (AUTOZERO_ON).**

  Wrong value was specified for the auto zero parameter of set_adc or set_adc3458 function. The value must be 0 (auto zero off) or 1 (auto zero on). Check the value and specify the proper value.

- **6-20038 Port must be an SMU or a pin connected to an SMU.**

  Port address setting is not correct. The port address must be the SMU port address or the pin number of the pin board connected to the SMU. Check the value and specify the proper value.

- **6-20039 Port must be an SMU or a pin connected to an SMU.**

  Port address setting is not correct. The port address must be the SMU port address or the pin number of the pin board connected to the SMU. Check the value and specify the proper value.

- **6-20040 Search mode must be 0 to 15 for Set_bsearch, 0 to 11 for Set_lsearch.**

  Wrong value was specified for the search mode parameter of the set_lsearch or set_bsearch function. The search mode value for set_lsearch must be 0 to 11, and the value for set_bsearch must be 0 to 15.

- **6-20041 Upper search boundary is equal to lower search boundary.**

  Same value was set for the start/stop parameters of set_lsearch function or the min/max parameters of set_bsearch function. Start (or min) value must be different from the stop (or max) value.

- **6-20042 Step value must be greater than 0 for Set_lsearch or Set_bdv_search.**

  Wrong value was specified for the step parameter for set_lsearch or set_bdv_search function. The value must be greater than 0.

- **6-20043 Target value must be less than measurement compliance.**

  Too large value was specified for the *target* parameter of set_lsearch or set_bsearch function. The value must be less than the value of the compliance parameter.

- **6-20044 Convergence condition in Accuracy mode must be 0 to 100.**

  Value of the convergence condition parameter of set_bsearch function in the Accuracy mode is out of range. The value must be 0 to 100.

- **6-20045 Convergence condition in Times mode must be 0 to 16.**

  Value of the convergence condition parameter of set_bsearch function in the Times mode is out of range. The value must be 1 to 16.

- **6-20046 Delay time must be 0 to 65.535 sec.**

  Setting value of the delay time is out of range. The value must be 0 to 65.535 sec. Check the value and specify the proper value.

- **6-20047 Skip_back value must be 0 to 100.**

  Wrong value was specified for the skip back parameter of set_lsearch function. The value must be 0 to 100.

- **6-20048 Set_skip may not be used with Set_bsearch or Set_bdv_search.**

  Cannot use set_skip with set_bsearch or set_bdv_search function in C language programming environment. Use set_skip with set_lsearch.

- **6-20049 Set_bsearch or Set_lsearch is not called before Search is called.**

  Before executing search function, set_lsearch or set_bsearch function must be executed to define the search measurement setup. The setup defined by set_lsearch or set_bsearch function is canceled by the init_system function.

- **6-20050 Compliance mode is not available for voltage measurement of Set_lsearch.**

  For the search mode 8 to 11 (compliance mode) of set_lsearch function, the output mode of sense port SMU must be set to VF (voltage force) mode. Before executing set_lsearch function, enter function (for example, force_v) to set the SMU output mode to VF mode.

- **6-20051 Sweep mode setting must be –4 to 4 (integer).**

  Improper sweep mode was specified for set_iv or set_piv function. The value must be –4 (Current, log, double), –3 (Voltage, log, double), –2 (Current, log, single), –1 (Voltage, log, single), 1 (Voltage, linear, single), 2 (Current, linear, single), 3 (Voltage, linear, double), or 4 (Current, linear, double). Check the sweep mode setting and set the proper mode.

- **6-20052 Number of steps must be 2 to 1001.**

  Improper number of steps was specified for set_iv or set_piv function. Value must be 2 to 1001. Check the value and set the proper value.

- **6-20053 Improper power compliance is specified.**

  Setting of the power compliance is out of range. Check the value and set the proper value.

- **6-20054 Sweep stop mode must be 1 (continue on error) or 2 (stop on error).**

  Improper stop mode was specified for the sweep measurement function. The value must be 1 (continue on error) or 2 (stop on error). Check the setting and set the proper value.

- **6-20056 Measurement mode must be 1 (voltage) or 2 (current).**

  Improper measurement mode was specified for sweep_iv or rsweep_iv function. The value must be 1 (voltage measurement mode) or 2 (current measurement mode).

- **6-20057 Start and stop values have different polarity.**

  In the logarithmic sweep measurement mode, polarity of the start and stop value must be the same.

- **6-20058 Hold time must be 0 to 655.35 sec.**

  Setting of hold time is wrong. The value must be 0 to 655.35.

- **6-20059 Difference of start and stop must be greater than** *Voltage*

  For set_bdv function, the difference between the *start* value and the *stop* value must be 10 V or more.

  For set_ileak function, the difference between the *output voltage* value and the *start* value must 10 V or more.

- **6-20060 Detection interval mode must be 0 (short) or 1 (long).**

  Improper *detection interval* was specified for measure_bdv or measure_ileak function. The value must be 0 (short mode) or 1 (long mode).

- **6-20061 Skip value must be 1 to 20000.**

  Improper *skip* value was specified for set_lsearch function. The value must be 1 to 20000.

- **6-20062 Target value must be less than the sense port compliance.**

  Improper *target* value was specified for set_lsearch function. The value must be less than compliance of sense port.

- **6-20063 Sync port must have the same V/I output mode as the control variable.**

  SMU output mode of the sync SMU defined for set_sync function must be the same output mode of the search SMU defined for set_bsearch or set_lsearch function. Set the SMU output mode of the sync SMU by entering force_v or force_i function before entering set_sync function.

  force_v function is used to set the voltage output mode, and force_i function is used to set the current output mode.

- **6-20064 Improper synchronous search offset value is specified.**

  Offset value of set_sync function is out of range. Or the offset value is not correct for the settings of the start/stop values of set_lsearch function or the min/max values of set_bsearch function. Set the proper value.

- **6-20066 Offset value of synchronous search port or compliance is improper.**

  Sync port SMU output did not match to the compliance value. Check the offset and compliance values of set_sync function, and the start/stop values of set_lsearch function or the min/max values of set_bsearch function, and set the proper value.

- **6-20067 Synchronization polarity mode must be 0 (positive) or 1 (negative).**

  Improper synchronization polarity mode was specified for set_sync function. The value must be 0 (positive sync) or 1 (negative sync).

- **6-20068 Synchronous sweep ratio must be 0.01 to 10.0.**

  Improper synchronous sweep ratio value was specified for set_sync function. The value must be 0.01 to 10.0.

- **6-20069 Improper synchronous sweep offset value specified.**

  Improper synchronous sweep offset value was specified for set_sync function. Check the value and set the proper value.

- **6-20070 Calculated start, stop or compliance of synchronous sweep is improper.**

  Sync port SMU output is out of range, or does not match to the compliance value. Check the offset, ratio and compliance values of set_sync function, or the start/stop values of the function for sweep measurement.

- **6-20071 Measurement mode must be 0 (linear) or 1 (saturation).**

  Invalid mode was specified for measure_vth function. The mode must be 0 (linear region) or 1 (saturation region).

- **6-20072 Number of points must be 2 to 20.**

  Invalid number of points was specified for measure_vth function. The number of points must be 2 to 20.

- **6-20073 Compliance must be specified when SMU V/I output mode is changed.**

  In the optimization level is 2 or 3, compliance value cannot be omitted for a function (force_v, force_i, etc.) that changes the SMU output mode. Confirm the optimization level and set the compliance value properly.

- **6-20074 Invalid pin number specified for Hpin.**

  Invalid pin number was specified for the CMU high port of corr_cmu function. Check the pin number and set the correct number.

- **6-20075 Invalid pin number specified for Lpin.**

  Invalid pin number was specified for the CMU low port of corr_cmu function. Check the pin number and set the correct number.

- **6-20076 C Compensation file is not loaded.**

  TIS function requires that capacitance compensation was executed before loading the capacitance compensation data file. Check /etc/opt/hp4070/ccdata file and /etc/opt/hp4070/ccdata0 file, and the permission of the files.

- **6-20077 Cp should not be 0.**

  Cp value of conv_mode function was set to zero. Cp value must not be zero.

- **6-20078 Proper sweep measurement is not done.**

  status_miv function could not read the measurement status of the last sweep measurement. Sweep measurement was not done properly. Check the measurement program. Or sweep measurement may be now in progress.

- **6-20079 Current is too large.**

  Too large current value was set for set_bdv function. Set the proper value.

- **6-20080 Vg_start and Vg_stop should not be same.**

  Same value was set for gate start and stop voltages of set_vth function. The same value cannot be set for these parameters. Set the values properly.

- **6-20081 Gate step voltage value must be greater than 0 V.**

  Gate step voltage must be greater than 0 V for the set_vth function. Set the value correctly.

- **6-20082 Id_start must be less than measurement port compliance.**

  Drain search start current of set_vth function must be less than the compliance value of the measurement port. Set the value properly.

- **6-20083 Delay time must be 0 to 65.535 sec.**

  Delay time value of set_vth function is out of range. Value must be 0 to 65.535 seconds. Set the value properly.

- **6-20084 Vg Skip_back value must be 0 to 100.**

  Vg Skip_back value of set_vth function is out of range. Value must be 0 to 100. Set the value properly.

- **6-20085 Vg Skip value must be 1 to 20000.**

  Vg Skip value of set_vth function is out of range. Value must be 1 to 20000. Set the value properly.

- **6-20086 Id_start value must be –0.1 to 0.1 A.**

  Drain start current value of set_vth function is out of range. Value must be –0.1 to 0.1 A. Set the value properly.

- **6-20090 Sweep_miv or Rsweep_miv can be used with Set_iv, not Set_piv.**

  sweep_miv and rsweep_miv functions for the multi-channel sweep measurement can only be used for sweep that was set up by set_iv function. The functions cannot be used with pulsed sweep, which is set up by set_piv function. Check and change the program.

- **6-20100 Measurement port is not connected to any pins.**

  Measurement SMU port is not connected to any pin board. Connect the SMU to the measurement pin.

- **6-20101 Sweep port is not connected to any pins.**

  Sweep measurement SMU port is not connected to any pin board. Connect the SMU to the measurement pin.

- **6-20102 The synchronous sweep port is not set up.**

  An array was specified in sweep_miv function for returning source values of synchronous sweep port, but synchronous port was not set up. Either delete the array name from sweep_miv or set up synchronous sweep port by using set_sync.

- **6-20103 The synchronous sweep port is not connected to any pins.**

  Synchronous sweep port is not connected to any pin board. Connect the SMU to the measurement pin.

- **6-20104 Specified sampling port is not connected to any pins.**

  SMU sampling measurement port is not connected to any pin board. Connect the SMU to the measurement pin.

- **6-20120 Force mode must be 1 (voltage force) or 2 (current force).**

  Improper mode was specified for the force mode of set_pbias function. The value must be 1 (voltage force) or 2 (current force).

- **6-20121 Pulse sweep mode setting must be –4 to 4 (integer).**

  Improper *pulse sweep mode* was specified for set_piv function. The value must be 1 (linear voltage), 2 (linear current), 3 (linear voltage double), 4 (linear current double), –1 (logarithmic voltage), –2 (logarithmic current), –3 (logarithmic voltage double), or –4 (logarithmic current double).

- **6-20122 Measurement mode must be 1 (voltage) or 2 (current).**

  Improper *measurement mode* was specified for measure_p function. The value must be 1 (voltage measurement) or 2 (current measurement).

- **6-20123 Peak and base current must have the same polarity.**

  For current, the *pulse base* and *pulse peak* values must be the same polarity for set_pbias function. Check the values and set the proper values.

- **6-20124 Peak, start and stop current must have the same polarity.**

  For current, *pulse base*, *start value*, and *stop value* must be the same polarity for set_piv function. Check the values and set the proper values.

- **6-20125 Width value must be 0.0005 to 2 sec.**

  Pulse width value is out of range. Value must be 0.0005 to 2 sec with 0.0001 sec resolution. Set the correct value.

- **6-20126 Period value must be 0.005 to 5 sec.**

  Pulse period value is out of range. Value must be 0.005 to 5 sec with 0.0001 sec resolution. Set the correct value.

- **6-20128 Period value must be greater than width + 2ms.**

  Pulse period value must be greater than pulse width + 2 ms. Set the pulse period or pulse width value correctly.

- **6-20130 Coefficient value must be greater than or equal to 0.**

  Coefficient value for wait time must be greater than or equal to 0.

- **6-20131 Parameter cannot be omitted, or specify NULL.**

  Parameter cannot be omitted, or you can specify NULL.

- **6-20200 Set_pbias must be executed before Measure_p.**

  No setup for measure_p function. Execute set_pbias function before executing measure_p function.

- **6-20201 Set_iv or Set_piv must be executed before Sweep_iv.**

  No setup for sweep_iv function. Execute set_iv or set_piv function before executing sweep_iv function.

- **6-20202 No measurement to stop by Rsweep_stop.**

  No real time sweep measurement to stop. rsweep_stop is used for stopping *real-time* sweep measurements only.

- **6-20203 Set_iv or Set_piv must be executed before Rsweep_iv.**

  No setup for rsweep_iv function. Execute set_iv or set_piv function before executing rsweep_iv function.

- **6-20204 Set_iv must be executed before Rsweep_miv.**

  No setup for rsweep_miv function. Execute set_iv function before executing rsweep_miv function.

- **6-20205 Set_iv must be executed before Sweep_miv.**

  No setup for sweep_miv function. Execute set_iv function before executing sweep_miv function.

- **6-20206 Set_lsearch, Set_bsearch, or Set_bdv_search must be executed before Search or Search_iv.**

  No setup for search or search_iv function. Execute set_lsearch, set_bsearch, or set_bdv_search function before executing search or search_iv.

- **6-20207 Synchronous port must be different from the primary sweep port.**

  Must use different SMU ports for the primary sweep port and synchronous port.

- **6-20208 Synchronous port must be different from the primary search port.**

  Must use different SMU ports for the primary search port and synchronous port.

- **6-20209 No sweep or search setup was set before Set_sync.**

  No setup for set_sync function. Execute set_iv, set_piv, set_bsearch, or set_lsearch function before executing set_sync function.

- **6-20210 Set_bdv must be executed before Measure_bdv.**

  No setup for measure_bdv function. Execute set_bdv function before executing measure_bdv function.

- **6-20211 Set_ileak must be executed before Measure_ileak.**

  No setup for measure_ileak function. Execute set_ileak function before executing measure_ileak function.

- **6-20212 Set_lsearch, Set_bsearch, or Set_bdv_search must be executed before Rsearch or Rsearch_iv.**

  No setup for rsearch or rsearch_iv function. Execute set_lsearch, set_bsearch, or set_bdv_search function before executing rsearch or rsearch_iv.

- **6-20213 No measurement to stop by Rsearch_stop.**

  No real time search measurement to stop. rsearch_stop is used for stopping *real-time* search measurements only.

- **6-20214 Set_vth must be executed before Measure_vth.**

  No setup for measure_vth function. Execute set_vth function before executing measure_vth function.

- **6-20215 Synchronous port must be different from the pulse bias port.**

  Must use different SMU ports for the pulse bias port and synchronous port.

- **6-20216 Pulse bias port must be different from the Synchronous port.**

  Must use different SMU ports for the synchronous port and the pulse bias port.

- **6-20220 Cannot call Status_miv during a sweep measurement.**

  Cannot execute status_miv function while *real-time* sweep measurement is executing.

- **6-20221 Fine search start current value must be smaller than breakdown current value.**

  The current value for fine search start must be smaller than the breakdown current value which sets compliance of an SMU in the set_bdv_search function.

- **6-20222 Fine search start current and breakdown current must have the same polarity.**

  The setup current values for fine search start and breakdown current must be the same polarity for the set_bdv_search function. Check the polarity and set the correct values.

- **6-20223 Cannot omit fine search start parameter.**

  Fine search start value must be specified when the skip parameter value is greater than 1 for the set_bdv_search function.

- **6-20224 Compliance check delay time must be 0 to 65.535 sec.**

  Setting value for comp check delay is out of range. The value must be 0 to 65.535 sec. Check the value and specify it correctly.

- **6-20301 Invalid mode number specified.**

  Mode number specified in the set_rangemode is incorrect. The mode number must be 0 or 1.

- **6-20402 Integration time mode setting must be 0, 1, 2, 3, or 4.**

  Integration time mode setting must be 0, 1, 2, 3, or 4. Enter a valid value.

- **6-20403 Must specify integration time mode setting for higher current range (0, 1, 2, 3, or 4).**

  The integration time mode is not set for the higher current range. You must specify the integration time.

- **6-20405 Mode value of advanced ranging must be 1, 2, or 3.**

  The mode value for changing the auto-ranging operation must be 1, 2, or 3.

- **6-20406 Ratio for advanced ranging must be from 11 to 100.**

  The ratio value to calculate the boudaries for auto-ranging operation must be 11 to 100.

- **6-20411 Boundary range must be -1.0 to 1.0.**

  The range value for boundary must be –1.0 to 1.0.

- **6-20520 Parameter on_off must be 0 or 1.**

  On_off parameter must be 0 or 1.

- **6-20601 The number of elements of array for parallel measurement must be 1 to 8.**

  The number of elements of array for multi-channel synchronous parallel measurement must be 1 to 8.

- **6-20602 Sweep mode must be same (single or double) among all ports.**

  On the sweep mode of single or double sweep, the same mode must be specified for all ports.

- **6-20603 P_set_iv must be executed before P_sweep_miv.**

  p_set_iv must be executed before p_sweep_miv.

- **6-20604 Start and stop must be same if source port is common.**

  Start and stop values must be same if source port is common.

- **6-20605 Offset and ratio must be same if sync port is common.**

  Offset and ratio values must be same if synchronous port is common.

- **6-20606 P_status_miv is not allowed here.**

  p_status_miv cannot be executed. The multi-channel synchronous parallel measurement is currently executed or has not been setup.

- **6-20607 P_set_iv must be executed before P_rsweep_miv.**

  p_set_iv must be executed before p_rsweep_miv.

- **6-20608 P_set_lsearch or P_set_bdv_search must be executed before P_search_iv.**

  p_set_lsearch or p_set_bdv_search must be executed before p_search_iv.

- **6-20609 P_set_lsearch or P_set_bdv_search must be executed before P_rsearch_iv.**

  p_set_lsearch or p_set_bdv_search must be executed before p_rsearch_iv.

- **6-20610 P_rsearch_stop is invalid. No parallel search measurement is executed.**

  p_rsearch_stop is invalid. No multi-channel synchronous parallel search measurement is executed.

- **6-20611 The number of array elements for p_set_sync must be less than or equal to the number of source ports specified by p_set_iv or p_set_lsearch.**

  The number of array elements for p_set_sync must be less than or equal to the number of source ports specified by p_set_iv or p_set_lsearch.

- **6-20612 Invalid parameter value for post mode. It must be 1 or 2.**

  The mode value of p_set_post_mode must be 1 or 2.

- **6-20613 Invalid parameter value for stop mode. It must be 1, 2, 3, or 4.**

  The mode value of p_set_stop_mode must be 1, 2, 3, or 4.

- **6-20614 P_rsweep_stop is invalid. No parallel sweep measurement is executed.**

  p_rsweep_stop is invalid. No multi-channel synchronous parallel sweep measurement is executed.

- **6-20615 P_set_iv, P_set_lsearch or P_ivt_set_iv must be executed before P_set_post_mode.**

  p_set_lsearch, p_set_iv, or p_ivt_set_iv must be executed before p_set_post_mode.

- **6-20616 P_set_iv, P_set_lsearch or P_ivt_set_iv must be executed before P_set_stop_mode.**

  p_set_lsearch, p_set_iv, or p_ivt_set_iv must be executed before p_set_stop_mode.

- **6-20617 P_set_vth must be executed before P_measure_vth.**

  p_set_vth must be executed before p_measure_vth.

- **6-20618 Sense ports must be all different.**

  Sense ports for p_set_bdv_search, p_set_lsearch, p_set_vth, or p_sweep_miv must be all different.

- **6-20619 P_set_skip must be executed with single sense port in one group.**

  p_set_skip cannot be executed if the multiple measurement ports are assigned to a source port or a synchronous port.

- **6-20620 P_ivt_set_iv must be executed before P_ivt_status.**

  p_ivt_set_iv must be executed before p_ivt_status.

- **6-20621 Invalid value for post mode. It must be 1 or 3.**

  Post mode for sampling measurement must be 1 or 3.

- **6-20622 Invalid value for group ID on multi-channel measurement. It must be 1 to 8, or 0 for clear.**

  Port group ID for multi-channel measurement must be 1 to 8, or 0 for clear the group setting.

- **6-20623 Multiple group IDs are assigned to an SMU. An SMU cannot belong to multiple port groups.**

  An SMU can belong to one port group.

- **6-21001 Specified pin is not connected to any ports.**

  Pins specified by disable_port function are not connected to any port. Specify the pin number of a pin board that is connected to a port.

- **6-21002 SMU output voltage incompatible with set voltage range.**

  Wrong value was set for the SMU output voltage and voltage range. Set the values properly.

- **6-21003 SMU output current incompatible with set current range.**

  Wrong value was set for the SMU output current and current range. Set the values properly.

- **6-22001 CMU bias voltage must be –40 to 40 Vdc.**

  Wrong value was specified for the bias voltage of CMU. The value must be 0 to ±40 Vdc.

- **6-22002 CMU bias voltage must be –2, –1.5, 0, 1.5, or 2 Vdc.**

  Wrong value was specified for the bias voltage of CMU. The value must be 0, ±1.5, or ±2 Vdc if CMU is 4284A without option 001.

- **6-22003 CMU integration time mode setting must be 1, 2, or 3.**

  Invalid parameter was set for the integration time of set_cmu84 function. The value must be 1 (short), 2 (medium) or 3 (long).

- **6-22004 CMU signal level must be 0.005 V or more.**

  Too small value was set for the signal level by set_cmu84 function. The value must be greater than 0.005 V.

- **6-22005 CMU signal level must be 20.0 V or less.**

  Too large value was set for the signal level by set_cmu84 function. The value must be 20 V or less.

- **6-22006 CMU signal level must be 2.0 V or less.**

  Too large value was set for the signal level by set_cmu84 function. The value must be 2 V or less if CMU is 4284A without option 001.

- **6-22007 CMU frequency must be 19.8 Hz or more.**

  Too small value was set for the measurement frequency of CMU. The value must be 19.8 Hz or more.

- **6-22008 CMU frequency must be 1 MHz or less.**

  Too large value was set for the measurement frequency of CMU. The value must be 1 MHz or less.

- **6-22009 CMU is defective or not installed.**

  CMU is not installed in the 4080, or may be defective. Execute hp4070 -login command and check the configuration of the 4080. If necessary, do the diagnostics.

- **6-22010 There is no capacitance and conductance data in buffer.**

  Capacitance/conductance measurement was not executed properly. CMU may be defective. Do the diagnostics.

- **6-22011 Set_cv84 must be executed before Sweep_cv84.**

  No setup for sweep_cv84 function. Execute set_cv84 function before executing sweep_cv84 function.

- **6-22012 Number of C-V sweep steps must be 2 to 1001.**

  Number of steps for set_cv84 function is out of range. The value must be 2 to 1001.

- **6-22013 Hold time and delay time must be 0.0 to 650.0 sec.**

  Hold time or delay time value of set_cv84 function is out of range. The value must be 0.0 to 650.0.

- **6-22014 4284A option 001 is not present.**

  C-V sweep measurement needs 4284A equipped with the option 001. Execute hp4070 -login command and check the configuration of the 4080.

- **6-22101 HSCMU DC bias voltage must be –10 to 10V.**

  The specified DC bias voltage setting is out of range for HSCMU. Specify a DC voltage from –10 to 10 V.

- **6-22102 Integration value for SHORT integration mode of HSCMU must be 0, or 1 to 75.**

  The specified integration value for SHORT integration mode of HSCMU is out of range. Specify an integer value of 0, or 1 to 75.

- **6-22103 Integration value for LONG integration mode of HSCMU must be 0, or 1 to 100.**

  The specified integration value for LONG integration mode of HSCMU is out of range. Specify an integer value of 0, or 1 to 100.

- **6-22104 Test signal level of HSCMU must be more than or equal to –100mV.**

  The specified test signal level of HSCMU is out of range. Specify a signal level from –100 to 100 mV.

- **6-22105 Test signal level of HSCMU must be less than or equal to 100mV.**

  The specified test signal level of HSCMU is out of range. Specify a signal level from –100 to 100 mV.

- **6-22106 Test signal level of HSCMU must be 10, 30, 50, or 100mV.**

  The specified test signal level value of HSCMU is invalid.

  Specify 10, 30, 50, or 100 mV for the test signal level of HSCMU.

  Or, if you set the parameter rounding mode to on by using pround_cmu function, you can specify any voltage from –100 and 100 mV and the test signal level is automatically set to a proximate valid level.

- **6-22107 Test frequency of HSCMU must be more than or equal to 1kHz.**

  The specified measurement frequency of HSCMU is out of range. Specify a frequency from 1 kHz to 2 MHz.

- **6-22108 Test frequency of HSCMU must be less than or equal to 2MHz.**

  The specified measurement frequency of HSCMU is out of range. Specify a frequency from 1 kHz to 2 MHz.

- **6-22109 Invalid test frequency of HSCMU is specified.**

  The specified measurement frequency of HSCMU is invalid.

  Specify a valid frequecy of HSCMU. Or, If you set the parameter rounding mode to on by pround_cmu function, you can specify any frequency from 0 to 2 MHz and the measurement frequency is automatically set to an proximate valid frequency.

- **6-22111 Set_cv must be executed before Sweep_cv.**

  Before executing sweep_cv, execute set_cv to set the sweep parameters for C-V measurement.

- **6-22151 Set_zf must be executed before Sweep_zf or Get_freq_step.**

  Before executing sweep_zf or get_freq_step, set_zf must be executed to set the sweep parameters for Z/θ-f measurement.

- **6-22152 Set_cvf must be executed before Sweep_cvf.**

  Before executing sweep_cvf, set_cvf must be executed to set the sweep parameters for C/G-V-f measurement.

- **6-22153 Size of frequency array must be greater than or equal to 1.**

  Specify multiple measurement frequencies for set_cvf.

- **6-22177 The value of C or G parameter of Conv_ztr must be other than 0.**

  Specify a capacitance or conductance value other than 0 for conv_ztr.

- **6-22178 The value of frequency parameter of Conv_ztr must be other than 0.**

  Specify a frequecy other than 0 for conv_ztr.

- **6-22179 The value of CV sweep mode must be 1 or 3.**

  Specify 1 (LINEAR-V) or 3 (LINEAR_V_DBL) for CV sweep mode.

- **6-22180 Cannot call Status_cvf during a sweep measurement.**

  status_cvf cannot execute during sweep measurement by IDP.

- **6-22200 Invalid value for sampling mode. It must be 1 to 7.**

  Specify 1 (linear), 2 (log, 10 data/decade), 3 (log, 25 data/decade), 4 (log, 50 data/decade), 5 (log, 100 data/decade), 6 (log, 250 data/decade), or 7 (log, 500 data/decade) for sampling mode.

- **6-22201 Invalid value for hold time before bias force. It must be 0 to 655.35 if log sampling mode, or it must be 0 to 655.35, or -1e-6 to -9e-3 if linear sampling mode.**

  If linear sampling mode, specify the value from 0 to 655.35, or from -1E-6 to -9E-3.

  If logarithm sampling mode, specify the value from 0 to 655.35.

- **6-22202 Invalid value for sampling interval. It must be 1e-6 to 65.535 if linear sampling mode, or 2e-3 to 65.535 if log sampling mode.**

  If linear sampling mode, specify the value from 1E-6 to 65.535.

  If logarithm sampling mode, specify the value from 2E-3 to 65.535.

- **6-22203 Invalid value for sampling count. It must be 1 to 100001 if linear sampling mode, or 1 to 11*(number of data per decade)+1 if log sampling mode.**

  If linear sampling mode, specify the value from 1 to 100001 on C or 32767 on BASIC.

  If logarithm sampling mode, specify the value from 1 to 11×(number of data per decade)+1.

- **6-22204 Invalid value for hold time before base force. It must be 0 to 655.35.**

  Specify the value from 0 to 655.35.

- **6-22210 Invalid value for stop mode of sampling measurement. It must be 1 or 2.**

  Stop mode for sampling measurement must be 1 or 2.

- **6-22211 Force mode must be 1 (voltage force) or 2 (current force).**

  Force mode of bias source for sampling measurement must be 1 (voltage force) or 2 (current force).

- **6-23001 DVM is defective or not installed.**

  DVM is not installed in the 4080, or may be defective. Execute hp4070 -login command and check the configuration of the 4080. If necessary, do the diagnostics.

- **6-23002 DVM is not set to DC voltage measurement mode.**

  DVM is not in the DC voltage measurement mode. Set the mode correctly using front panel or GPIB command. Or DVM may be defective. If necessary, do the diagnostics.

- **6-23003 DVM GPIB address is not set or invalid.**

  Address3458 command was executed even though the DVM is not installed in the 4080. Confirm the measurement program.

- **6-23100 Failed to call igpibbusstatus().**

  GPIB interface card may be defective.

- **6-24001 Testhead Power Fail state.**

  Testhead is turned off. Turn on the testhead. If testhead cannot turn on, check the testhead power supply.

- **6-24002 Testhead Emergency state.**

  Testhead is in the emergency status. Do the diagnostics.

- **6-24100 TIS for multi-channel synchronous parallel measurement function is not available on 4070 series parametric testers.**

  The 4080 does not support TISs for the multi-channel synchronous parallel measurement functions.

- **6-25000** *Command* **is UNSUPPORTED.**

  The 4080 system software does not support the command specified by *Command*. Do not use *Command*.

- **6-25001 Improper PGU unit address** *Address***. Enter correct value.**

  The available value depends on the configuration of the PGUs. To get the unit address from the port number, use FNPort or PORT.

- **6-25002 No PGU on the unit address. Enter correct value or check PGU.**

  There is no PGU on the unit address you specified, or the PGU is defective. Confirm the unit address and enter a valid value. Verify the PGU is operating correctly.

- **6-25003 Must specify unit address of PGU disconnected from testhead.**

  You must specify the unit address of the PGU disconnected from the testhead. Execute pgconn4070 and verify that the value in the PG connection file matches the actual PGU connection, then specify the unit address of the PGU disconnected from the testhead.

- **6-25004 Pin number must be 1 to 24 for the HF port. Enter correct value.**

  An invalid pin number was entered. Pin numbers 1 to 24 are available for the HF port you specified.

- **6-25005 Pin number must be 25 to 48 for the HF port. Enter correct value.**

  An invalid pin number was entered. Pin numbers 25 to 48 are available for the HF port you specified.

- **6-25006 Pin number must be 1 to 48 for the THF port. Enter correct value.**

  An invalid pin number was entered. Pin numbers 1 to 48 are available for the THF port you specified.

- **6-25007 Port address must be 20201 to 20206, or 20211 to 20213.**

  An invalid port address was entered. Addresses 20201 to 20206 are available for the HF port, and addresses 20211 to 20213 are available for the THF port. Enter a valid port address.

- **6-25008 Must specify the port connected to the PGU.**

  You must specify the port connected to the PGU. Execute pgconn4070 and verify that the value in the PG connection file matches the actual PGU connection, then specify the port connected to the PGU.

- **6-25009 Only CH1 is available for the PGU in 3-level pulse output mode.**

  The PGU specified is in the 3-level pulse output mode. Only CH1 is available for the PGU. Use CH1.

- **6-25010 Only CH1 is available for the PGU in 3-level pulse output mode.**

  The PGU specified is in the 3-level pulse output mode. Only CH1 is available for the PGU. Use CH1.

- **6-25011 Improper port address. Enter correct value.**

  Improper port address value is specified. You must specify the correct value for port connected to the PGU.

- **6-25012 Mode value must be 0 or 1. Enter correct value.**

  set_sr_control TIS error. The mode parameter value must be 0 (independent) or 1 (synchronous).

- **6-25013 Port number must be PSC1, 2, or PSO1 to 5.**

  set_sr_mode TIS error. The port address parameter must be PSC1, 2, or PSO2 to 5.

- **6-25014 Mode value must be 0 or 1. Enter correct value.**

  set_sr_mode TIS error. The mode parameter value must be 0 (Normally Closed) or 1 (Normally Open).

- **6-25015 Port number must be PGU address, PSC1, 2, or PSO1 to 7.**

  connect_sr TIS error. The port address parameter must be PGU address, PSC1, 2, or PSO1 to 7.

- **6-25016 PGU must be connected to the pulse switch control port 1 or 2.**

  connect_sr TIS error. When a PGU unit address is specified for the port parameter the PGU must be connected to pulse switch control port 1 or 2. Specify a valid unit address and verify the PGU is connected correctly.

- **6-25017 To use Connect_sr/Switch_sr, independent mode must be set by Set_type_pg.**

  set_sr_pg TIS error. To control switching by using connect_sr or switch_sr the mode parameter value must be 1 (independent mode).

- **6-25018 Port address must be HF, THF, AUX, PSI, or PGU address.**

  An illegal port address was specified. Port address must be HF, THF, AUX, PSI, or PGU address.

- **6-25019 Must specify the port connected to the PGU.**

  You must specify the port connected to the PGU. Execute pgconn4070 and verify that the value in the PG connection file matches the actual PGU connection, then specify the port connected to the PGU.

- **6-25020 Only CH1 is available for the PGU in 3-level pulse output mode.**

  set_type_pg TIS error. The PGU you specified is in the 3-level pulse output mode. Only CH1 is available for the PGU. Use CH1.

- **6-25021 Mode value must be 1 to 3. Enter correct value.**

  set_type_pg TIS error. The mode parameter value must be 1 (2-level pulse), 2 (3-level pulse by one PGU), or 3 (3-level pulse by multiplexer). Enter a valid mode value.

- **6-25022 Only CH1 is available for the PGU in 3-level pulse output mode.**

  set_type_pg TIS error. Mode=2 is available only for CH1 of the PGU. Use CH1.

- **6-25024 PGU must have two channels for 3-level pulse output.**

  The PGU you specified has only one channel. For 3-level pulse output use a PGU with two output channels.

- **6-25025 Must specify the port connected to the PGU.**

  You must specify the port connected to the PGU. Execute pgconn4070 and verify the value in the PG connection file matches the actual PGU connection, then specify the port connected to the PGU.

- **6-25026 Only CH1 is available for the PGU in 3-level pulse output mode.**

  set_level_pg TIS error. The PGU specified is in the 3-level pulse output mode. Only CH1 is available for the PGU. Use CH1.

- **6-25027 Invalid port for using pulse switch.**

  set_level_pg TIS error. The PGU specified is in the 3-level pulse output mode. Pulse switch is available for the port set to the 2-level pulse or 3-level pulse by the multiplexer. Enter a valid port value.

- **6-25028 Port address must be HF, THF, AUX, PSI, or PGU address.**

  An improper port address was specified. Port address must be HF, THF, AUX, PSI, or PGU address.

- **6-25029 Must specify the port connected to the PGU.**

  You must specify the port connected to the PGU. Execute pgconn4070, and verify the value in the PG connection file matches the actual PGU connection, then specify the port connected to the PGU.

- **6-25030 Only CH1 is available for the PGU in 3-level pulse output mode.**

  The PGU specified is in the 3-level pulse output mode. Only CH1 is available for the PGU. Use CH1.

- **6-25031 PGU must have two channels for 3-level pulse output.**

  The PGU specified has only one channel. Use a PGU with two output channels for 3-level pulse output.

- **6-25032 Invalid port for using pulse switch.**

  set_time_pg TIS error. The PGU specified is in the 3-level pulse output mode. Pulse switch is available for the port which is set to the 2-level pulse or 3-level pulse by the multiplexer. Enter a valid port value.

- **6-25033 Specify width2 parameter value.**

  set_time_pg TIS error. A width2 parameter must be specified. Enter a valid value.

- **6-25034 Delay2 parameter value must equal width1 plus delay1 or more.**

  set_time_pg TIS error. The value delay2 must be equal to or greater than width1 + delay1 if a multiplexer of the pulse switch is used to create 3-level pulse. Set the applicable values. Or do not use the multiplexer.

- **6-25035 Mode value must be 0, 1, or 2. Enter correct value.**

  set_sr_pg TIS error. The mode parameter value must be 0 (reset), 1 (independent), or 2 (synchronous with pulse). Enter a valid mode value.

- **6-25036 Specify width parameter value, if mode is set to 2.**

  set_sr_pg TIS error. A width parameter must be specified. Enter a valid value.

- **6-25037 Must specify the PGU connected to the pulse switch control port.**

  set_sr_pg TIS error. Connect the PGU to the pulse switch control port correctly and execute pgconn4070. Verify the value in the PG connection file matches the actual PGU connection, then specify the PGU connected to the pulse switch control port.

- **6-25038 period value too short.**

  force_pg and prep_pg TIS error. The period parameter must be more than or equal to the sum of the timing parameters defined by set_time_pg. Enter a valid value.

- **6-25039 Cannot execute the TIS during pulse output.**

  During pulse output the 4080 reports this error by the TIS which controls the PGU setup or the switching matrix PGU path. The TIS stops the program with this error, and also stops pulse output.

- **6-25040 Count value must be 0 to 10000000.**

  force_pg or start_pg TIS error. The count parameter value must be from 0 to 10000000. If the count value is 1 to 10000000, force_pg or start_pg starts the pulse output, and waits for the last count of the pulse. For a count value of 0, the force_pg or start_pg triggers a continuous pulse output and finishes immediately. The pulse output continues until a stop_pg TIS is received. During continuous pulse output the 4080 can receive TIS, but causes error 6-25039. See 6-25039.

- **6-25041 Install or turn on master PG.**

  Cannot detect the master PG. Install or turn on the master PG. The master PG is the pulse generator unit which has the PG logical number 1. See the configuration file (/etc/opt/hp4070/config/1) to specify the master PG. This file contains the PG name, PG logical number, bus address, and GPIB address.

- **6-25042 Time accuracy mode must be 0 or 1.**

  set_mode_pg TIS error. The mode parameter value must be 0 (normal) or 1 (pattern).

- **6-25046 Do not enter Set_level_pg for the pulse switch control port.**

  You cannot set the pulse output level of the PGU connected to the pulse switch control port. Do not enter set_level_pg.

- **6-25047 Do not enter Set_type_pg for the pulse switch control port.**

  You cannot set the pulse mode of the PGU connected to the pulse switch control port. Do not enter set_type_pg.

- **6-25048 Must specify the pulse switch control port connected to the PGU.**

  You must specify the pulse switch control port connected to the PGU. Execute pgconn4070 and verify the value in the PG connection file matches the actual PGU connection, then specify the pulse switch control port connected to the PGU.

- **6-25052 PGU** (*N*) **is defective or not connected . Check PGU and enter correct port.**

  You must specify the port connected to to the PGU. Specify the correct port, and verify the PGU is installed and working correctly.

- **6-25053 Cannot execute** *Command Name* **during pulse output.**

  During pulse output, the 4080 causes this error by the TIS which controls the PGU setup or the switching matrix PGU path. The TIS stops the program with this error, and also stops pulse output.

- **6-25056 PGU output voltage exceeds the limit. Check PG*XX* output amplitude.**

  Check the amplitude of the PG*XX* output. The limit is -19 V. If necessary, reduce the amplitude of the signal. If this does not correct the error, contact the nearest Keysight Technologies Sales and Service office.

- **6-25057 Mode value must be 0 to 2. Enter correct value.**

  switch_sr TIS error. Mode must be 0, 1, or 2. Enter a valid mode number.

- **6-25058 Mode value must be 0 to 2. Enter correct value.**

  connect_sr TIS error. Mode must be 0, 1, or 2. Enter a valid mode number.

- **6-25059 Port number must be PSC1, 2, or PSO1 to 7.**

  switch_sr TIS error. The port address parameter must be PSC1, 2, or PSO1 to 7.

- **6-25060 Pulse period is out of range. The value must be 999 sec or less.**

  Set the valid value to the pulse period. It must be between 0 and 999 s. If 0 is specified, the minimum period is set automatically.

- **6-25501 PG Error.**

  An error occurred in the pulse generator unit. The pulse generator will provide detailed error messages.

- **6-25502 PG*N* pulse level is out of range.**

  The pulse level value is out of range.

- **6-25503 Load impedance value must be 2.5 ohm to 999 kilohm.**

  The load impedance value must be 2.5 ohm to 999 kilohm. Enter a valid impedance value.

- **6-25505 Amp*N* value is out of range.**

  set_level_pg TIS error. The Amp*N* parameter value is out of range.

- **6-25506 3-level pulse output value (Amp*N*) is out of range.**

  set_level_pg TIS error. The Amp*N* parameter value is out of range for the 3-level pulse output port (set_type_pg, mode=2).

- **6-25507 Sum of 3-level pulse output value is out of range.**

  set_level_pg TIS error. The sum of Amp1 and Amp2 is out of range for the 3-level pulse output port (set_type_pg, mode=2).

- **6-25508 Invalid setup combination of Amp and Base.**

  set_level_pg TIS error. Cannot set the pulse level due to an invalid combination of the Amp value and Base value.

- **6-25509 Pulse width (*Value*) is out of range.**

  set_time_pg TIS error. Enter a valid width value.

- **6-25510 Delay time (*Value*) plus skew is out of range.**

  set_time_pg TIS error. Enter a valid delay time. The skew between the PG outputs can cause this error.

- **6-25511 Invalid pulse leading time (*Value*).**

  set_time_pg TIS error. Enter a valid leading-edge transition time.

- **6-25512 Invalid pulse trailing time (*Value*).**

  set_time_pg TIS error. Enter a valid trailing-edge transition time.

- **6-25513 Leading time and trailing time must be in the same range.**

  set_time_pg TIS error. Both the leading and trailing time must be in the same range.

- **6-25514 PG Error. Cannot change arming source.**

  Cannot change the arming source. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25516 PG Error. Cannot measure trigger period.**

  The pulse generator cannot measure the trigger period. The trigger cable may be disconnected or defective. Or the PGU may be defective. Verify the trigger cable connection and the operation of the PGU.

- **6-25517 Cannot open file (errno** = *Error No.***) :** *File name*

  Cannot open the specified file. Check the file name, permission, and so on.

- **6-25518 Cannot write file (errno** = *Error No.***)**

  Cannot write the file. Check the file name, permission, and so on.

- **6-25519 Cannot read file (errno** = *Error No.***)**

  Cannot read the specified file. Check the file name, permission, and so on.

- **6-25520 PG setup file revision mismatch.**

  The PG setup file does not match the present revision.

- **6-25521 Configuration mismatch.**

  The current configuration does not match the configuration written in the PG setup file. The PG setup file is the file saved by save_pg TIS.

- **6-25522 PG (*N*) Error. No response to combined channel query.**

  No response from the PGU for the combined channel query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25523 PG (*N*) Error. No response to arming source query.**

  No response from the PGU for the arming source query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25524 PG (*N*) Error. No response to period source query.**

  No response from the PGU for the period source query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25525 PG (*N*) Error. No response to trigger count query.**

  No response from the PGU for the trigger count query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25527 PG (*N*) Error. No response to period query.**

  No response from the PGU for the period query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25528 PG (*N*) Error. No response to switch query.**

  No response from the PGU for the switch query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25529 PG (*N*) Error. No response to rload query.**

  No response from the PGU for the rload query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25530 PG (*N*) Error. No response to delay query.**

  No response from the PGU for the delay query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25531 PG (*N*) Error. No response to width query.**

  No response from the PGU for the width query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25532 PG (*N*) Error. No response to leading edge query.**

  No response from the PGU for the leading edge query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25533 PG (*N*) Error. No response to trailing edge query.**

  No response from the PGU for the trailing edge query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25534 PG (*N*) Error. No response to complement query.**

  No response from the PGU for the complement query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25535 PG (*N*) Error. No response to voltage level query.**

  No response from the PGU for the voltage level query. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25536 PG Error. Cannot change period source.**

  Cannot change period source. The PGU may be defective or it may be a TIS fatal error. Verify the PGU is operating correctly.

- **6-25538 Can't open file. filename:** *File*

  Could not open the file specified by *File*. Confirm the file name and permissions of the file.

- **6-25539 Can't load the offline configuration file. File format incorrect.**

  Could not read the offline configuration file because the file format is incorrect. Check the format of the file you loaded.

- **6-25540 Can't save the offline configuration file.**

  Could not save the offline configuration file. Check the permissions of the file and directory.

- **6-25542 Leading edge exceeds the width.**

  Set the valid value to the leading edge transition time. It must be less than or equal to pulse width $\times$ 0.8.

- **6-25543 Load impedance value must be 2.5 ohm to 1000000 ohm.**

  Set the valid value to the load impedance. It must be between 2.5 Ω and 1 MΩ.

- **6-25901 [F_ROM] Port address (*Terminal Name*) must be unique.**

  Each terminal must be set to a unique port address. Set each terminal to a unique port address.

- **6-25902 [F_ROM] Relay mode must be enabled.**

  Relay mode must be enabled to use pulse switch. Execute F_set_relay_md(1) to enable the relay mode.

- **6-25903 [F_ROM] Port (*Terminal Name*) must be connected to pulse switch output.**

  A port must be connected to the pulse switch output. Connect the port and specify the correct port.

- **6-25904 [F_ROM] Port (*Terminal Name*) must be connected to pulse switch output 1, 6, or 7.**

  A port must be connected to the pulse switch output 1, 6, or 7. Connect the port and specify the correct port.

- **6-25905 [F_ROM] Port (*Terminal Name*) must be connected to pulse switch output 2, 3, 4, or 5.**

  A port must be connected to the pulse switch output 2, 3, 4, or 5. Connect the port and specify the correct port.

- **6-25906 Invalid count (*Value*). The value must be 1 to 1000000000.**

  The count parameter value must be 1 to 1000000000. Enter a valid value.

- **6-25907 [F_ROM] PGU (*Terminal Name*) must be unique.**

  Each terminal must be connected to a unique PGU. Connect each terminal to a unique PGU.

- **6-25908 [F_ROM] Invalid port address for** *Terminal Name*.

  F_set_fport PARA error. The port address must be 20201 to 20206 for HF port, and 20211 to 20213 for THF port, and 20101 to 20108. Enter a valid port address.

- **6-25909 [F_ROM] Invalid PGU connection for** *Terminal Name*.

  The PGU must be connected to the specified port through the pulse switch. Connect the PGU to the pulse switch input and specify a valid port.

- **6-26100 Port must be connected to VNA.**

  Connect the specified pin to VNA before setting measurement conditions of VNA.

- **6-26105 Specified sweep type for VNA is invalid.**

  Specify a correct sweep type. Log (–1) or Linear (1).

- **6-26106 Specified Deembedding mode for VNA is invalid.**

  Specify a correct deembedding mode. Off (0), Open (1), or Short (2).

- **6-26108 Specified port is not VNA.**

  A port other than VNA was specified. Specify a VNA port.

- **6-26109 DCBias does not present.**

  DC bias source does not exist or may be defective. Check the system configuration or do troubleshooting.

- **6-26110 VNA port can not be connected to multiple pins.**

  Only one pin can be connected to a VNA port.

- **6-26111 VNA stimulus must be specified before any measurement command.**

  Execute a function for setting VNA stimulus condition before VNA measurement function.

- **6-26112 Two VNA ports must be in the same instrument.**

  Specify two ports of the same VNA.

- **6-26113 VNA and DCBias must be connected.**

  Connect the DC bias source to the VNA to use the RF subsystem.

- **6-26114 VNA RFCV Sweep condition must be specified before measurement.**

  Execute a function for setting RF CV sweep condition before RF CV measurement function.

- **6-26117 Cannot read measured data from VNA instrument.**
  2-26118 VNA measured frequency is illegal.

  The VNA may have a problem. Do troubleshooting.

- **6-26119 VNA measurement is not executed yet.**

  Measurement by VNA is previously required.

- **6-26120 Allowable range of VNA (PNA) band width is from 1 Hz to 40 kHz.**

  Specify a valid IF bandwidth value.

- **6-26121 Allowable range of VNA (PNA) sweep points is from 2 to 16001.**

  Specify a valid sweep points value.

- **6-26122 Allowable range of VNA (PNA) frequency is from 10 MHz to 20 GHz.**

  Specify a valid frequency value.
- **6-26123 VNA (PNA) Log sweep requires at least 2 octave span.**

  A minimum of two octave span is required for VNA log sweep.
- **6-26124 Allowable range of VNA (PNA) Average is from 0 to 1024.**

  Specify a valid average value.
- **6-26125 RFMatrix does not exist.**

  The system does not have a RF matrix. Only direct RF outputs are available.
- **6-26127 Allowable range of VNA (PNA) Power is from** *min* **to** *max*.

  Specify a valid VNA power level.
- **6-26128 Allowable range of VNA Sweep CV points is from 2 to 1001.**

  Specify a valid sweep points value.
- **6-26129 Same VNA ports are specified. Select different two ports.**

  Specify two different VNA ports for this function.
- **6-26130 Allowable range of DCBias Sweep Voltage is from -7 V to 7 V.**

  Specify a DC bias sweep voltage value within the allowable range.
- **6-26131 Allowable range of DCBias Sweep hold time is from 0 s to 650 s.**

  Specify a DC bias sweep hold time within the allowable range.
- **6-26132 Allowable range of DCBias Sweep delay time is from 0 s to 650 s.**

  Specify a DC bias sweep delay time within the allowable range.
- **6-26133 Specified file for VNA ISS std. does not exist.**

  The specified file for calibration standard values does not exist. Check the file name.
- **6-26134 VNA Calibration Coefficient calculation Fail.**

  Failed to calculate VNA calibration coefficient. Check data and conditions.
- **6-26135 Open structure measured data must be specified for de-embedding.**

  Specify open structure measured data.

- **6-26136 Invalid DCBias voltage or compliance is specified.**

  Specify a valid voltage or compliance value for the DC bias source.

- **6-26137 Invalid DCBias current or compliance is specified.**

  Specify a valid current or compliance value for the DC bias source.

- **6-26138 Invalid DCBias voltage range is specified.**

  Specify a valid voltage range value for the DC bias source.

- **6-26139 Invalid DCBias current range is specified.**

  Specify a valid current range value for the DC bias source.

- **6-26140 Invalid characteristic Impedance is specified.**

  Specify a valid characteristic impedance value.

- **6-26147 VNA Error. Receiver Overload.**
  6-26148 VNA Error. PLL Unlocked.
  6-26149 VNA Error. Power Unleveled.

  VNA error occurred during measurement. Check measurement conditions or do troubleshooting.

- **6-26150 Specified calibration type for VNA is invalid.**

  Specify a valid calibration type for VNA.

- **6-26151 VNA Calibration is not initialized.**

  Initialize VNA calibration data/condition previously.

- **6-26152 Invalid VNA Calibration standard.**

  The calibration standard may have a problem. Check the calibration standard.

- **6-26153 Specify a valid COEF data file name for VNA.**
  6-26154 Cannot open COEF data file for VNA: *error*
  6-26155 Incorrect format is found in COEF data file for VNA: *error*

  The specified calibration coefficient data file does not exist or has a problem. Check the file name or the file.

- **6-26156 VNA Cal algorithm must be SOL or SOLT.**

  Specify SOL(=1) or SOLT(=2) as calibration algorithm.

- **6-26157 Fail to load VNA Cal COEF data.**

  Failed to load the calibration coefficient data. Check the file.

- **6-26159 Fail to store VNA Cal COEF data.**

  Failed to store the calibration coefficient data. Check the directory permission and specified conditions.

- **6-26168 Specified port number [%d] isn't related with RF function.**

- **6-26169 Specified port is invalid because VNA selftest failed.**

  Specified RFU port is invalid because of a fail on VNA selftest.

- **6-26170 Specified port is invalid because DC Bias selftest failed.**

  Specified RF DC bias port is invalid because of a fail on DC bias selftest.

- **6-27001 SPA is defective or not installed.**

  The SPA is not installed or defective. Confirm the configuration or troubleshoot the SPA.

- **6-27002 Minf must be smaller than Maxf.**

  The start frequency must be smaller than the stop frequency.

- **6-27003 Minf is too small.**

  The start frequency is too small. The value must be equal to or greater than 0 [Hz].

- **6-27004 Maxf is too large.**

  The stop frequency is too large. The maximum stop frequency is 1.5E+9 [Hz].

- **6-27005 Frequency (*freq*) is too small for Resolution Band Width.**

  Specified *freq* is too small for resolution bandwidth. The minimum resolution bandwidth is 1E+3 [Hz].

- **6-27006 Frequency (*freq*) is too large for Resolution Band Width.**

  Specified *freq* is too large for resolution bandwidth. The maximum resolution bandwidth is 5E+6 [Hz].

- **6-27007 Frequency (*freq*) is too small for Fcounter res.**

  Specified *freq* is too small for frequency counter resolution. The minimum frequency counter resolution is 0 [Hz].

- **6-27008 Frequency (*freq*) is too large for Fcounter res.**

  Specified *freq* is too large for frequency counter resolution. The maximum frequency counter resolution is 1E+5 [Hz].

- **6-27009 Number of inverters is necessary to calculate propagation delay time.**

  The number of inverters must be specified to calculate propagation delay time.

- **6-27111 Cannot Disconnect (NO RFMatrix).**

  The specified pin cannot be disconnected because there is no RF matrix.

- **6-28000 SPGU Calibration Fail.**

  SPGU failed the calibration. Do troubleshooting.

- **6-28001 SPGU is defective or not installed.**
  6-28002 SPGU Access Fail.
  6-28003 SPGU System Fail.

  The SPGU is not installed or may be defective. Confirm the configuration or troubleshoot the SPGU.

- **6-28011 PG Error:** *error*

  An error occurred in the SPGU. Detailed error message will be provided.

- **6-28012 PG*N* level value is out of range.**

  The pulse level value is out of range.

- **6-28013 Invalid pulse leading time (*Value*). The value must be 20.0 nsec to 400 msec.**

  Enter a valid transition value.

- **6-28014 Invalid pulse trailing time (*Value*). The value must be 20.0 nsec to 400 msec.**

  Enter a valid transition value.

- **6-28015 Load impedance value must be 0.1 ohm to 1.0 Mohm.**

  Enter a valid impedance value.

- **6-28016 Pulse width (*Value*) is out of range.**

  Enter a valid range value.

- **6-28017 Delay time (*Value*) is out of range.**

  Enter a valid range value.

- **6-28018 PG*N* pulse level is out of range.**

  Enter a valid range value.

- **6-28019 PG*N* Error: No response to source function query.**

  No response from the SPGU for the source function query. The SPGU may be defective or it may be a TIS fatal error. Verify the SPGU is operating correctly.

- **6-28020 Pulse period (*Period*) is out of range.**

  Set the correct value.

- **6-28021 Count value must be 0 to 1000000000.**

  Enter a value within the range.

- **6-28022 Cannot open file (errorno = *error*)**
  6-28023 Cannot read file (errorno = *error*)

  Check the file name or permission.

- **6-28024 Port address in Alwg file must be HF, THF, AUX, PSI, PSC or PGU address.**

  Wrong port address is specified in Alwg file. Check the file.

- **6-28025 Port address in Alwg file must be connected to the SPGU.**

  Port address specified in Alwg file is not connected to SPGU. Check the connection and file.

- **6-28026 Port address in Alwg file is not installed in this system.**

  Port address specified in Alwg file does not exist. Check the system configuration and the file.

- **6-28027 Illegal string found in Alwg file.**

  The Alwg file has a problem. Check the Alwg file.

- **6-28028 *Parameter* in Alwg file is out of range.**

  The *Parameter* is out of range. Use a valid range value.

- **6-28029 Cannot execute the TIS in PG mode.**

  The TIS function cannot be executed in PG mode. Change the mode to ALWG.

- **6-28030 Cannot execute the TIS in ALWG mode.**

  The TIS function cannot be executed in ALWG mode. Change the mode to PG.

- **6-28031 Mode value must be 0 or 1. Enter correct value.**

  Specify 0 or 1 as SPGU's mode.

- **6-28032 Delay must be 0 sec to 9.9 sec.**

  Use a valid range value.

- **6-28033 No TIS part found in the Alwg file.**

  The Alwg file has a problem. Check the file.

- **6-28034 Absolute Voltage level must be over 100 mV.**

  Specify 100 mV or higher absolute value.

- **6-30002 SMU Calibration Fail.**

  SMU failed the calibration. Do the diagnostics, and do troubleshooting.

- **6-30003 Automatic SMU Calibration Fail.**

  SMU failed the automatic calibration. Do the diagnostics, and do troubleshooting.

- **6-30004 Automatic SMU Calibration Pass.**

  SMU passed the automatic calibration.

- **6-30006 Automatic CMU Calibration Fail.**

  HSCMU failed the automatic calibration. Do the diagnostics, and do troubleshooting.

7-xxxxx

- **7-10001 Power Fail occurred.**

  Testhead power fail occurred. Troubleshoot the testhead power supply and the testhead.

- **7-10002 Over Current or Voltage occurred.**

  Over current or over voltage occurred in the testhead. Troubleshoot the testhead.

- **7-10003 Fixture is opened.**

  Prober/Fixture Sense switch of the testhead is open. Close the sense switch of the testhead.

- **7-10004 Interlock is opened.**

  Interlock of the testhead is open. Close the interlock switch of the testhead.

- **7-10006 Hardware Status returned to the normal state.**

  The 4080 hardware status returned to the normal state. The 4080 can be used normally.

- **7-10100 SPGU emergency occurred.**

  Emergency state occurred in the SPGU. Troubleshoot the SPGU.

- **7-10101 SPGU Over Voltage occurred.**

  Over voltage occurred in the SPGU. Troubleshoot the SPGU.

- **7-10102 SPGU Over Current occurred.**

  Over current occurred in the SPGU. Troubleshoot the SPGU.

- **7-10103 SPGU High Temperature occurred.**

  High temperature was detected in the SPGU. Troubleshoot the SPGU.

- **7-10104 SPGU Status returned to the normal state.**

  The SPGU hardware status returned to the normal state. The SPGU can be used normally.

8-xxxxx

- **8-10001 Failed to fork TIS Server (errno =** *Error_No.***).**

  hp4070 command or START program failed to start up the TIS server in fork().

  - If *Error_No.* is 11: Too many processes are running. Kill unnecessary processes or reconfigure kernel to extend maximum number of processes.
  - If *Error_No.* is 12: Too small swap space or memory to execute TIS server. Increase the swap space or memory.

- **8-10002 Failed to exec TIS Server (errno =** *Error_No.***).**

  hp4070 command or START program failed to start up the TIS server in execv().

  - If *Error_No.* is 13: Check the permission and directory path of tis_online or tis_offline file.
  - If *Error_No.* is 249: Too many symbolic links are made for directory path of tis_online or tis_offline file.
  - If *Error_No.* is 8: tis_online or tis_offline file is not set as an executable file. Set the file as an executable file.
  - If *Error_No.* is 12: Too small memory to execute TIS server.
  - If *Error_No.* is 26: tis_online or tis_offline file is opened without write protection.

- **8-11001 TIS server died.**

  hp4070 -shutdown command was executed, or testhead was turned off. Or fatal error occurred in testhead.

- **8-11002 Inconsistent exception (***Error_No.***).**

  TIS server detected illegal error that is not defined as 4080 error condition.

- **8-20001 Invalid parameter.**

  Improper command parameter was specified for hp4070 command. Check the parameter.

- **8-20002 Inconsistent parameters.**

Invalid multiple command parameters were specified for hp4070 command. Check the parameter.

- **8-20003 Incomplete parameter specification.**

  Command parameter defined for hp4070 command is insufficient. Check the parameters.

- **8-20004 Already logged in.**

  You have already logged into the 4080 operating environment.

- **8-20005 No such session (***Error_No.***).**

  You have not logged into the 4080 operating environment yet. No session to log out.

- **8-20006 System busy (Used by session:** *Session_ID***).**

  The 4080 has already been logged into as the on-line mode by another user specified by *Session_ID*. On-line mode allows one user to use the 4080.

- **8-20007 Session ID must be 1 to 999999.**

  Improper session ID was specified for the environmental parameter PCS_SESSION_ID. The value must be 1 to 999999.

- **8-20008 Optimization level setting must be 0, 1, 2 or 3.**

  Improper value was specified for hp4070 -optimize command. Value must be 0, 1, 2, or 3.

- **8-20009 Too many OFF LINE sessions.**

  Too many users logged into the 4080 offline mode. Maximum 5 users can log into the 4080 offline mode.

- **8-20010 Invalid unit on port** *Port_No.***.**

  Invalid measurement unit is assigned for the 4062 port number *Port_No.* on the port map file. Maybe the port map file was edited incorrectly. Define the port number mapping properly.

- **8-20011 Invalid 4070 port selection on 4062 port** *Port_No.***.**

  Improper 4080 port number is assigned for the 4062 port number *Port_No.* on the port map file. Maybe the port map file was edited incorrectly. Define the port number mapping properly.

- **8-20012 Duplicate 4142B slot specification on 4062 port** *Port_No.***.**

  Multiple measurement units are assigned for the 4062 *Port_No.* on the port map file. Only one unit must be defined for one port. Define the port number mapping properly.

- **8-20013 41420A SMU is in slot 1 on 4062 port** *Port_No.***.**

  Cannot assign the 41420A SMU for 4062 port *Port_No.* on the port map file because the 41420A uses two module slots of the 4142B. Define the port number mapping properly.

- **8-20014 Can't read the port map file.** *Reason***.**

  Port map file could not be read because of the reason shown in *Reason*. Check the file name and the permission of the file and directory.

- **8-20015 Can't load the portmap file. File format incorrect.**

  Port map file could not be loaded because the file format is incorrect.

- **8-20016 Can't load the DUMCONFG file. File format incorrect.**

  The 4062UX DUMCONFIG file used for port mapping could not be loaded because the file format is incorrect.

- **8-20017 Can't read the DUMCONFIG file** *Reason***.**

  The 4062UX DUMCONFIG file could not be read because of the reason shown in *Reason*. Check the file name, and the permission of the file and directory.

- **8-20018 Can't save the portmap file.** *Reason***.**

  Port map file could not be saved because of the reason shown in *Reason*. Check the directory name, and the permission of the file and directory.

- **8-21001 Testhead Over Voltage.**

  SMU in testhead is over voltage. Testhead is automatically shut down.

- **8-22001 Insufficient args (Something's wrong).**
  8-22002 Invalid auto cal interval specified (Something's wrong).

  Error occurred in cal4070 command. Internal error.

- **8-22003 Failed at Matrix Mother Board C/G measurement (status: *Status*).**
  8-22004 Failed at Matrix Mother Board offset measurement (status: *Status*).
  8-22005 Failed at Matrix Mother Board R/L measurement (status: *Status*).
  8-22006 Failed at Testhead offset measurement (status: *Status*).

  Improper capacitance compensation data was returned by CMU. Check the measurement cable, or execute the relay test of testhead.

- **8-22008 Invalid configuration; Pin Board *Pin_No.* is not present.**
  8-22009 Invalid configuration; Chuck Connection Pin Board is not present.

  Cannot execute the capacitance compensation measurement, because the pin board specified by *Pin_No.* or the chuck connection pin board is not installed in the testhead, or defective. Check the configuration and install the pin board properly.

- **8-22010 Cannot read the file. The file does not exist or does not have read permission.**

  The specified file cannot be read because the file does not exist or does not have read permission.

- **8-22011 Cannot write the file. The file does not exist. Or the file or the directory does not have write permission.**

  The specified file cannot be written. Because the file does not exist, or the file or specified directory does not have write permission.

- **8-22021 Unable to use the same matrix Pin_No. for CMH and CML. Specify different pin number.**

  Unable to measure compensation data for level 2 capacitance measurement because the same matrix pin number shown as Pin_No. for CMH and CML is specified. Use a different pin number for each port.

- **8-22022 The specified matrix Pin_No. does not exist. Use a valid pin.**

  When executing compensation data measurement for level 2 capacitance measurement, a non-existing matrix pin was specified as Pin_No. Check available pins and use a valid pin number.

- **8-22030 Cannot find available SMU.**

  The system cannot find any SMUs to measure the PGU compensation data. To execute the PGU calibration, at least one SMU must be installed. Check the actual SMU configuration.

- **8-22031 Cannot find available Matrix Pin.**

  The system cannot find any pin boards for the PGU calibration. To execute the PGU calibration, at least one pin board must be installed for each matrix block. Check the actual pin board configuration.

- **8-22032 Can't save pg calibration data into "/etc/opt/hp4070/pgcaldata" file.**

  The compensation data cannot be saved in the /etc/opt/hp4070/pgcaldata file. Check the following items:

  - The /etc/opt/hp4070/pgcaldata must exist.
  - The permission, owner, and group of this file must be -rw-r--r--, “root”, and “sys”, respectively.

- **8-22033 Can't load pg calibration data file "/etc/opt/hp4070/pgcaldata".**

  The cal4070 program cannot load the PGU compensation data correctly because of the wrong data format. Copy the /etc/opt/hp4070/pgcaldata file from the /opt/hp4070/newconfig directory, then execute the cal4070 again.

- **8-23001 Can't open C compensation data file** *File_name* **for loading (errno =** *Error_No.***).**

  Capacitance compensation data file could not be open because of the reason specified by *Error_No.*.

- **8-23002 Can't open C compensation data file** *File_name* **for saving (errno =** *Error_No.***).**

  Capacitance compensation data file could not be saved because of the reason specified by *Error_No.*.

- **8-23003 Can't read C compensation data file** *File_name* **(errno =** *Error_No.***).**

  Capacitance compensation data file could not be read because of the reason specified by *Error_No.*.

- **8-23004 Invalid C compensation data file format.**

  Capacitance compensation data file could not be read because of invalid file format. Check the file name.

- **8-23005 C compensation data file does not match the present Pin Board configuration.**

  Pin board configuration might be changed. Cannot execute the capacitance compensation properly for the present pin board configuration.

- **8-23501 Cannot find HF Matrix board.**
  Check *Port* value in PG connection file.

  The HF matrix board is not installed or it is defective. Edit the PG connection file and delete the PG address set to the specified port (*Port*).

- **8-23502 One PG must be defined for the relative HF and THF ports.**
  Check *Port* value in PG connection file.

  Different PG addresses are defined for the relative HF and THF ports. A single PG address must be defined for the ports. Edit the PG connection file and set a sinlge PG or PSO address, or delete the address set to the specified port (*Port*).

- **8-23503 Invalid PG connection data on *Port*.**
  Specified PG or PSO does not exist.

  An invalid PG or PSO address is defined in the PG connection file. Set the available PG or PSO address to the specified port (*Port*).

- **8-23504 Cannot find Pulse Switch Block *N*.**
  Check *Port* value in PG connection file.

  Pulse switch block 1 or 2 is not installed or is defective. Edit the PG connection file and delete the PG address set to the specified port (*Port*) and to port PSI1/2/3/4 or PSI5/6/7.

- **8-23505 Invalid parameter for pgconn4070.**

  An invalid parameter was set for the pgconn4070 command execution. Set the parameter properly or execute pgconn4070 with no parameter.

- **8-23506 Syntax error in dump command.**

  An improper command is entered in the print dialog box. Enter the proper command.

- **8-23507 Cannot search connection for OFF LINE session.**

  Actual PG connection cannot be searched for offline session.

- **8-24001 Usage:** *Parameter*

  Improper command parameter was specified for setlf command. Specify the command parameter properly.

- **8-24002 Can't open Optical Interface device file (***Reason***).**

  Failed to open the optical interface device file by setlf command because of the reason shown in *Reason*.

- **8-24003 Can't open Optical Interface device file (***Reason***). Perhaps shutting down the system by hp4070 -shutdown is required.**

  Failed to open the optical interface device file by setlf command because the 4080 is used by another user (process ID).

  Execute hp4070 -shutdown before executing setlf command.

- **8-24004 Can't write to the Optical Interface device file (***Reason***).**

  Failed to write the optical interface device file by setlf command because of the reason shown in *Reason*.

- **8-24005 Can't read from the Optical Interface device file (***Reason***).**

  Failed to read the optical interface device file by setlf command because of the reason shown in *Reason*.

- **8-25020 Can't load the offline configuration file. File format incorrect.**

  Could not read the offline configuration file because the file format is incorrect. Check the format of the file you loaded.

- **8-25021 Can't save the offline configuration file.**

  Could not save the offline configuration file. Check the permission of the file and directory.

- **8-27001 Failed to monitor the log file (/var/opt/hp4070/syslog4070). The log file might not exist.**

  The System Management Panel failed to monitor the log file (/var/opt/hp4070/syslog4070). Verify the file exists. Check the permissions of the file or directory.

- **8-27002 Online TIS daemon is not running.**

  The online TIS daemon is not running. Execute the hp4070 -start command.

- **8-27003 Online TIS daemon is already running.**

  Another online TIS daemon is already running. Only one online TIS daemon can run at a time.

- **8-27004 The selected file is not writable. Select another file.**

  The selected file is not writable. Select another file or check the permissions of the file.

- **8-32002 File already exists. OK to overwrite the file?**

  The file name you specified already exists. Click OK to overwrite the file. Click No to cancel.

- **8-40004 The port map table has been modified. Is it OK to discard the changes?**

  The port map table was modified. Click OK to close the dialog box without saving the changes. Click NO to cancel.

- **8-40005 File already exists. Do you want to overwrite it?**

  The file name you specified already exists. Click OK to overwrite the file Click No to cancel.

- **8-40006 Is it OK to overwrite the default port map file?**

  You are going to overwrite the default port map file. Click OK to overwrite the file. Click No to cancel.

- **8-40007 HF port map table has been modified. Is it OK to discard the changes?**

  The HF port map table was modified. Click OK to close the dialog box without saving the changes. Click No to cancel.

- **8-40008 Is it OK to overwrite the default HF port map file?**

  You are going to overwrite the default HF port map file. Click OK to overwrite the file. Click No to cancel.

- **8-40009 Pulse switch table has been modified. Is it OK to discard the changes?**

  The pulse switch table was modified. Click OK to close the dialog box without saving the changes. Click No to cancel.

- **8-40010 Is it OK to overwrite the default pulse switch file?**

  You are going to overwrite the default pulse switch file. Click OK to overwrite the file. Click No to cancel.

- **8-42016 Failed to measure calibration data of following PGU.**
  PG*nn*, PG*nn*, PG*nn*
  Please check using diag4070.
  PG level calibration data will not be updated.

  Abnormal data is meadured for the PG*nn*. The level calibration data for the PG*nn* is not updated. Confirm the PG functions by using the diag4070.

- **8-43001 PG connection table was modified. OK to exit the program without saving?**

  The PG connection table was modified. Click OK to exit the program without saving the changes. Click No to cancel.

- **8-43002 OK to overwrite the default PG connection file?**

  You are going to overwrite the default PG connection file. Click OK to overwrite the file. Click No to cancel.

- **8-43003 PG connection table was modified. OK to open new file without saving?**

  The PG connection table was modified. Click OK to open the new file without saving the changes. Click No to cancel.

- **8-43004 OK to exit the program?**

  You are going to exit the program. Click OK to exit the program. Click No to cancel.

- **8-43005 Searching PGU connections.**

  This message is displayed while searching the PG connection.

- **8-43051 RF connection table was modified. OK to exit the program without saving?**

  To exit the RF cable connection tool without saving, click the **OK** button. If you do not want to exit, click the **NO** button.

- **8-43052 OK to overwrite the default RF connection file?**

  To overwrite the default RF connection file, click the **OK** button. If you do not want to overwrite the file, click the **NO** button.

- **8-43053 RF connection table was modified. OK to open new file without saving?**

  To open the new file without saving the RF connection table, click the **OK** button. If you do not want to open the new file, click the **NO** button.

- **8-46001 Offline configuration was modified. Is it OK to exit the offline configuration tool without saving?**

  To exit the offline configuraiton tool without saving, click the **OK** button. If you do not want to exit, click the **Cancel** button.

- **8-46002 Is it OK to overwrite the default Offline configuration file?**

  To overwrite the default offline configuration file, click the **OK** button. If you do not want to overwrite the file, click the **Cancel** button.

- **8-46003 Offline configuration was modified. Is it OK to open a new file without saving?**

  To open a new file without saving the existing file, click the **OK** button. If you want to save the existing file, click the **Cancel** button.

- **8-46004 Is it OK to exit the offline configuration tool?**

  To exit the program, click the **OK** button. If you do not want to exit, click the **Cancel** button.

- **8-46005 Is it OK to select 12 pins (pins in multiples of 4)?**

  To select 12 pins (pins in multiples of 4), click the **OK** button. If you do not want 12 pins, click the **Cancel** button.

- **8-46006 Is it OK to select 24 pins (even number of pins)?**

  To select 24 pins (even number of pins), click the **OK** button. If you do not want 24, click the **Cancel** button.

- **8-46007 Is it OK to select 48 pins?**

  To select 48 pins, click the **OK** button. If you do not want 48, click the **Cancel** button.

- **8-46008 Is it OK to clear all pins?**

  To clear all pins, click the **OK** button. If you do not want to clear all pins, click the **Cancel** button.

8-47001 Is it OK to exit the System Management Panel?

To exit the System Management Panel, click the **OK** button. If you do not want to exit, click the **Cancel** button.

42-xxxxx

- **42-11020 Parameter name can consist of alphanumeric, space, underscore and period characters only.**
  42-11022 Parameter name must be 15 characters or less.

  Maximum number of characters for parameter name in the IDP setup window is 15.

- **42-11023 Failed to start idpcgen program (Reason).**

  Number of processes may be more than the limitation specified in kernel parameter, or available swap space or memory is not sufficient. The error message reported by the kernel is shown in Reason.

- **42-11024 A directory name must be specified.**

  A directory name must be specified in the Code Directory field of the Generate Code window.

- **42-11025 A file name must be specified.**

  A file name must be specified in the Source Code Name field of the Generate Code window.

- **42-11026 BASIC line number and its incremental number must be positive.**

  The line number and the incremental number in the BASIC Program Line Numbering field of the Generate Code window must be positive.

- **42-11028 Must specify Algorithm Function Name.**

  An algorithm function name must be specified in the Code Generation Setup window.

- **42-11029 Algorithm function name must begin with an alpha character.**

- **42-23000 Syntax error.**
  Usage: idpcgen {-b|-c} [-mhpfvSP] [-t directory] [-i file] [-o name] [-l 0(TIS)|1(algorithm)|2(program)] [-L [begin_num[,increment_num]]]

-b	Generates BASIC program
-c	Generates C program
-m	Generates Makefile for C language compile

-h	Generates C language header (.h file extension) file
-p	Generates program code for IDP running in the same session idpcgen executed
-f	Forces the specified output file to be overwritten
-v	Displays warning messages
-S	Generates or overwrites SPECS algorithm library spec file
-P	Generates program code using template file for puless
-t *directory*	Specifies that directory template file exists
-i *file*	Specifies IDP setup file (.dps file extension) from which program code is generated
-o *name*	Specifies file name (without file extension) program is saved to
-l 0|1|2	Specifies code generation level
-L *begin_num*, *increment_num*	Specifies BASIC line numbering

- **42-23001 Must specify one option: -b (BASIC) or -c (C)**

  Idpcgen command must be used with -b (BASIC) or -c (C) option to specify generated programming language.

- **42-23002 Unable to specify -b (BASIC) and -c (C) language options together.**

  Idpcgen command cannot specify both -b (BASIC) and -c (C) language options at the same time.

- **42-23003 Unable to specify -i and -p options together.**

  Idpcgen command cannot specify both -i (specifies .dps IDP setup file) and -p (specifies IDP setup window currently open) options at the same time.

- **42-23004 Failed to open input file File_name (Reason).**

  Idpcgen command failed to open specified File_name.dps IDP setup file. Cause of the error is shown in Reason.

- **42-23005 Must specify IDP setup file name with -i option.**

  Ipdcgen command must specify IDP setup file name from which the command generates the measurement program file.

- **42-23006 Failed to open output file File_name (Reason).**

  Idpcgen command failed to open output file specified as File_name. The cause of the error is shown in Reason.

- **42-23007 The specified file name already exists. Use -f option to overwrite it.**

  The output file name for measurement program generated by the idpcgen command already exists. To overwrite the file, specify the -f option with the idpcgen command, or if you do not want to overwrite the file, change the output filename.

- **42-23008 Must specify file name with -o option.**

  When -o option is specified, the idpcgen command must specify a file name without a file extension.

- **42-23009 Failed to open template directory Directory_name (Reason).**

  The idpcgen command, used with the -t option, failed to open the specified directory Directory_name. The cause of the error is shown in Reason.

- **42-23010 Must specify directory name with -t option.**

  When -t option is specified, the idpcgen command must specify a directory name.

- **42-23011 Improper generation level Level_number. Specify 0, 1, or 2 with -l option.**

  Idpcgen command has the wrong Level_number setup for -l option (code generation level option). With -l option, specify 0, 1, or 2.

- **42-23012 Must specify 0, 1 or 2 with -l option.**

  When -l option is specified, the idpcgen command must specify a code generation level number.

- **42-23013 Failed to open IPC connection to IDP (Reason).**

  Idpcgen command with -p option failed to open IPC connection to IDP. The cause of the error is shown in Reason.

- **42-23014 Failed to connect IPC to IDP (Reason).**

  Idpcgen command with -p option failed to connect IPC to IDP. The cause of the error is shown in Reason.

- **42-23015 idpcgen failed to communicate with IDP process (Reason).**

  Idpcgen command with -p option failed to communicate with the process IDP generated because of the error shown in Reason.

- **42-23016 Possible incorrect IDP setup file. Record format mismatched.**

  The file specified by the idpcgen command includes an unrecognized record. May be using an incorrect or incorrectly modified IDP setup file.

- **42-23017 Unable to generate program coding for IDP file created by Rev. B.01.20 or earlier.**

  Idpcgen cannot generate program code for IDP setup file created by 4080 system rev. B.01.20 or earlier. Load the IDP setup file in the 4080 rev. B.01.20 IDP setup window, then save the setup and execute the idpcgen command again. Or, after loading the IDP setup file in the rev. B.01.20 environment, generate program code in the IDP environment.

- **42-23018 Possible incorrect IDP setup file. Record format mismatched.**

  The file specified by the idpcgen command includes an unrecognized record. May be using an incorrect or incorrectly modified IDP setup file.

- **42-23100 Identical name (Parameter_name) cannot be used for different data types.**

  The IDP setup file, specified by the idpcgen command, has the identical parameter name setup shown in Parameter_name for a different data type. Change the parameter name in the IDP setup window.

- **42-23101 Failed to open template file File_name (Reason).**

  Idpcgen command failed to open the template file shown as File_name. The cause of the error is shown in Reason.

- **42-23102 Failed to read template file (Reason).**

  Idpcgen command failed to read the template file. The cause of the error is shown in Reason.

- **42-23200 Unknown Token. Replace `@' with `@@'.**

  There is an unknown token starting with `@' in the template file specified by the idpcgen command. Replace `@' with `@@'.

- **42-23301 Unknown variable type found: Variable_type (Variable_name); assumed REAL.**

  In the Algorithm Spec Editor, the setup for Variable_type shown in Variable_name is incorrect. The setup variable type is ignored and the code is generated as type real.

- **42-33000 Unrecognizable input record: Record**

  The IDP setup file specified by the idpcgen command, has an incorrect record that is not recognized by the code generation function. The IDP setup file may be modified incorrectly. The incorrect record is ignored.

- **42-33001 Duplicate Parameter_name exists. Change the Parameter_name .**

  The IDP setup file, specified by the idpcgen command, has a duplicate Parameter_name. Change the Parameter_name in the IDP setup window.

- **42-33002 Template file File_name is not in directory shown in Directory_name.**

  Template file File_name does not exist in the directory specified by Directory_name, when using idpcgen with the -t option.

- **42-33003 Unable to access the specified template directory. Default template File_name is loaded.**

  Unable to find the template file. The idpcgen command cannot access the specified template directory.

4 Unsupported 4062/4070 TIS Functions

This chapter lists Keysight 4062/4070 TIS functions that are not supported by Keysight 4080.

The following 4062 TIS functions are unsupported by 4070/4080 and cause an error during program execution. You must modify your program if you use these functions in your program.

- set_vm
- set_asearch
- set_cmu
- measure_cmu
- set_cv
- sweep_cv
- set_ct
- sweep_ct
- fncmu
- run_dcs_program
- readout_dcs
- readsweep_dcs
- ch_sw_off
- pgm_ch_sw_off
- pgm_ch_sw_on
- pgm_disable_dcs
- pgm_end
- pgm_force_i
- pgm_force_v
- pgm_measure_i
- pgm_measure_p
- pgm_measure_v
- pgm_search
- pfm_set_asearch
- pgm_set_pbias

- pgm_set_smu
- pgm_set_vm
- pgm_start
- pgm_wait
- pcs_hpib_open
- pcs_hpib_output
- pcs_hpib_enter
- pcs_hpib_clear
- pcs_hpib_spoll
- fnaddr4142
- fnunit4142
- fnchannel4142
- fnrange4142
- readerror4142

The following 4062 TIS functions are unsupported by the 4070/4080, and the 4070/4080 just ignores them during program execution.

- init_io
- lock_system
- unlock_system
- prepare
- ch_sw_on
- off_line_data
- data_creation
- open_tis_hpib_fd
- set_tis_hpib_fd

The following 4062/4070 TIS function is unsupported by the 4080, and the 4080 just ignores it during program execution.

- set_mode_pg

The following 4070 TIS function is unsupported by the 4080, and the 4080 just ignores it during program execution.

- timing_cal_pg

Edition 1, March 2020

N9100-90170
www.keysight.com

KEYSIGHT
TECHNOLOGIES