E5071C
RF

E5071C Agilent Technologies E5071C

Windows XP

E5091A

E5071C

E5071C	A.07.0x
	1.11
	June 17, 2007

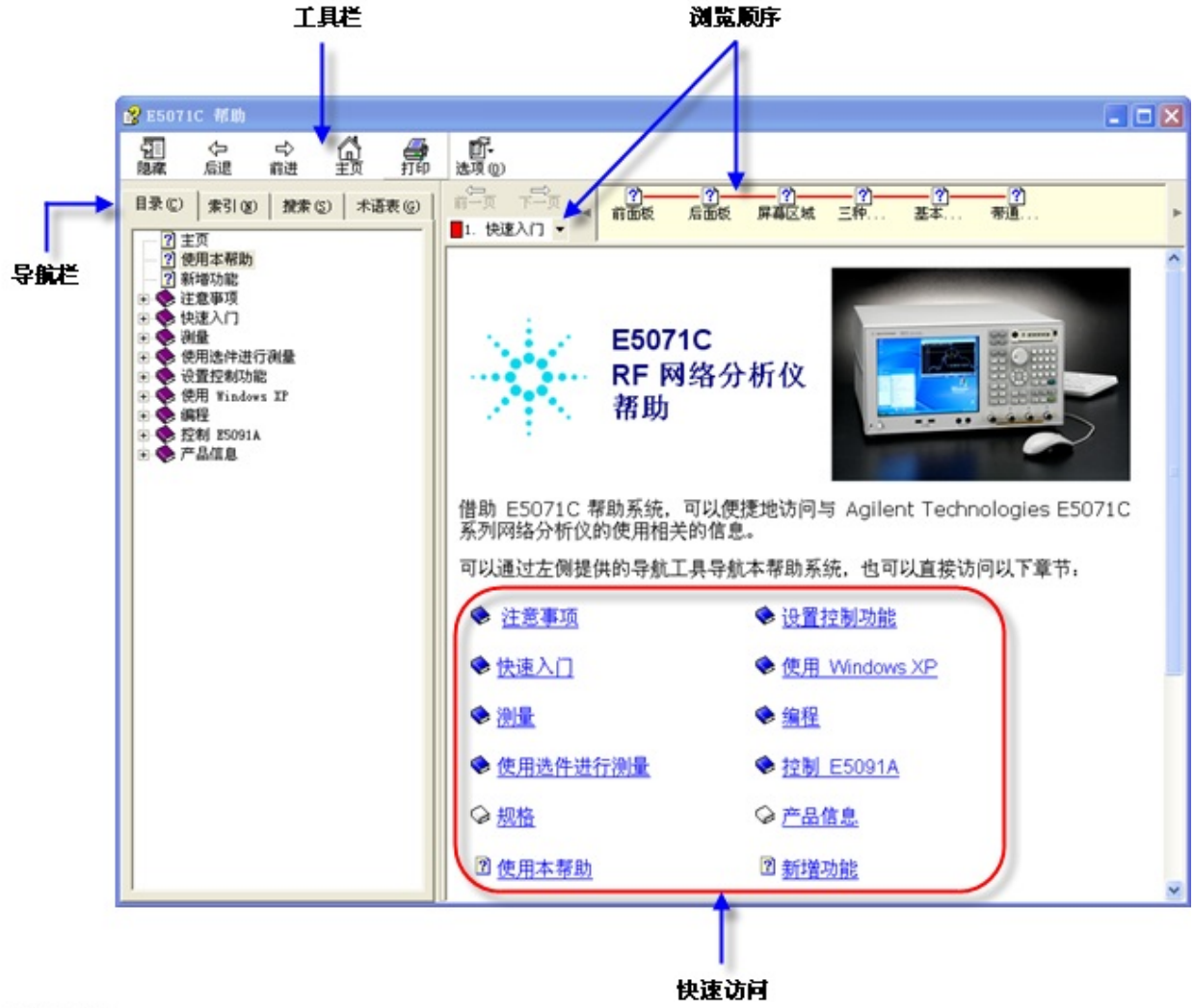
Agilent Technologies

© Agilent Technologies, Inc. 2006

E5071C

- E5071C
- E5071C
- PC E5071C
- E5071C
- E5071C
- E5071C

E5071C


工具栏
浏览顺序
导航栏
快速访问
E5071C 帮助
隐藏
后退
前进
主页
打印
选项(O)
目录(C)
索引(N)
搜索(S)
术语表(G)
主页
使用本帮助
新增功能
注意事项
快速入门
测量
使用选件进行测量
设置控制功能
使用 Windows XP
编程
控制 E5091A
产品信息
前一页
下一页
1. 快速入门
前面板
后面板
屏幕区域
基本...
帮助...
E5071C
RF 网络分析仪
帮助
借助 E5071C 帮助系统，可以便捷地访问与 Agilent Technologies E5071C 系列网络分析仪的使用相关的信息。
可以通过左侧提供的导航工具导航本帮助系统，也可以直接访问以下章节：
注意事项
设置控制功能
快速入门
使用 Windows XP
测量
编程
使用选件进行测量
控制 E5091A
规格
产品信息
使用本帮助
新增功能

e5071c199

E5071C

Microsoft HTML (ENA_C_Help.chm) PC

E5071C

- “ **Help**”
- E5071C “ **F1**”
- “ **ENA_C_Help.chm**” **D:\Agilent\Help**

E5071C

E5071C E5071C

PC E5071C

PC E5071C (**ENA_C_Help.chm**) E5071C
D:\Agilent\Help

http://www.agilent.com/find/ena

E5071C

“ **Focus**” E5071C

E5071C

E5071C E5071C E5071C

-
-
-
-

“ **Contents**”“ **Index**”“ **Search**”“ **Glossary**”
“ **Contents**” E5071C “ **Glossary**” E5071C
“ **Index**”“ **Search**” E5071C /

“ **Home**” E5071C “
“ **Print**” E5071C

E5071C

E5071C E5071C

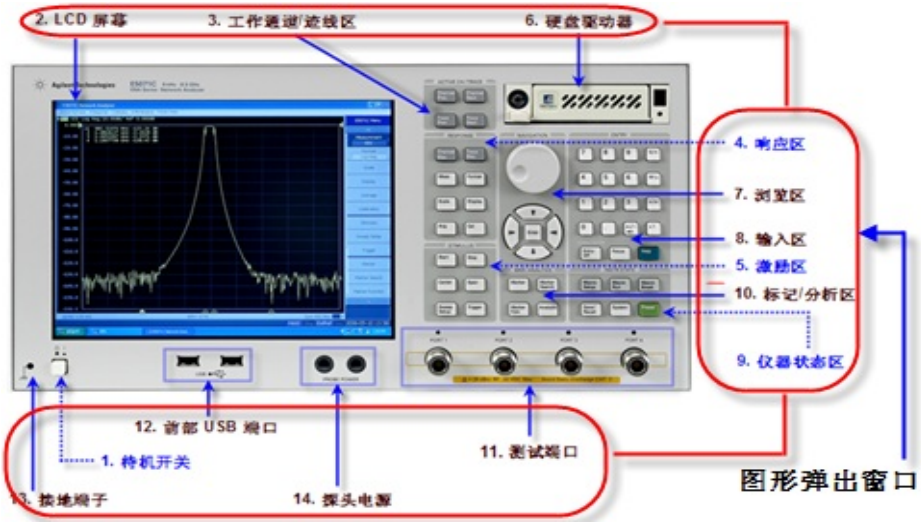
E5071C

	12 Verdana	
I	12 Arial	**C:\Doc and Settin** **examp**
	E5071C 12 Verdana	**Trace** **Focus** **Marke**
	E5071C 12 Arial	**Auto S All** **S22** **Target Transi**
	1.	

2.

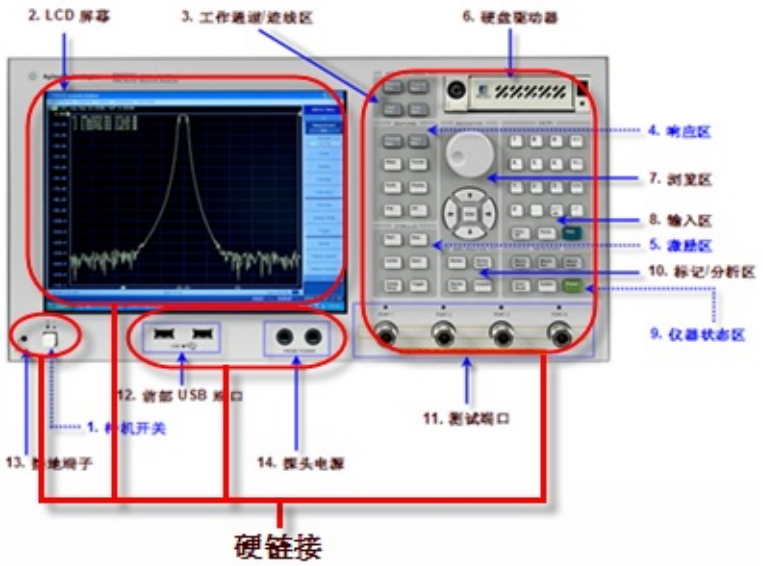
注意

E5071C



注意

E5071C



E5071C 7 12 Arial “”

注意

注意

警告

E5071C E5070B/71B

E5071C A.07.0x E5070B/71B A.06.50

-
- AUX
-
-
- 65dB 9 kHz 5 GHz
- IF 500 kHz
- 36
- ECal
-
- 008
-
- Windows XP
- XGA 10.4 LCD
- USB 2 4
-
- 235435285 485
- 9 KHz 230430280 480/100 kHz 235435285 485
- LCD
-

- /
-
-
-
- VBA

-
-
-
- ENA
-

Agilent Technologies

E5071C IEC61010-1 II 2E5071C

IEC60825-1 E5071C LED 1 1 LED

-
-
-
-
-

Agilent Technologies

-

⚠

~

⎓

I

O

⏚

⏚

⏚

⏻

Agilent Technologies

Microsoft®MS-DOS®Windows®Visual C++®Visual Basic®
VBA® Excel® Microsoft Corporation

Java® Sum Microsystems Corporation

© Copyright 2002-2006 Agilent Technologies

Agilent Technologies Agilent Technologies

“”Agilent Agilent
Agilent

Agilent Technologies

Agilent Technologies

Agilent Technologies

“”“”“”

“”Agilent Technologies “”Agilent Technologies “”

AGILENT TECHNOLOGIES

“”“”Agilent Technologies “”Agilent Technologies Agilent Technologies

VBA

E5071C Agilent (D:\Agilent) VBA VBA

VBA

VBA VBA

VBA VBA Agilent Technologies VBA
Agilent Technologies Agilent Technologies

Agilent http://www.agilent.com/find/ena_support/

E5071C Windows XP

ENA

ENA

1.

_____ E5071C HDD

2.

ENA Agilent Microsoft Windows XP

- ENA Windows XP
-
- ENA
- Agilent
- Agilent

3.

ENA ENA LAN (Internet)

ENA ENA DCOM

4. **Windows**

ENA ENA service pack Microsoft Windows
ENA http://windowsupdate.microsoft.com

5.

ENA ENA

HDD Windows Windows

Windows XP E5071C

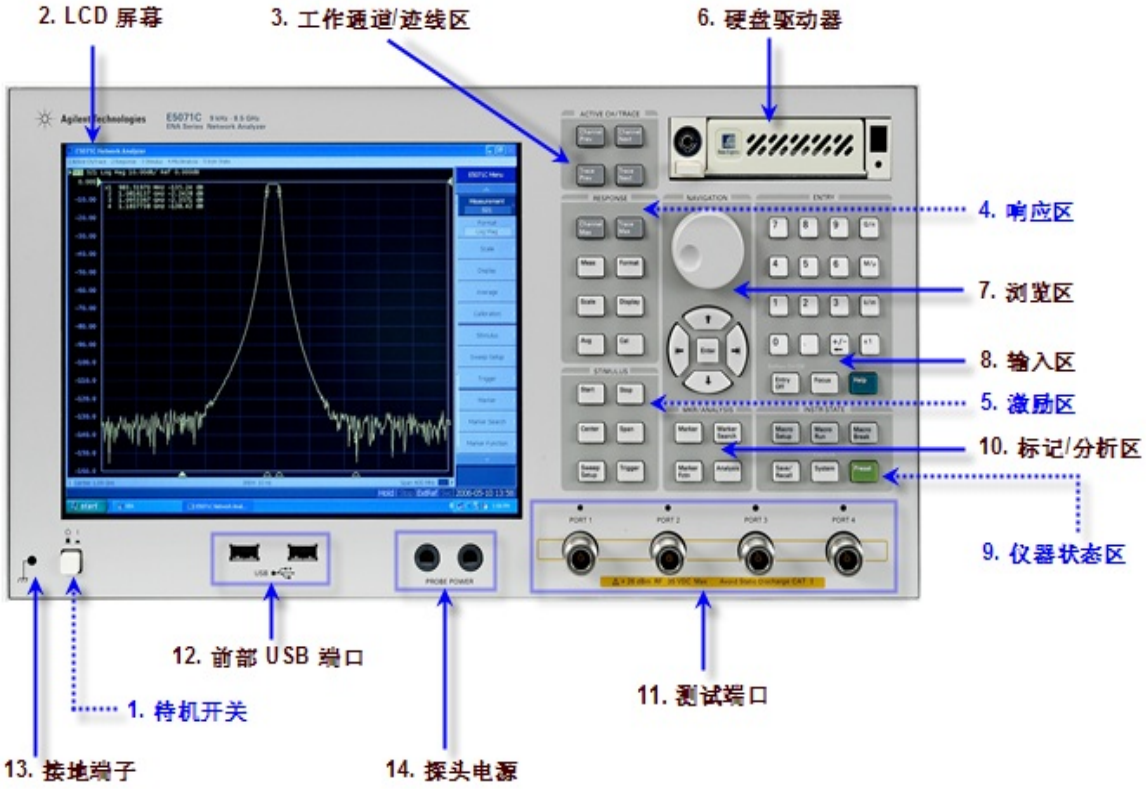
-
- /“Power on test fail”“Calibration data lost”
  SVC

- VBA PC
-
-

Agilent

E5071C

-
  -
  -
  -
-
-
-


2. LCD 屏幕
3. 工作通道/迹线区
6. 硬盘驱动器
4. 响应区
7. 浏览区
8. 输入区
5. 激励区
10. 标记/分析区
9. 仪器状态区
12. 前部 USB 端口
11. 测试端口
1. 待机开关
13. 接地端子
14. 探头电源

1.

E5071C (|) (O)

E5071C

1. E5071C
2. _____

_____ (|) (O)

_____ E5071C

“”

2. LCD

E5071C 10.4 TFT LCD LCD LCD
LCD

- LCD LCD

-

3. /

“Channel Next”	
“Channel Prev”	
“Trace Next”	
“Trace Prev”	

4.

E5071C

“**Channel Max**”	
“**Meas**”	“Measurement”“Measurement” S
“**Format**”	“Format”“Format”
“**Scale**”	“Scale”“Scale”
“**Display**”	“Display”“Display”
“**Avg**”	“Average”“Average” IF
“**Cal**”	“Calibration”“Calibration”/

5.

“**Start**”	“Stimulus”
“**Stop**”	“ **Start**”“Stimulus”
“**Center**”	“ **Start**”“Stimulus”
“**Span**”	“ **Start**”“Stimulus”
“**Sweep Setup**”	“Sweep Setup”“Sweep Setup”
“**Trigger**”	“Trigger”“Trigger”

6.

E5071C /

7.

“ **Focus**”

Enter	

	(|)

Enter	

8.

E5071C

01 23 ..... 9.	
+/- & ⬅	**+/-** + - ⬅ 注意 **+/-** **+/-** (⬅)
G/nM/uk/m **x1**	**+/-** **x1**
“**Softkey On/Off , Entry Off**”/	/
“**Help**”	E5071C
“**Focus**”	

9.

/ E5071C

“**Macro Setup**”	“Macro Setup”“Macro Setup” VBA VBA
“**Macro Run**”	“main” VBA Module1 VBA
“**Macro Break**”	VBA
“**Save/Recall**”/	“Save/Recall”/“Save/Recall” /
“**Capture/System**” /	LCD “System” “System”“ **Image**”“System” “ **Print**”
“**Preset**”	“Preset”“Preset”“ **OK**”

10. /

/ “”

“**Marker**”	“Marker”“Marker”/ 10
“**Marker Search**”	“Marker Search”“Marker Search”
“**Marker Fctn**”	“Marker Function”“Marker Function”
“**Analysis**”	“Analysis”“Analysis”

⚠ 11.

DUT LED

50 ΩN

(DUT)

IEC 61010-1 I

12. USB

USB ECalUSB ECal ECal
E5071C ECal

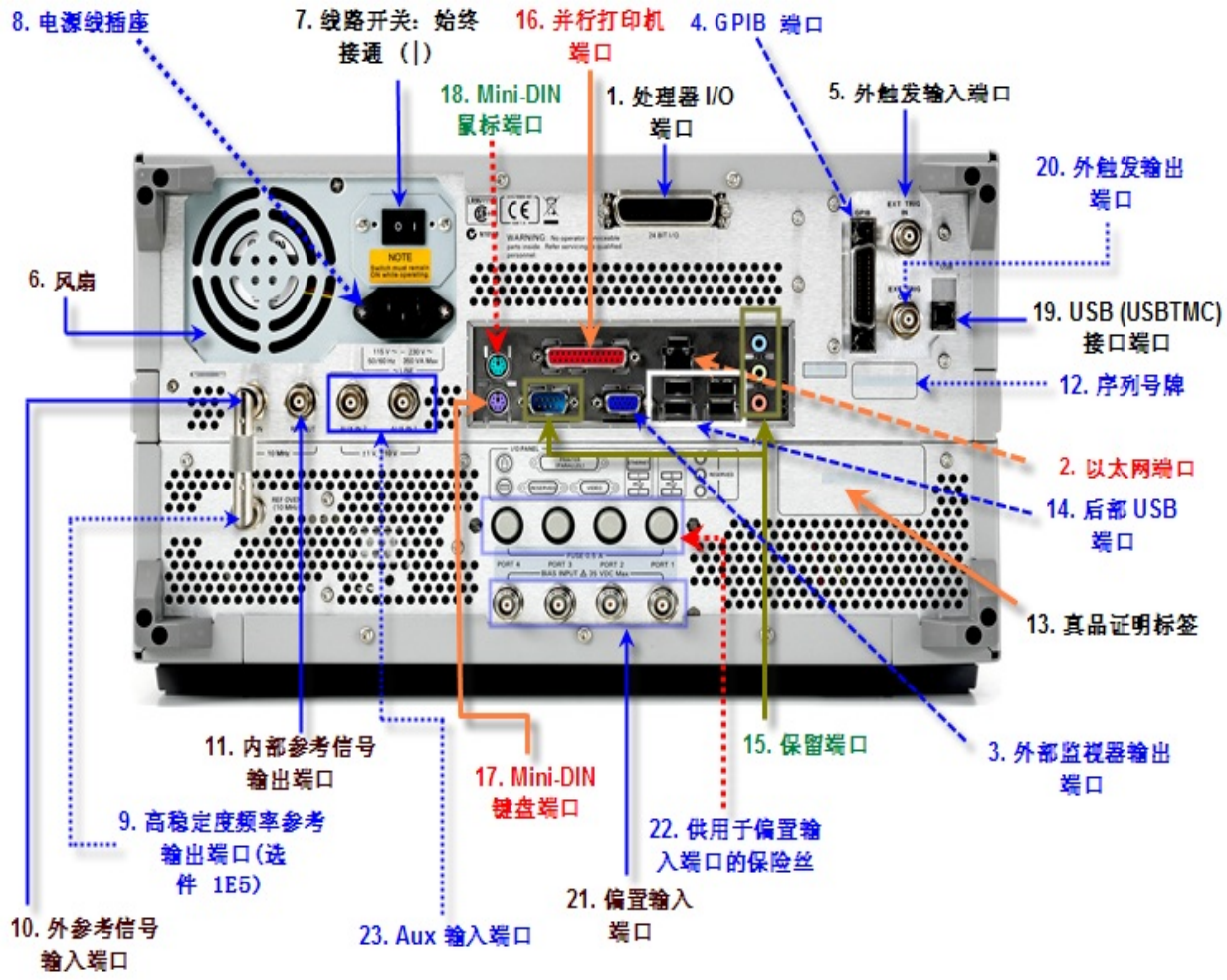
USB

13.

E5071C E5071C

14.

E5071C



1. I/O

I/O

36 Ribbon (Centronics)

2.

E5071C LAN LAN PC SICL-LAN Telnet

	8 RJ-45
Base	10Base-T/100Base-TX

3.

LCD

15 VGA

4. GPIB

(GPIB)

GPIB E5071C USB/GPIB E5071C GPIB
E5071C

5.

TTL “”“ **Trigger**”>“ **Trigger Source**”> “**External**”

BNC

6.

E5071C

7.

(|)

(O)

8.

E5071C

9. Ref Oven 1E5

1E5

	BNC
	10 MHz ± 1 ppm 0 dBm

1E5 BNC-BNC (Ref In)

10. (Ref In)

E5071C E5071C

	BNC
	10 MHz ± 10 ppm-3 dBm +10 dBm

E5071C E5071C ExtRef

1E5 BNC-BNCRef Oven
1E5

11. (Ref Out)

E5071C E5071C

	BNC
	10 MHz ± 5 ppm (0 dBm ± 3 dBm)
	50 Ω

12.

13.

“”

14. USB

USB ECalUSB ECal ECal E5071C

USB

15.

16.

25 E5071C

17. Mini-DIN

mini-DIN E5071C VBA E5071C

E5071C Mini-DIN USB

18. Mini-DIN

mini-DIN

E5071C Mini-DIN USB

19. USB (USBTMC)

E5071C USB USB

	(USB) B 4
	USBTMC-USB488 USB2.0

20.

50mA

		1	µsec	
VOH		5	V	I_{out}=-50µA
VOL		0	V	I_{out}=50µA

⚠ 21.

BNC

35 V

E5071C 235435285 485

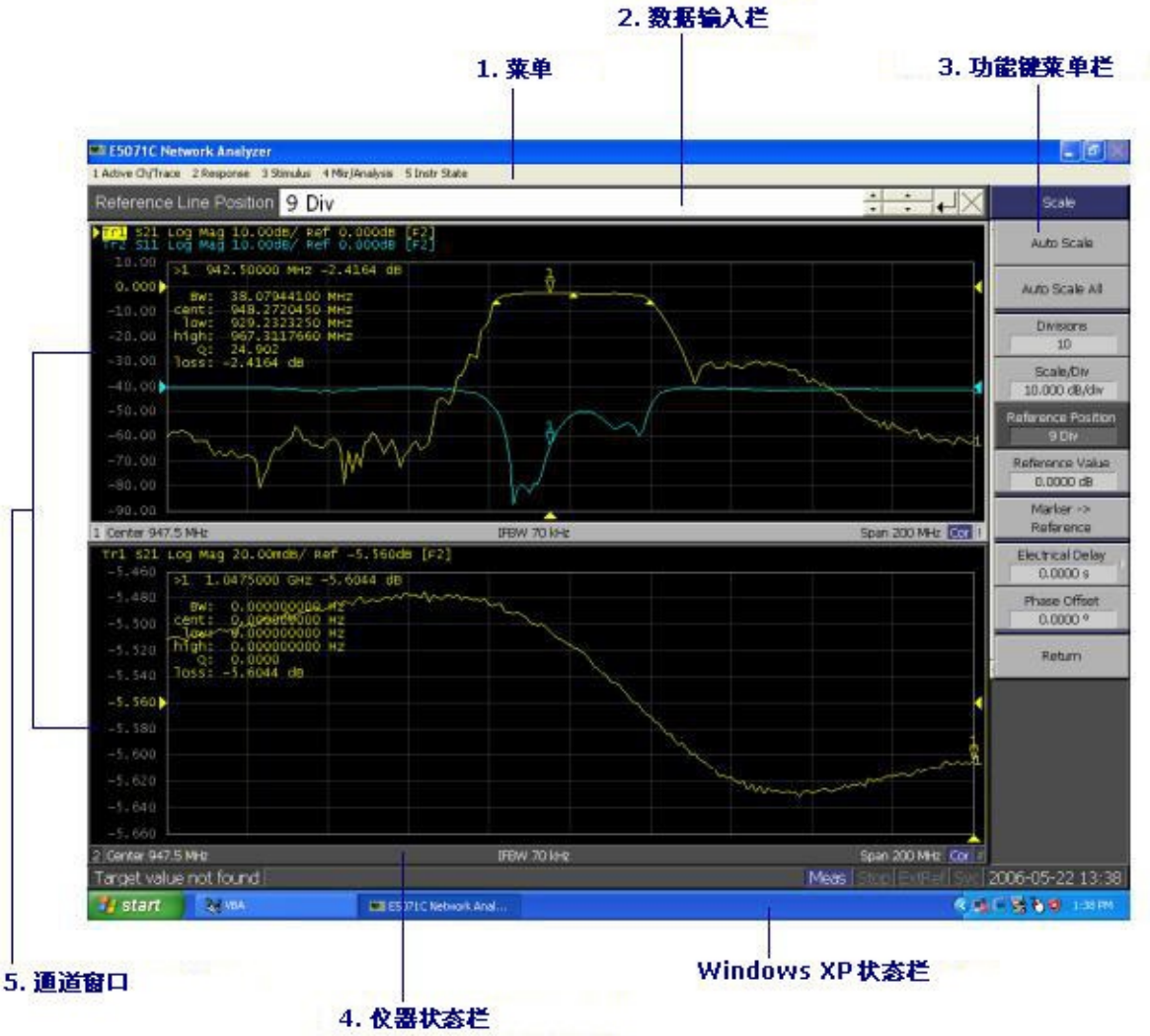
22.

	0.5A/125V
	2110-0046

E5071C 235435285 485

23. Aux

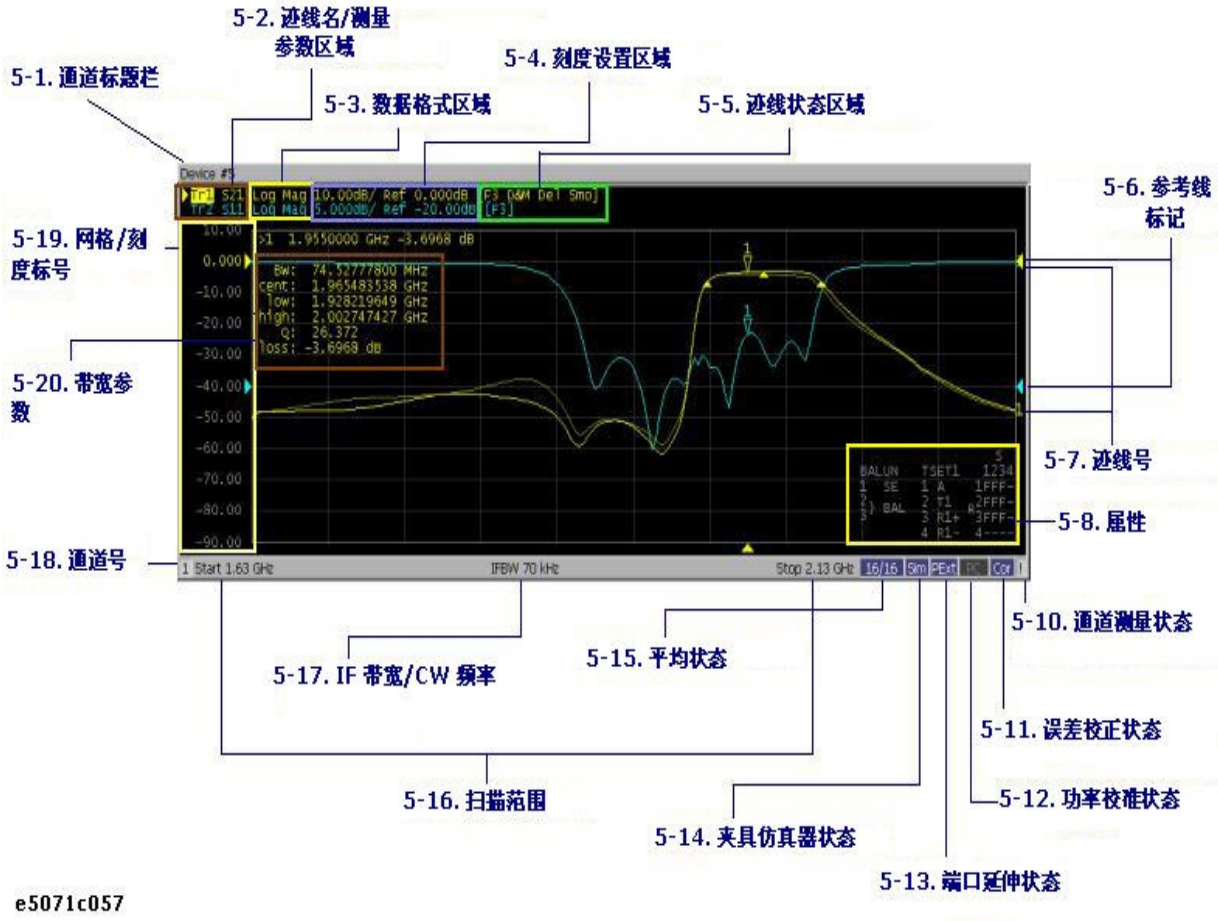
AUX



e5070bue053

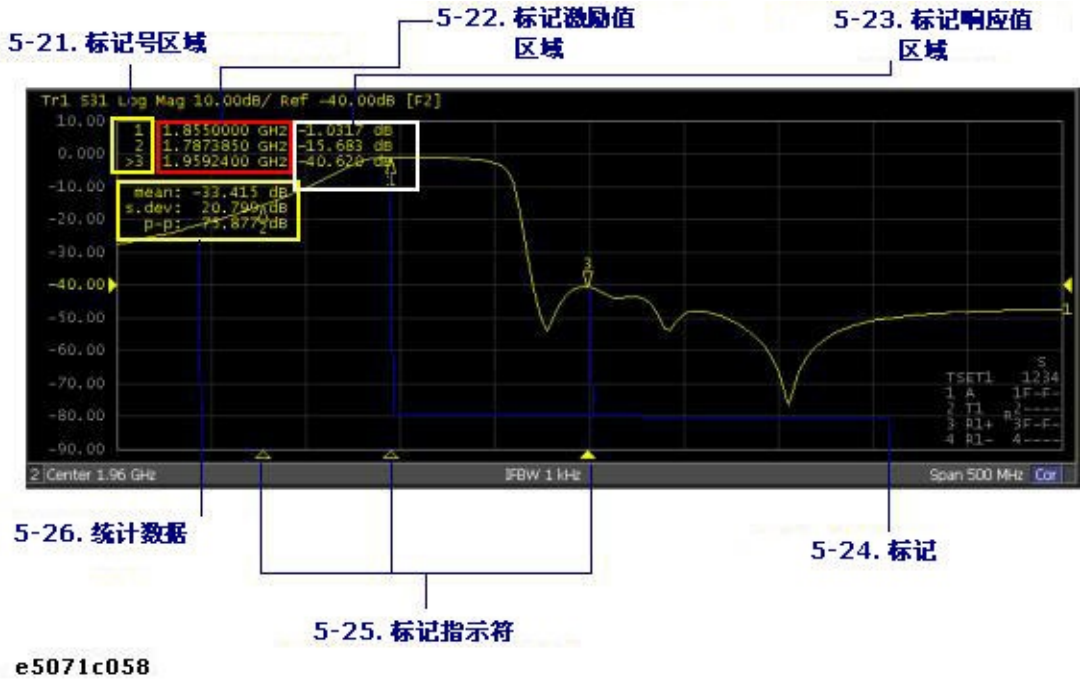
1“ **Channel Next**”
“ **Channel Previous**”

1



e5071c057

2



5-1.

5-2. /

Tr1 Tr9 ▶ “ **Trace Next**”“ **Trace Prev**”

5-3.

5-4.

“10.00dB/” 10 dB“Ref 0.000dB” 0 dB

5-5.

	[ ]	
	RO **RS** **RT** **ER** **F1** **F2** **F3** **F4**	(n) (n) (n) 1 2 /2 TRL 3 /3 TRL 4 /4 TRL
/	**Nothing** **M** **D&M**	
"ON" ()	**D+M (D+M&M)** **D- M (D- M&M)** **D*M (D*M&M)**	Data+Mem Data - Mem Data*Mem

	D/M (D/M&M)	Data/Mem
	Del	0
	Smo	
	Gat	
	Zr **Zt** **Ztsh** **Yr** **Yt** **Ytsh** **1/S** **Conj**	 S
	PC	
	RC	
	Cor	

5-6.

Y ▶ ◀“ **Scale**”>“ **Reference Position**”

5-7.

5-8.

e5071c444

E5091A	E5091A

5-9.

5-10 5-16

5-10.

!	1.5 ↑

#	

5-11.

5-12.

5-13.

“ON”“OFF”

PExt	

5-14.

“ON”“OFF”

Sim	

5-15.

n/m	mn

5-16.

“start”/“stop”“center”/“span”

5-17. IF /CW

/ IF CW

5-18.

5-19.

Y Y ▶ ◀ “**Scale**” +“Reference Value”

5-20.

5-21.

5-215-22 5-23 5-21 > △

5-22.

/

5-23.

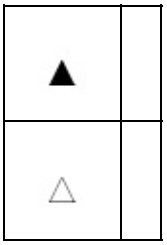
5-24.

10

n ▽	
△ n	

“n” n

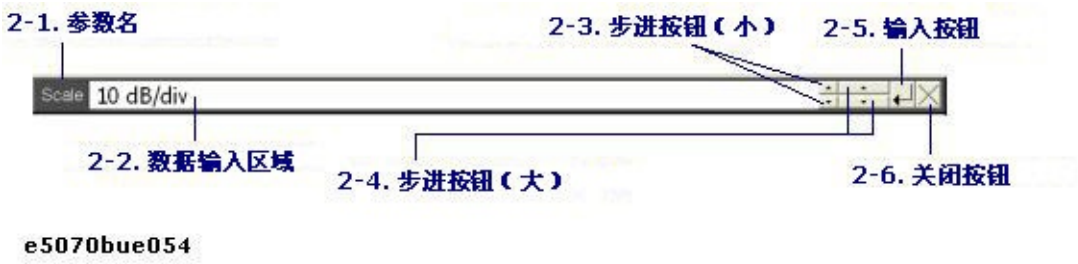
5-25.



5-26.

-

E5071C



“ **Focus**”

2-1.

2-2.

2-3.

2-4.

2-5.

2-6.

4-1. /

4-2.

LCD

4-3. RF

RF

4-4.

E5071C

Setup	
Hold	
Init	
Man	“Manual”

Ext	“External”
Bus	“Bus”
Meas	

4-5. VBA

E5071C VBA

Run	VBA
Stop	VBA

4-6.

(Ref In) E5071C “ ExtRef”

ExtRef	
ExtRef	

Ref Oven 1E5 (Ref In) “ExtRef”
“ExtRef”

4-7.

SVC	E5071C E5071C

SVC	E5071C
SVC	E5071C

4-8.

YYYY-MM-DD HH:MM

YYYY **MM**
DD
HH:MM0:00 23:59

/“ **System**”>“ **Misc Setup**” >“**Clock Setup**”
>“ **Show Clock**”

E5071C / /

“ **Focus**”

3-1.

3-2.

3-3.

3-4.

3-5.

3-6.

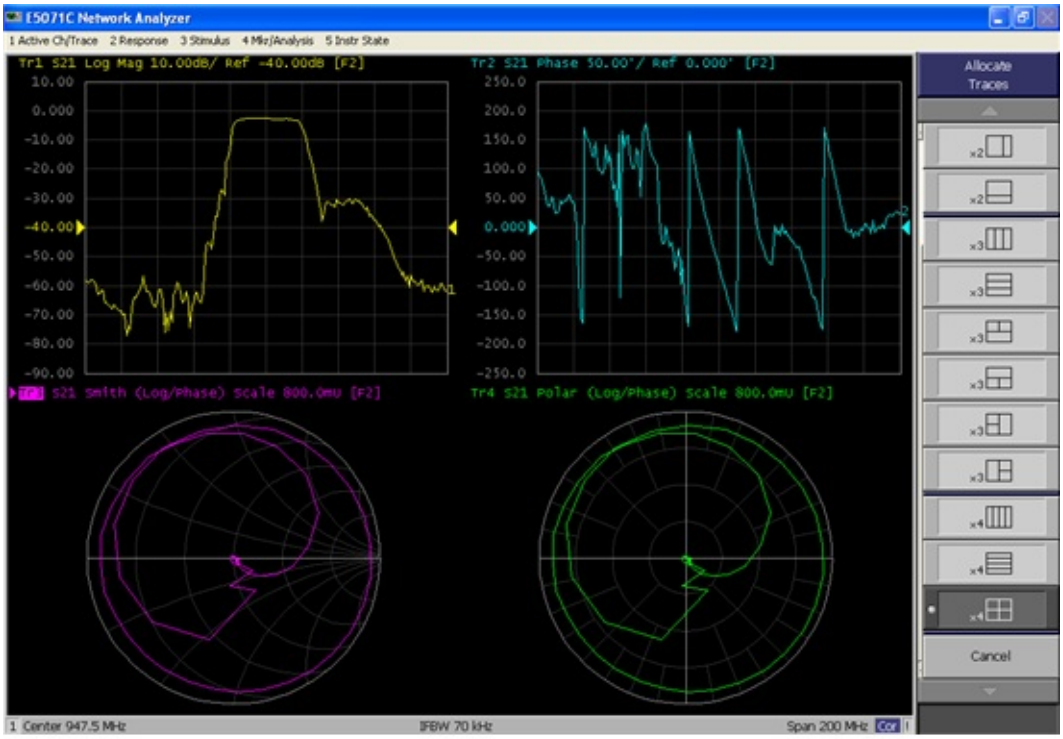
3-7.

3-8.

3-9.

E5071C

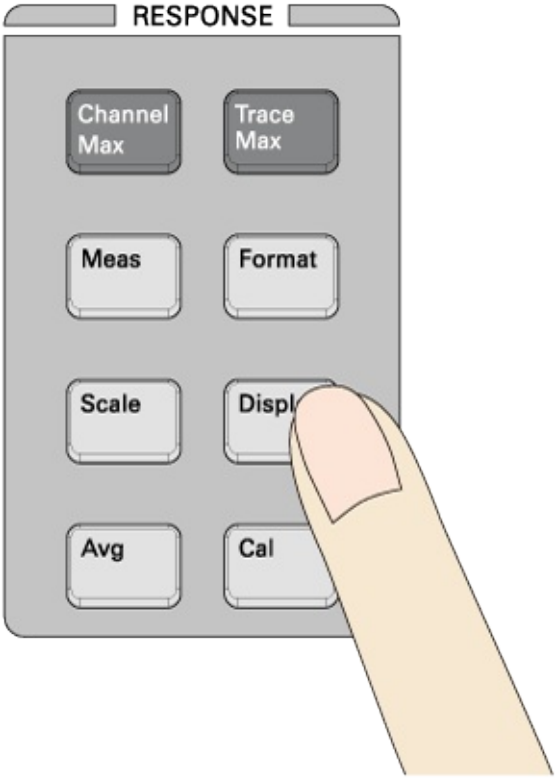
“**Display**”>“ **Allocate Traces**”>



e5071c193

-

1. “ **DISPLAY**”

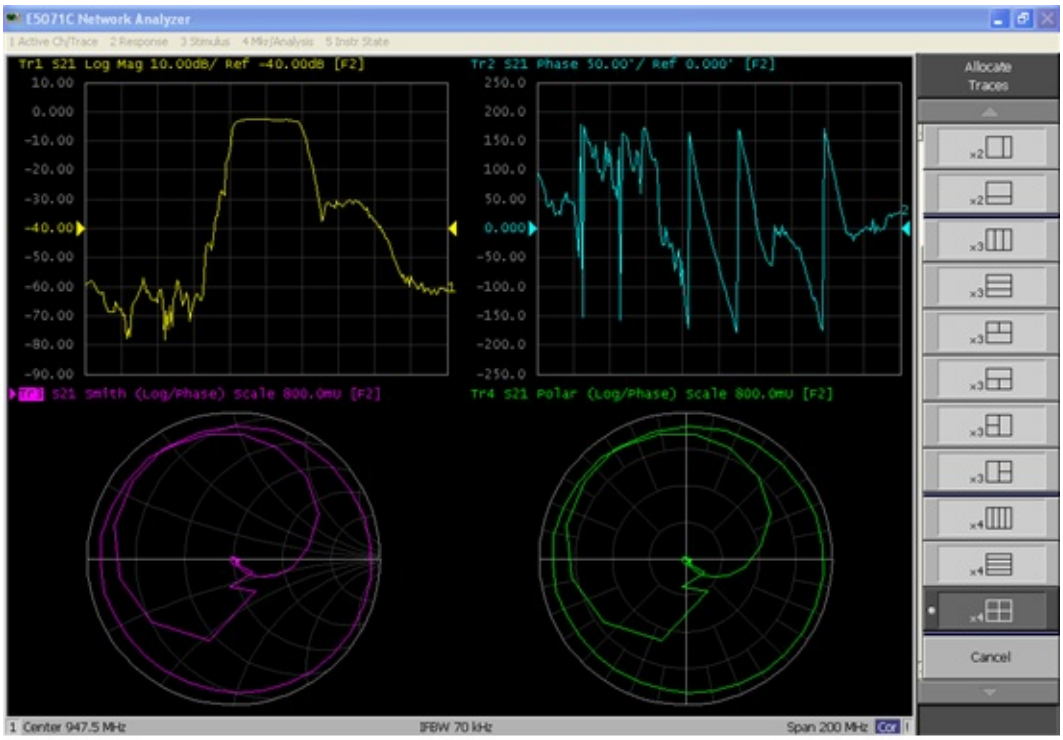


e5071c307

2. ↓ ↑ “ **Allocate Traces**” **Enter** ( )

e5071c002

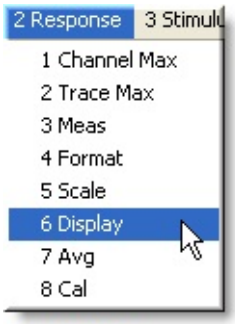
3. ↓ ↑ **Enter**



e5071c003

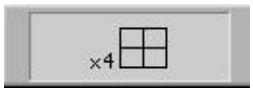
-

1. " **Response**"" **Display**"



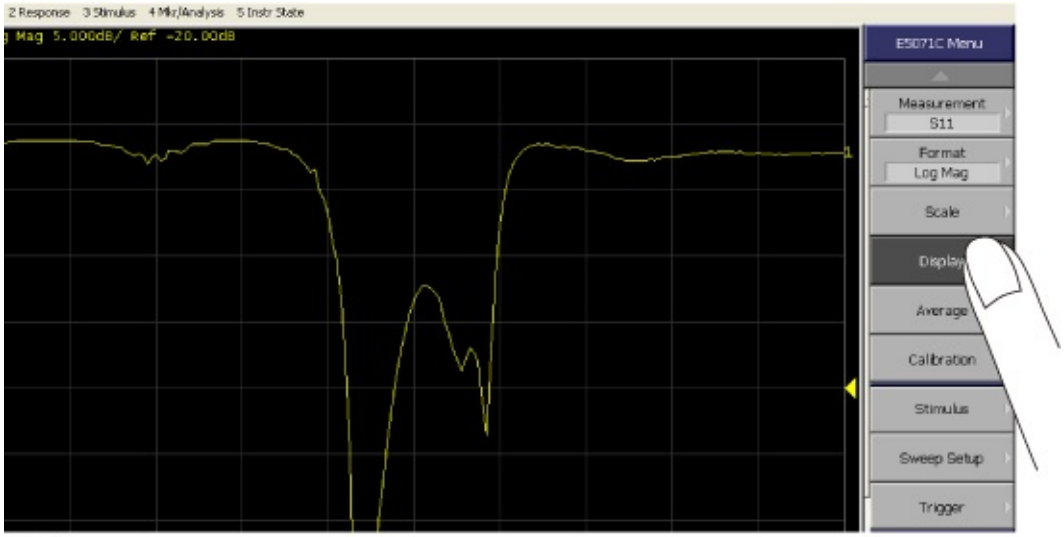
e5071c051

2. " **Allocate Traces**"

3. 

-

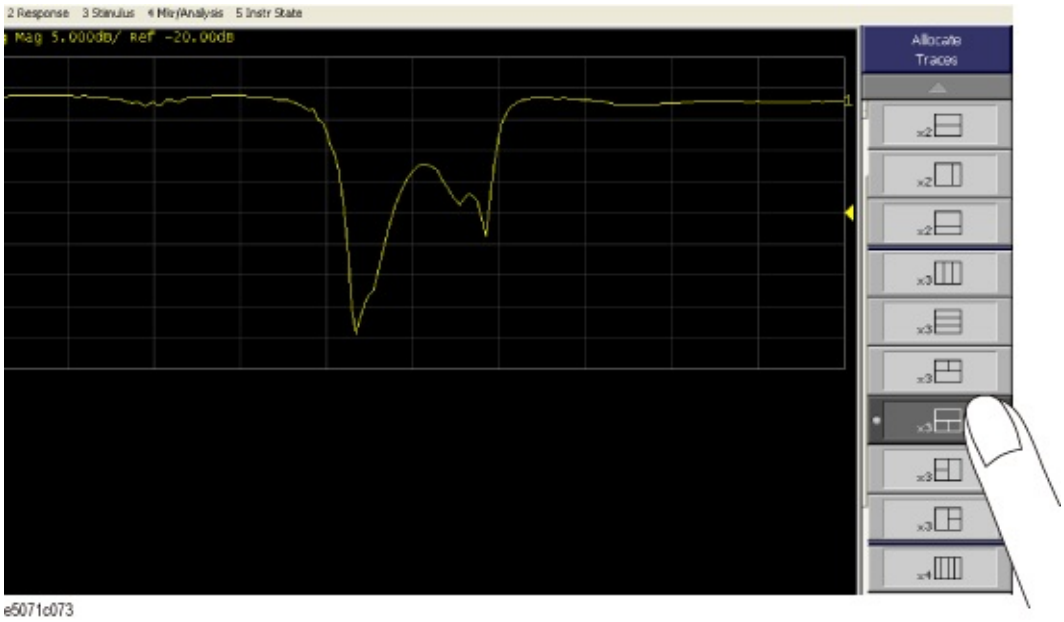
1. E5071C “ **Preset**”“ **Display**”



e5071c072

2. /“ **Allocate Traces**”

3. /



e5071c073

E5071C E5071C

1.

- E5071C
-
    - S
    -
    -
    -
    -
    - IF

2.

-
-
-
-
-

3. **(DUT)**

- DUT
-

4.

-

5.

-

947.5 MHz 947.5 MHz (DUT) DUT

1.

1. E5071C
   “Preset” >**“OK”**

2. S S21

   “Meas” > **S21**

   S S12

3.

   “Format” >**“Log Mag”**

4. 200 MHz

   “Center” > **9** > **4** > **7** > **.5** > **M/m**
   “Span” > **2** > **0** > **0** > **M/m**

   “G” GHz“M” MHz“k” kHz

5. 401

   “Sweep Setup”>“ **Points**”> **4** > **0** > **1** > **x1**

6. -10 dBm

   “Sweep Setup”>“ **Power**”> **+/-** > **1** > **0** > **x1**

7. IF IF 10 kHz

   “Avg”>“ **IF Bandwidth**”IF > **1** > **0** > **k/m**

2.

2

1. 85032F

   “**Cal**”>“ **Cal Kit**”>“ **85032F**”

2. 2 1 2

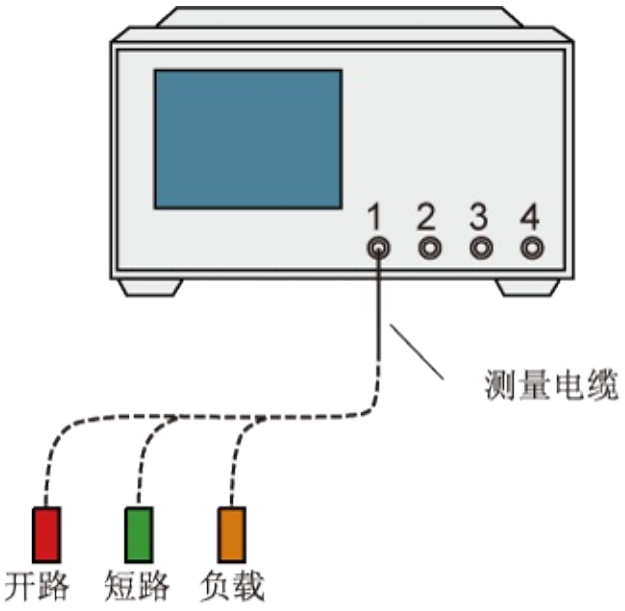
   “**Cal**”>“ **Calibrate**”>“ **2-Port Cal**”2 >“ **Select Ports - 1-2**” - 1-2

3. 1 1 “ **Port 1 Open**” 1

   “**Cal**”>“ **Calibrate**”>“ **2-Port Cal**”2 >“**Reflection**”>“ **Port1 Open**” 1

   1 /

   //

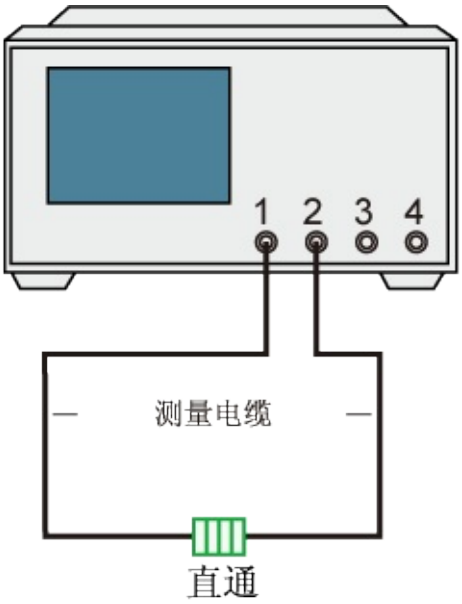


e5071c323

4. 2 //

5. “ **Port 1-2 Thru**” 1-2

“**Cal**”>“ **Calibrate**”>“ **2-Port Cal**”2 >“**Transmission**”>“ **Port 1-2 Thru**” 1-2



e5071c316

6. 2

“**Cal**”>“ **Calibrate**”>“ **2-Port Cal**”2 >“ **Done**”

7.

“**Save/Recall**”/>“ **Save Type**”>“ **State & Cal**”

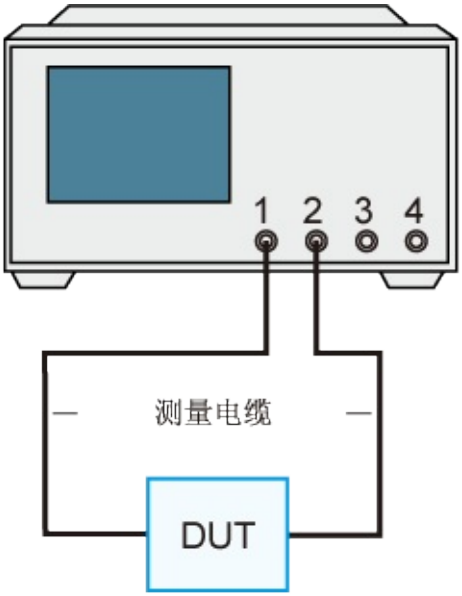
8. E5071C “X”

“**Save/Recall**”/>“ **Save Type**”>“ **State 0X**” 0X

3. (DUT)

1. DUT E5071C

DUT



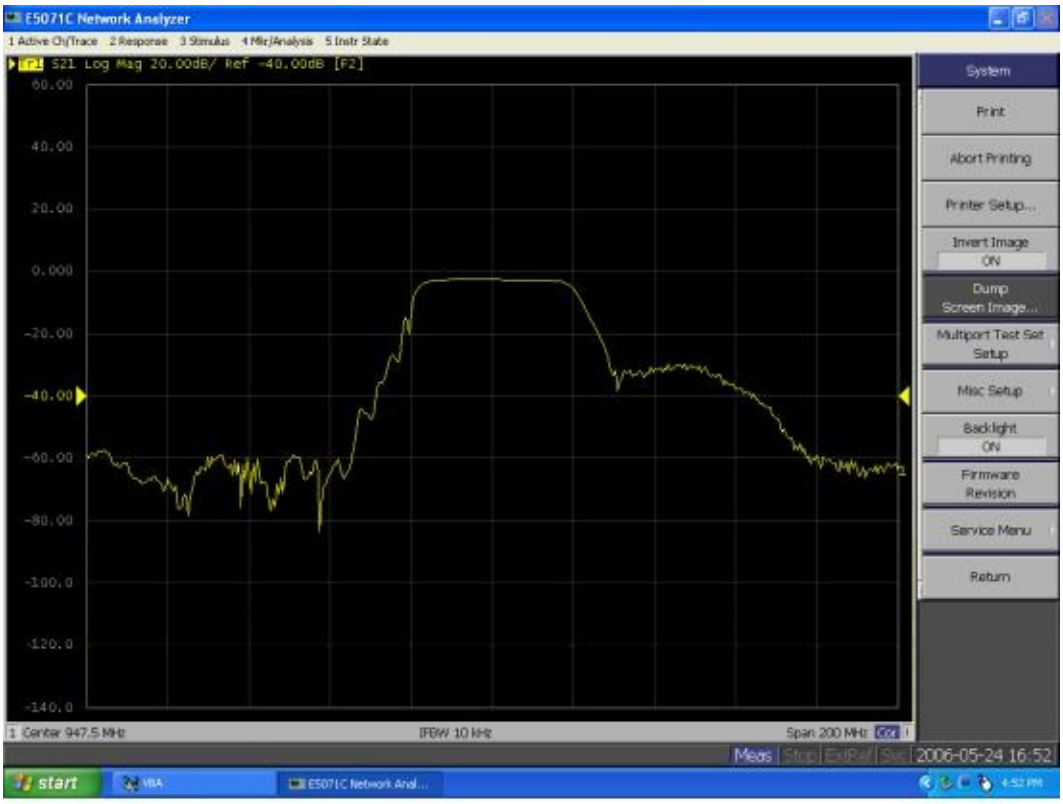
e5071c315

2.

“Scale”> **“Auto Scale”**

“ **Scale/Div**”/“ **Reference Position**”“ **Reference Value**”

S21

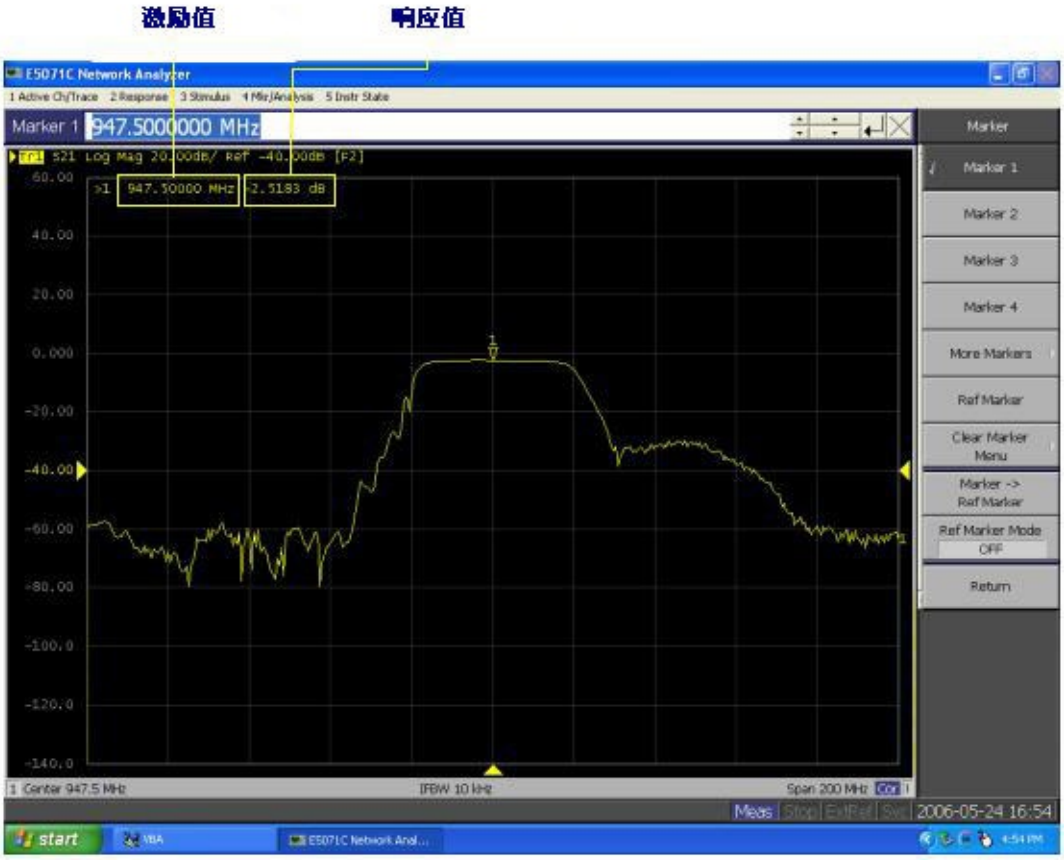

E5071C Network Analyzer
1 Active Ch/Trace 2 Response 3 Stimulus 4 Mkr/Analysis 5 Instr State
S21 Log Mag 20.00dB/ Ref -40.00dB [F2]
60.00
40.00
20.00
0.000
-20.00
-40.00
-60.00
-80.00
-100.0
-120.0
-140.0
1 Center 947.5 MHz
IFBW 10 kHz
Span 200 MHz
System
Print
Abort Printing
Printer Setup...
Invert Image
ON
Dump
Screen Image...
Multiport Test Set
Setup
Misc Setup
Backlight
ON
Firmware
Revision
Service Menu
Return
Meas
2006-05-24 16:52
start
VBA
E5071C Network Anal...
4:52 PM

e5071c053

4.

-3 dB

1. “**Marker**”>“ **Marker 1**” 1
2.
   - **9** > **4** > **7** > **.**> **5** > **M/m**
   - (947.5 MHz)
3.



e5071c054

-3 dB

Q

BW	low high
cent	low high
low	
high	
QQ	Q = cent/BW

loss	0 dBm

1.

“**Marker**”>“ **Marker 1**” 1

2.

- **9**> **4** > **7** > **.**> **5** > **M/m**
- (947.5 MHz)

3. -3 dB

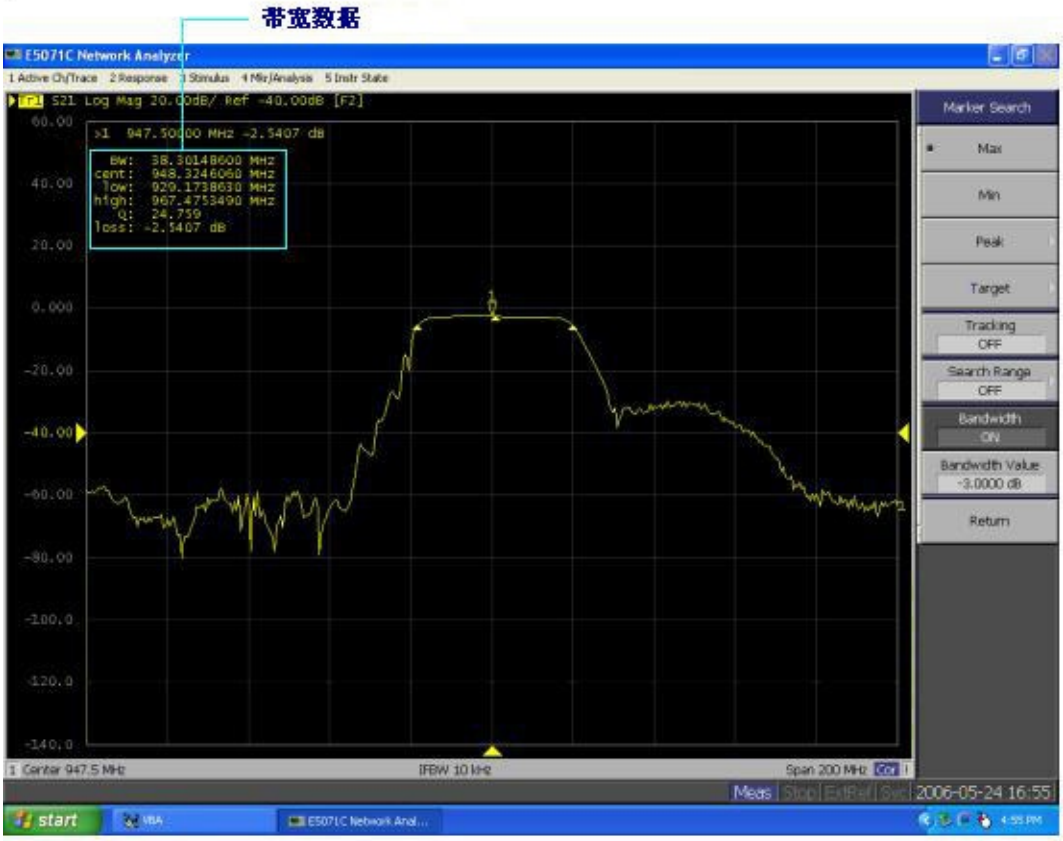
“**Marker Search**”>“ **Bandwidth Value**”> **+/-** > **3** > **x 1**

4. “ON”

“**Marker Search**”>“ **Bandwidth**”

5. BWcentlowhighQ loss

-3 dB



e5071c055

5.

CSV

CSV .csv E5071C CVS Microsoft Excel

"**Save/Recall**"/ >"**Save Trace Data**"

E5071C E5071C Windows .bmp.png

"**System**">" **Dump Screen Image**"

LCD “ **Capture/System**”/ LCD

E5071C

-
-
-
-
-
-
-
-

-
-
- Z0
-
-
-
-
-

E5071C

	“**Preset**”> “**OK**” SYST:PRES
*RST	*RST
	E5071C

- .
- /
-
- .
- .
-

E5071C 36 9 36

46912 16 E5071C

E5071C E5071C

/ ////

/

1	4	1601
2	4	1601
4	16	1601
9	9	1601
12	6	1601
16	4	1601
16	16	1601
24	12	1601
36	9	1601
1	4	20001

/“1-4-”“2-4-”

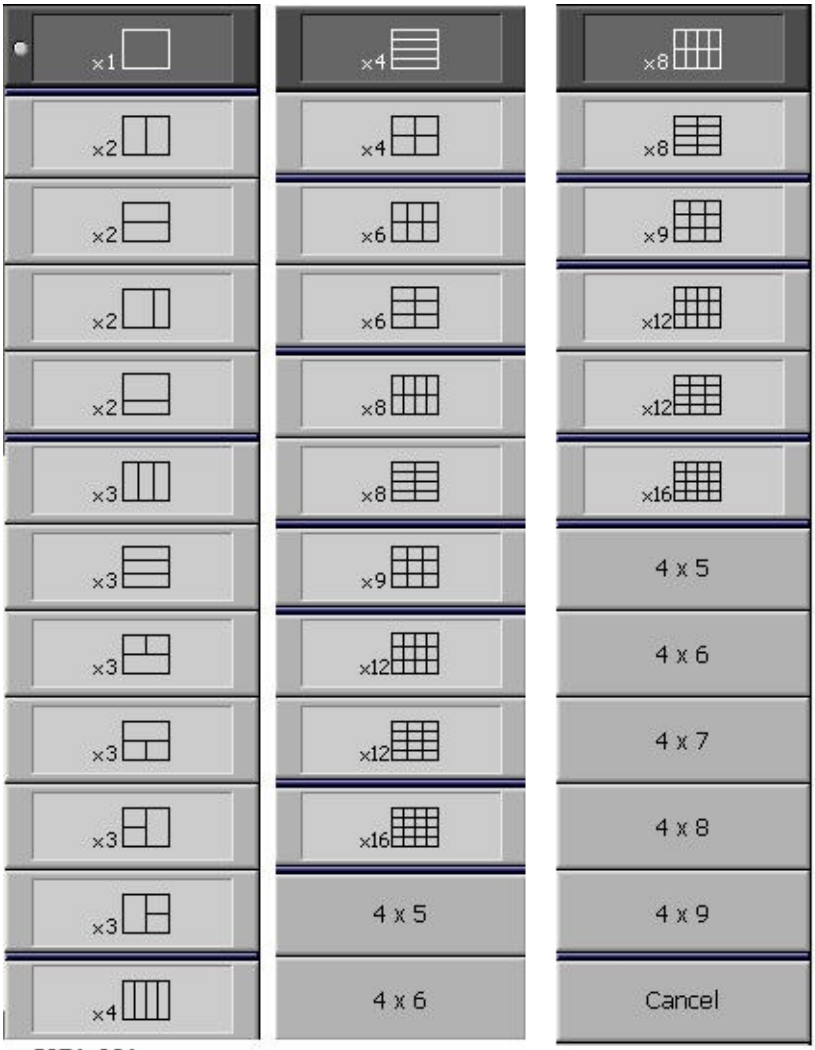
/

1. “ **System**”>“ **Misc Setup**”>“ **Channel/Trace Setup**” /
2.
3. “New Channel/Trace configuration will take effect after firmware restart.”/

4. “ **OK**”
5. “ **Preset**”
6. “Do you want to restart now?”?
7. “ **Yes**”

(≥2)

1. “ **Display**” >“ **Allocate Channels**”
2.



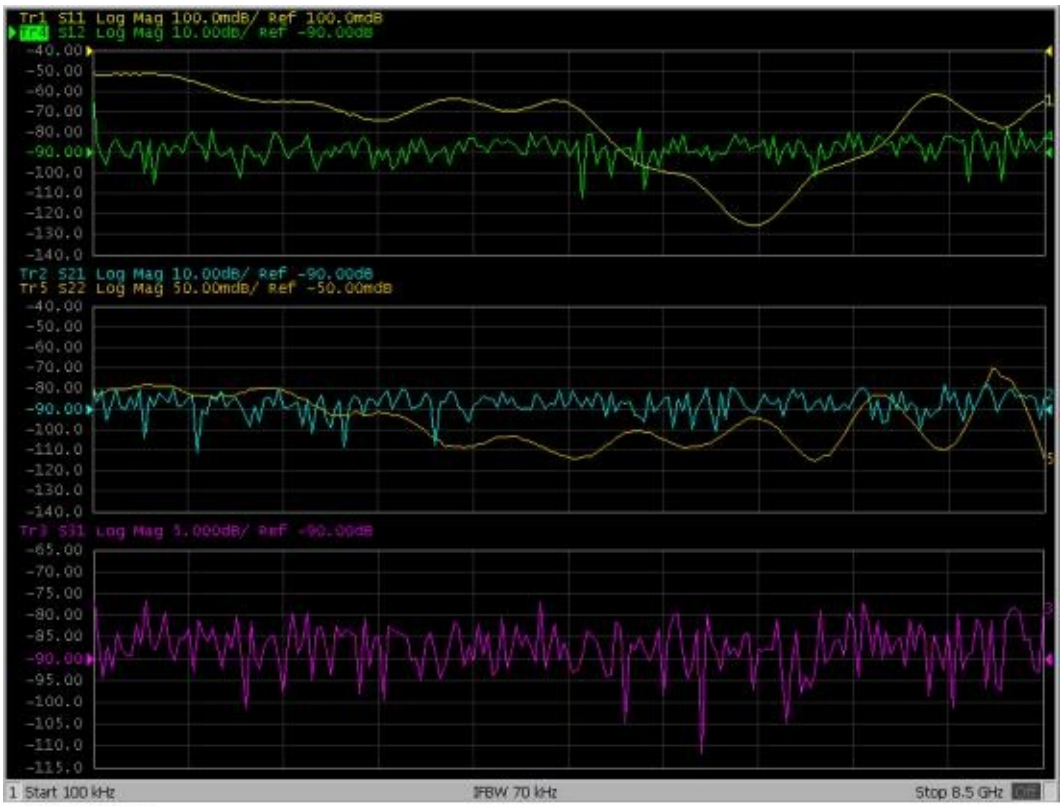
e5071c061

3 1 3

1. “ **Channel Next**”“ **Channel Prev**”
2. “ **Display**”>“ **Number of Traces**”
3.

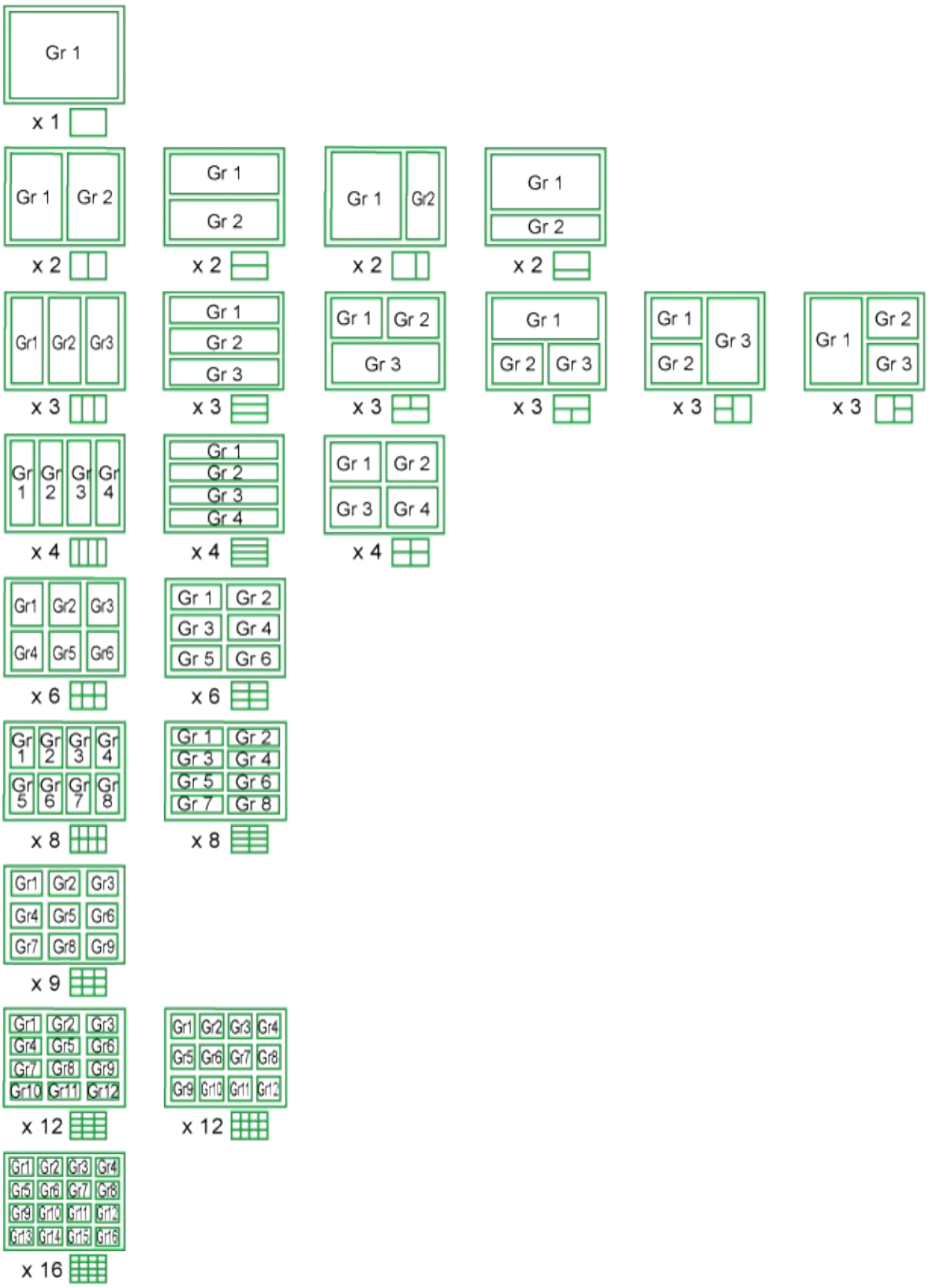
1

5 1 Gr1 1 4
2 Gr2 2 5 3 Gr3 3



e5071c059

1. “ **Channel Next**”“ **Channel Prev**”
2. “ **Display**”>“ **Allocate Traces**”
3.


Gr 1
x 1
Gr 1 Gr 2
x 2
Gr 1
Gr 2
x 2
Gr 1 Gr2
x 2
Gr 1
Gr 2
x 2
Gr1 Gr2 Gr3
x 3
Gr 1
Gr 2
Gr 3
x 3
Gr 1 Gr 2
Gr 3
x 3
Gr 1
Gr 2 Gr 3
x 3
Gr 1 Gr 3
Gr 2
x 3
Gr 1 Gr 2
Gr 3
x 3
Gr 1 Gr 2 Gr 3 Gr 4
x 4
Gr 1
Gr 2
Gr 3
Gr 4
x 4
Gr 1 Gr 2
Gr 3 Gr 4
x 4
Gr1 Gr2 Gr3
Gr4 Gr5 Gr6
x 6
Gr 1 Gr 2
Gr 3 Gr 4
Gr 5 Gr 6
x 6
Gr 1 Gr 2 Gr 3 Gr 4
Gr 5 Gr 6 Gr 7 Gr 8
x 8
Gr 1 Gr 2
Gr 3 Gr 4
Gr 5 Gr 6
Gr 7 Gr 8
x 8
Gr1 Gr2 Gr3
Gr4 Gr5 Gr6
Gr7 Gr8 Gr9
x 9
Gr1 Gr2 Gr3
Gr4 Gr5 Gr6
Gr7 Gr8 Gr9
Gr10 Gr11 Gr12
x 12
Gr1 Gr2 Gr3 Gr4
Gr5 Gr6 Gr7 Gr8
Gr9 Gr10 Gr11 Gr12
x 12
Gr1 Gr2 Gr3 Gr4
Gr5 Gr6 Gr7 Gr8
Gr9 Gr10 Gr11 Gr12
Gr13 Gr14 Gr15 Gr16
x 16

e5071c471

Channel Next	
Channel Prev	

Tr3 ▶

Trace Next	
Trace Prev	

		x		**Start Stop Center Span**
CW		x		“**Sweep Setup**”>“ **Power**”
/		x		“**Sweep Setup**”>“ **Sweep Time”/“Sweep Delay**”/
		x		“**Sweep Setup**”>“ **Points**”
		x		“**Sweep Setup**” >“**Sweep Type**” “**Edit Segment Table”/“Segment Display**” /
		x		“**Sweep Setup**”>“ **Sweep Mode**”
	x			“**Trigger**”>“ **Trigger Source**” /“ **Restart**”/“ **Trigger**”
		x		“**Trigger**”>“ **Hold**”/“ **Hold All Channels**”/“ **Single**” /“ **Continuous**”/“ **Continuous Disp Channels**”

			x	**Meas**
			x	**Format**
			x	**Scale**
			x	“**Display**”>“ **Display**”/“ **Data -> Mem**” -> /“ **Data Math**”
		x		“**Display**”>“ **Edit Title Labe**” /“ **Title Label (ON/OFF)**”/
		x		“**Display**”>“ **Graticule Label (ON/OFF)**” /
	x			“**Display**”>“ **Invert Color**”
/	x			“**Display**”>“ **Frequency (ON/OFF)**”/
/	x			“**Display**”>“ **Update (ON/OFF)**”/
		x		“**Avg**”>“ **Averaging Restart**” /“ **Avg Factor**”/“ **Averaging (ON/OFF)**”/
			x	“**Avg**”>“ **Smo Aperture**” /“ **Smoothing (ON/OFF)**” (/)
IF		x		“**Avg**”>“ **IF Bandwidth**”IF
		x		**Cal**
			x	“**Marker**”“ **Marker Search**” “ **Maker Fctn**”

		x		“**Analysis**”>“ **Fixture Simulator**”
			x	“**Analysis**”>“ **Gating**” /“ **Transform**”
			x	“**Analysis**”>“ **Conversion**”
			x	“**Analysis**”>“ **Limit Test**”
	x			**Save/Recall/**
	x			“**Macro Setup**”“ **Macro Run**” “ **Macro Break**”
// /GRIB /// ///	x			**System**
	x			**Preset**

Z 0

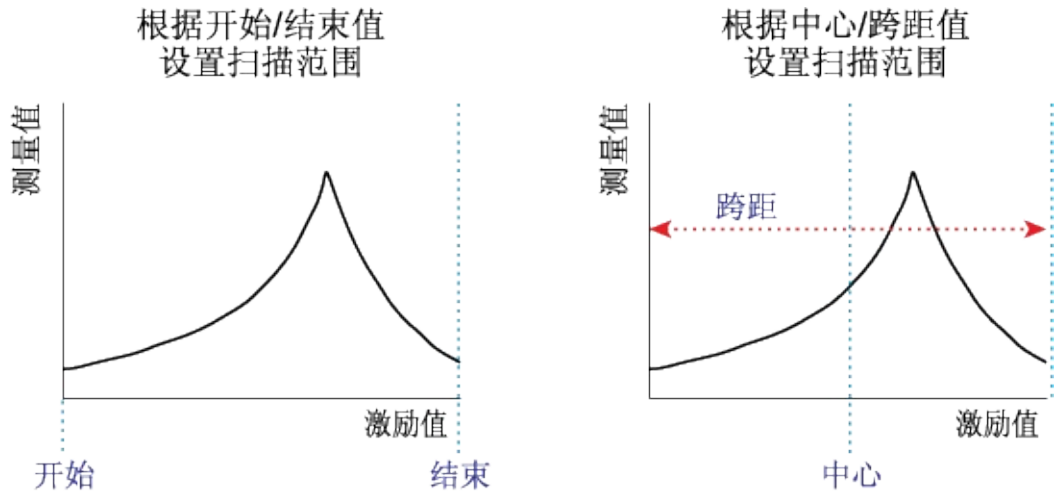
(Z 0)

1. “ **Cal**”
2. “**Set Z_0**” Z **0** Z 0

-
-
-
-
-
-
-

Linear Freq	
Log Freq	
Segment	_____
Power Sweep	

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”>“ **Sweep Type**”
3.



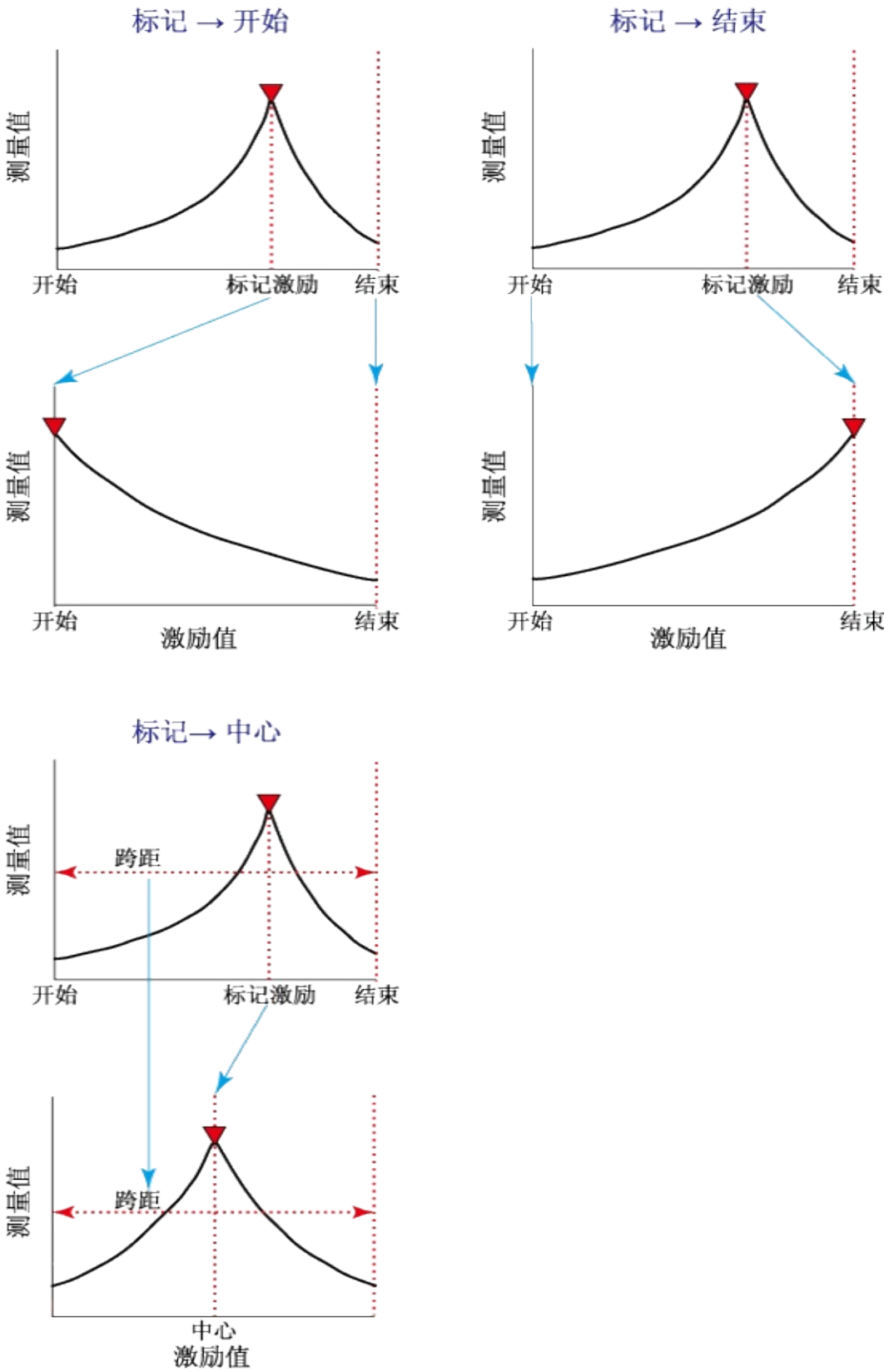
e5071c443

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Start**”
3. “ **Stop**”

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Center**”
3. “ **Span**”

1.
2. “ **Marker Fctn**”
3.

“Marker” -> “Start”	
“Marker” -> “Stop”	
“Marker” -> “Center”	


标记 → 开始
测量值
开始
标记激励
结束
测量值
开始
激励值
结束
标记 → 结束
测量值
开始
标记激励
结束
测量值
开始
激励值
结束
标记→ 中心
测量值
跨距
开始
标记激励
结束
测量值
跨距
中心
激励值

e5071c395

/

/

1. “ **Sweep Setup**”
2. “ **Power**”>“ **RF Out**”RF

“ RF OFF”RF

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”
3. “ **Power**”>“ **Port Couple**”/

ON	
OFF	

- 1
- / 1

4.

-
  a. “ **Power**”
-
  a. “ **Port Power**” 1 4
  b.

DUT

/

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”
3. “ **Power**”>“ **Slope [OFF]**” []“ **Slope [ON]**” [ ]/

1 GHz

1. “ **Channel Next**”/“ **Channel Prev**”

2. “ **Sweep Setup**”
3. “ **Power**”>“ **Slope [xxx dB/GHz]**” [xxx dB/GHz]
   “ xxx”
4. “ **ENTRY**”

CW

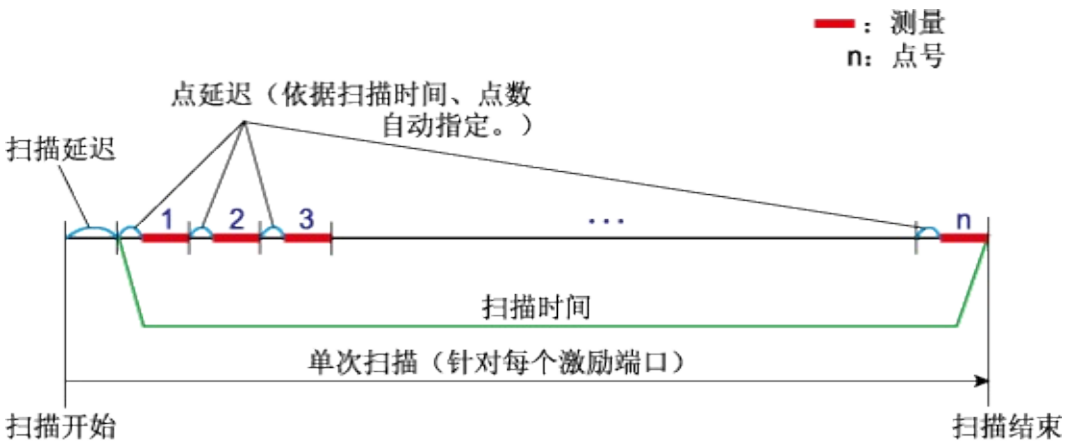
1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”
3. “ **Power**”>“ **CW Freq**”CW

2 1601

-
-
-

“ **1Ch / 4 Tr 20001 Points**”1 /4 20001
20001

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”
3. “ **Points**”



e5071c396

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup** ”“ **Sweep Time**”
3. “ **ENTRY**”

1. “ **Channel Next**”/“ **Channel Prev**”

2. “ **Sweep Setup**” “ **Sweep Time**” **0** > **x1**

E5071C DUT

- S
- Absolute
- AUX InputAUX

1. “ **Channel Next**”“ **Channel Prev**”“ **Trace Next**”“ **Trace Prev**”
2. “ **Meas**”
3.

S

S DUT S S

S

DUT

DUT

S S $_{21}$ DUT 2 DUT 1

- S -

Absolute

Absolute

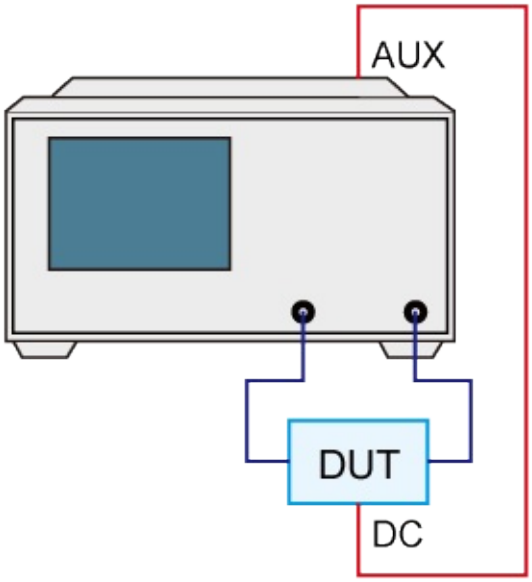
1	R1	A
2	R2	B
3	R3	C
4	R4	D

R1(1) 1 A(1) 1

AUX InputAUX

AUX DUT E5071C

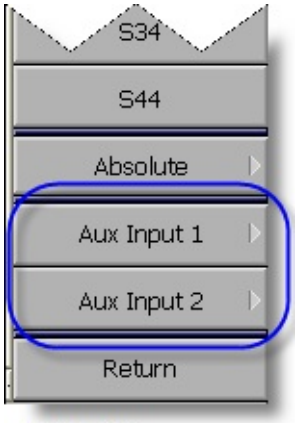
AUX



e5071c209

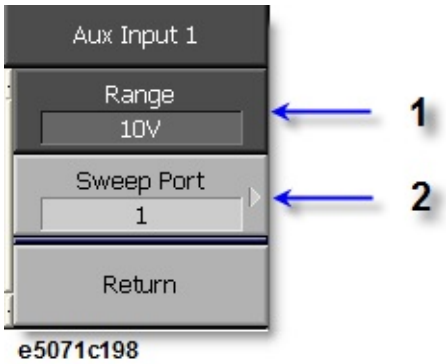
AUX

1. “ **Meas**”“ **Aux Input 1**”Aux 1“ **Aux Input 2**”Aux 2 Aux



e5071c197

2. “ **Range**” *1V* *10 V*[**1**]



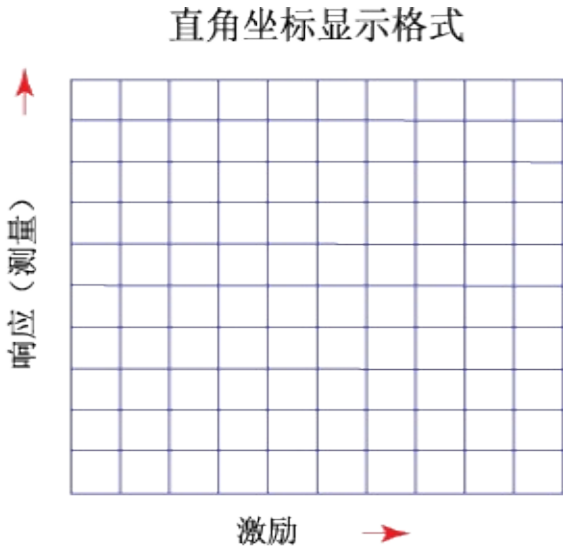
3. “ **Sweep Port**” *1-4*[**2**]
4. “ **Format**” >“ **Real**”
5.

“Sweep Port”[**2**] 1 1 RF AUX

E5071C S

-
-
-

X Y Y



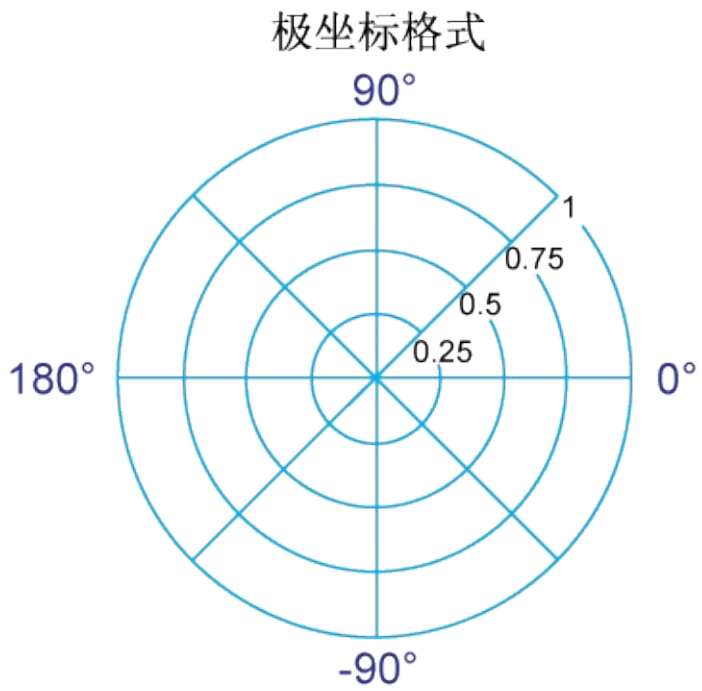
e5071c451

	Y	Y	
		dB	• •
	-180° +180 °	(°)	
	+180 ° -180°	(°)	
	0 ° +360 °	(°)	
	DUT	(s)	

SWR	$\frac{1+\rho}{1-\rho}$ ρ		

X

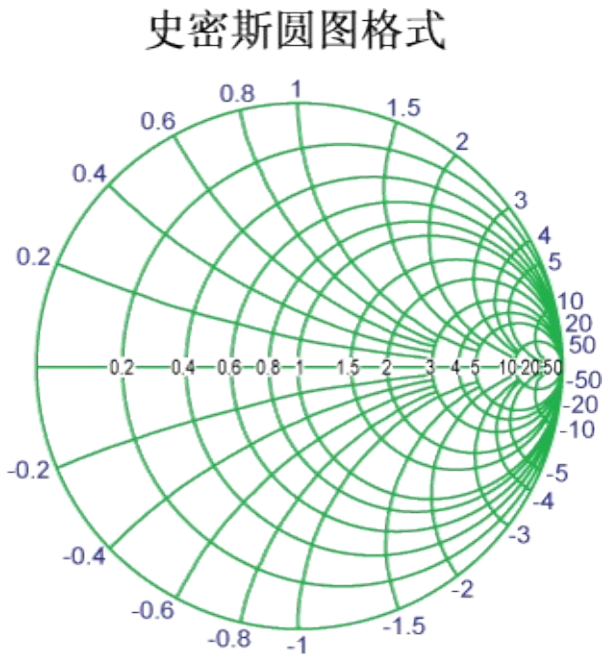
- (°)
- (°)
-



e5071c382

DUT

- (°)
- (°)
-
- (Ω) (Ω) (H) (F)
- (S) (S) (F) (H)



e5071c383

1. “ **Channel Next**”“ **Channel Prev**”“ **Trace Next**”“ **Trace Prev**”
2. “ **Format**”

3.

Log Mag	
Phase	
Group Delay	
Smith - Lin / Phase - /	
Smith - Log / Phase - /	
Smith - Real / Imag - /	
Smith - R + jX - R + jX	
Smith - G + jB - G + jB	
Polar - Lin / Phase - /	
Polar - Log / Phase - /	
Polar - Real / Imag - /	
Lin Mag	
SWR	SWR
Real	

Imaginary	
Expand Phase	
Positive Phase	

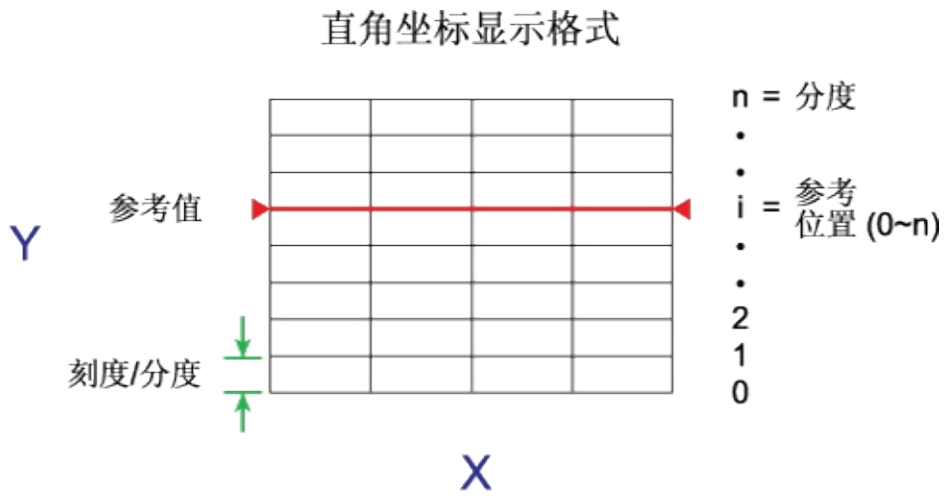
-
-

/

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Scale**”>“ **Auto Scale**”

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Scale**”>“ **Auto Scale**”

	Y 4 30
/ (Scale/Div)	Y
	Y 0
	Y



e5071c385

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”

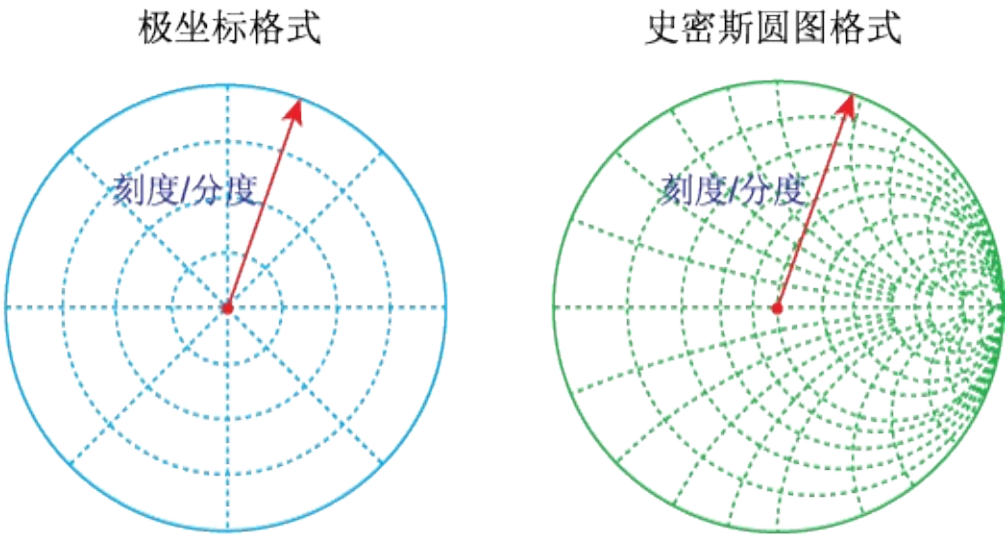
2. “ **Scale**”

3.

Divisions	Y
Scale/Div/	Y
Reference Position	
Reference Value	

/

“ **Scale/Div**”/



e5071c384

/

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Scale**”
3. “**Scale/Div**”/

1.
2. “ **Scale**”“ **Marker Fctn**”
3. “ **Marker -> Reference**” ->
4.

- /
-
-
-
-
-

/

1. “ **Channel Next**” “ **Channel Prev**”
2. “ **Channel Max**”
3. “ **Channel Max**”

1. “ **Channel Next**” “ **Channel Prev**”
2. “ **Trace Next**”“ **Trace Prev**”
3. “ **Trace Max**”
4. “ **Trace Max**”

1. “ **Channel Next**”“ **Channel Prev**”
2. “ **Display**”
3. “ **Graticule Label**”

1. “ **Display**”
2. “ **Frequency**”

“ **Display**”>“ **Frequency**”“ **Stimulus**”
“ **Start**”“ **Stop**”“ **Center**”“ **S**
“ **Softkey On/Off**”/

“Stimulus” Hz (***)

1. “ **System**”
2. “ **Service Menu**”“ **Security Level**”

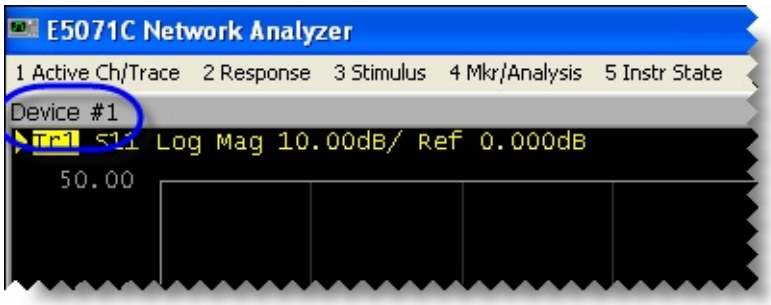
None	
Low	“Security Level”
High	“Security Level” “Preset”“Recall”

/

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Display**”
3. “ **Edit Title Label**”

e5071c060

3. **Enter**
4. “ **Title Label**”



e5071c195

LCD

1. “ **Display**”
2. “ **Invert Color**”“OFF”“ON”

- /
-
-
-

(R) (G) (B) 6 0 5 RG B 216
RG B

	R	G	B		R	G	B		R	G	B
	5	5	5		2	2	2		0	0	0
	5	3	3		5	0	0		2	0	0
	5	5	3		5	5	0		2	2	0
	3	5	3		0	5	0		0	2	0
	3	5	5		0	5	5		0	2	2
	3	3	5		0	0	5		0	0	2
	5	3	5		5	0	5		2	0	2

1. “ System”>“ Misc Setup”>“ Display Setup” >“Color Setup”
2. “ Normal”“ Invert”
3.

Data Trace 1 to 16 1 16	1 16
Mem Trace 1 to 16 1 16	1 16
Graticule Main	
Graticule Sub	
Limit Fail	
Limit Line	
Background	

4. “ Red”“ Green”“ Blue”
5. 0 5

1. “ System”>“ Misc Setup”>“ Display Setup” >“Color Setup”
2. “ Normal”“ Invert”

3. “ **Reset Color**” >“**OK**”

“Small”“Normal”“Large”

1. “ **System**”>“ **Misc Setup**”>“ **Display Setup**” >“**Magnification**”
2. **“Normal”“Small” “Large”**

-
-
-

-
- /
-
-
- 1
- 2
- 3
- 4

ECal

- ECal
- ECal
- 2 ECal 1
- 4 ECal

Ecal 2

- 2 ECal 2
- 2 ECal 3 4

ECal

- ECal
- ECal
- ECal
- ECal

-
-
-
-
-
-
-
-
-

-
-

TRL

- 2 TRL
- 3 TRL
- 4 TRL

-
-
-
-

-
-
-

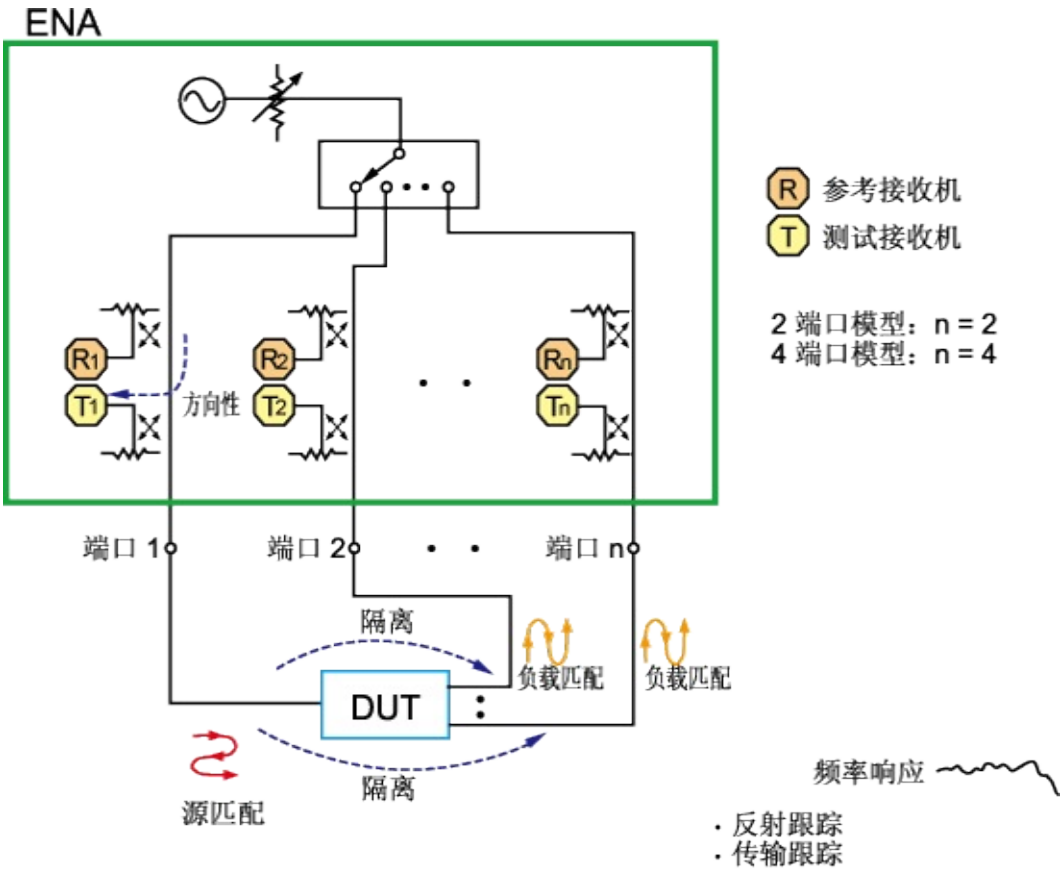
DUT IF

RF
E5071C RF

-
-
-
-
-
-

E5071C

E5071C



e5071c240

(Ed)

T1 DUT E5071C

Ed1	1
Ed2	2
Ed3	3
Ed4	4

(Ex)

DUT E5071C

Ex21Ex31 Ex41	1
Ex12Ex32 Ex42	2
Ex13Ex23 Ex43	3
Ex14Ex24 Ex34	4

(Es)

DUT DUT E5071C

Es1	1
Es2	2
Es3	3
Es4	4

(El)

DUT E5071C

El21El31 El41	1

El12El32 El42	2
El13El23 El43	3
El14El24 El34	4

(Er)

E5071C

Er1	1
Er2	2
Er3	3
Er4	4

(Et)

E5071C

Et21Et31 Et41	1
Et12Et32 Et42	2
Et13Et23 Et43	3

Et14Et24 Et34	4

				• •
	• •	2 • (Er) • (Ed)	S111	• • • DUT
	• •	2 • (Et) • (Ex)	S21 2 1	• • •
1	ECal 2 /4	3 • (Ed) • (Es) • (Er)	S11 1	• 1 •
	• • •			• 1
	Ecal 2	5		• 2

		• (Ed1) • (Ex21) • (Es1) • (Et21) • (Er1)	S11 S212 1 /	•
	•			• 2
2	ECal 2 /4	12 • Ed1Ed2 • Ex21 Ex12 • Es1Es2 • El12 El21 • Et21 Et12 • Er1Er2	S11S21S12S22 2 S	• 2
	•			• 2
3	ECal 2 /4	27 • Ed1Ed2 Ed3 • Ex21 Ex31 Ex12 Ex32 Ex13 Ex23 • Es1Es2 Es3 • El21 El31El12	S11S21S31S12S22S32 S13S23S33 3 S	• 3
	•			• 3

	• •	El32El13 El23 • Et21 Et31Et12 Et32Et13 Et23 • Er1Er2 Er3		
3	• • • •	3	S11S21S31S12S22[S32] S13[S23]S33	• 3 •
4	ECal 2 /4	48 • Ed1Ed2 Ed3Ed4 • Ex21 Ex31 Ex41 Ex12 Ex32 Ex42 Ex13 Ex23 Ex43 Ex14 Ex24 Ex34 • Es1Es2 Es3Es4 • El1El2 El3El4 •	S11S21S31S41S12S22 S32S42 S13S23S33S43S14S24 S34S44 4 S	• 4 •
	• • • •			• 4

		Et21 Et31Et41 Et12Et32 Et42Et13 Et23Et43 Et14 Et24Et34 • Er1Er2 Er3Er4		
4	• • • •	4	S11S21S31[S41]S12S22 [S32] [S42]S13[S23]S33S43 [S14][S24]S34S44	• 4 •
2 **TRL**	• • • •	12 • Ed1Ed2 • Es1Es2 • El1El2 • Et21 Et12 • Er1Er2	S11S21S12S22 2 S	• 2 •
3 **TRL**	• •	27 • Ed1Ed2 Ed3 • Es1Es2 Es3 • El21 El31El12 El32El13	S11S21S31S12S22S32 S13S23S33 3 S	• 3

	• •	El23 • Et21 Et31Et12 Et32 Et13Et23 • Er1Er2 Er3		•
3 **TRL**	• • • •	3 TRL	S11S21S31S12S22[S32] S13[S23]S33	• 3 • //
4 **TRL**	• • • •	48 • Ed1Ed2 Ed3Ed4 • Es1Es2 Es3Es4 • El1El2 El3El4 • Et21 Et31Et41 Et12Et32 Et42 Et13Et23 Et43Et14 Et24Et34 • Er1Er2 Er3Er4	S11S21S31S41S12S22 S32S42S13 S23S33S43S14S24S34 S44 4 S	4
4	•		S11S21S31[S41]S12S22	• 4

TRL	• • •	4 TRL	[S32][S42]S13[S23]S33 S43[S14][S24]S34S44	• //

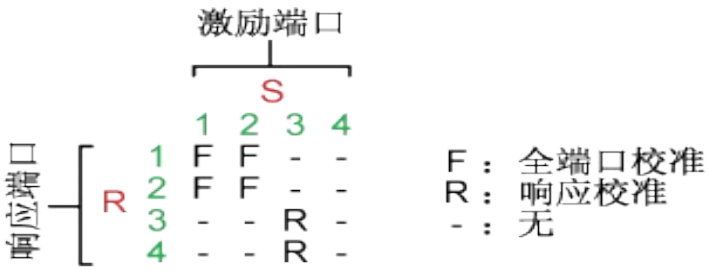
-
-
-

Cor	
Cor	
Off	

C?	IF
C!	

RO	
RS	
RT	
ER	
F1	1
F2	2 /2 TRL
F3	3 /3 TRL
F4	4 /4 TRL

1 2 2 3 4 3



e5071c355

3/4 3/4 TRL TRL **F**

/

/

1. “ **Channel Next**”/“ **Channel Prev**”/
2. “ **Cal**”
3. “ **Property**”/

-
- S S “ **Done**” S 4 1 2 3 2 1 2 2 3

-
-

(m) (f) (m) (f)

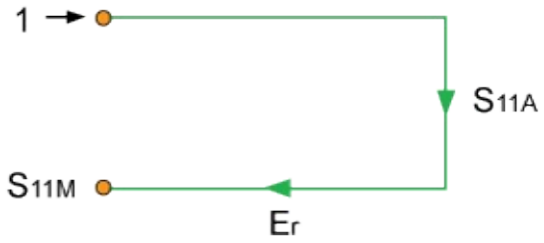
1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Cal Kit**”

-
- (*)

/

-
-

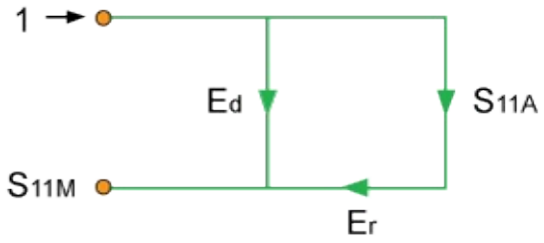
1 /



Er：反射跟踪误差

e5071c398

1 / +

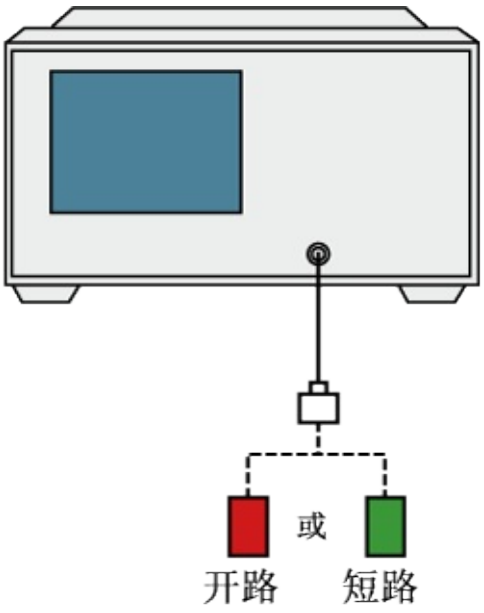


Er：反射跟踪误差
Ed：方向性误差

e5071c399

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”
4.
5. “ **Select Port**”
6. /
7. DUT
8. “ **Open**”“ **Short**”
9.
10. DUT
11. “ **Load (Optional)**”
12. “ **Done**”

/



e5071c318

-
-

2

Et：传输跟踪误差

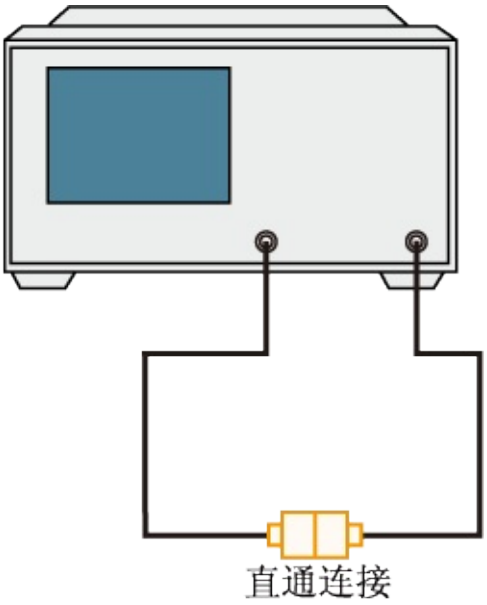
e5071c400

2 +

Et：传输跟踪误差
Ex：隔离（串扰）误差

e5071c401

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”>“ **Response (Thru)**”>“ **Select Port**”
4. S
5. DUT



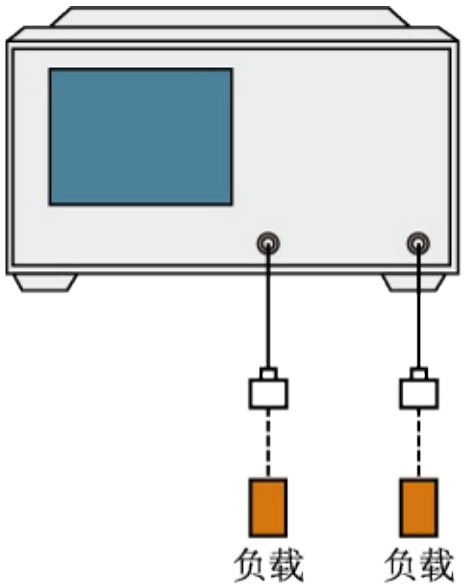
e5071c317

6. “ **Thru**”

注意

7.

   a. DUT



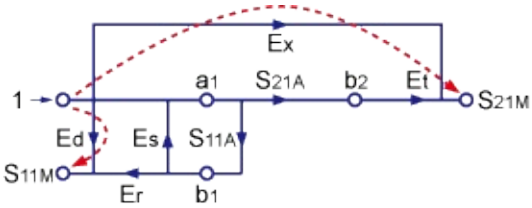
e5071c319

b. “ **Isolation (Optional)**”

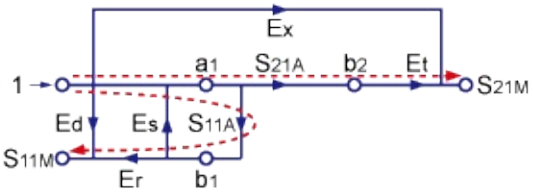
8. “ **Done**”

-
-

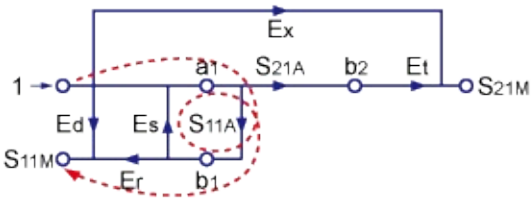
2



隔离



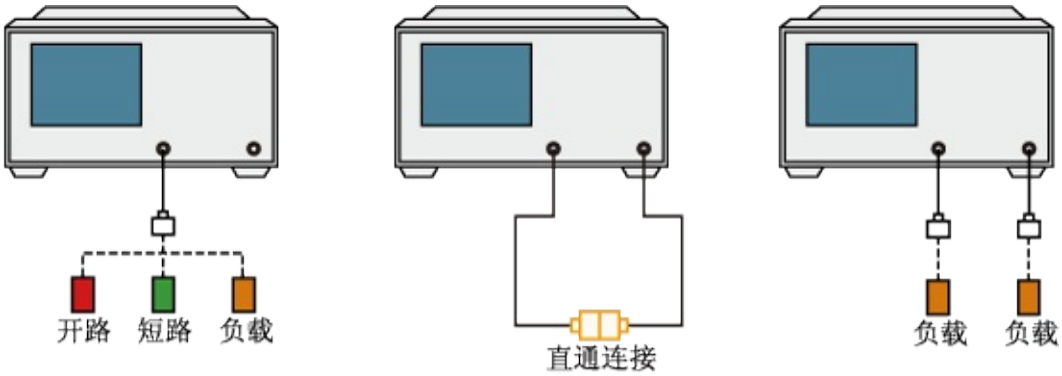
频率响应



反射

Ed：方向性误差
Es：源匹配误差
Er：反射跟踪误差
Et：传输跟踪误差
Ex：隔离（串扰）误差

e5071c301



e5071c332

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”
4. “ **Enhanced Response**”
5. “ **Ports**”
6.
7. “ **Open**”
8.
9. “ **Short**”
10.
11. “ **Load**”
12.
13. “ **Thru**”
14.
    a.

b. “ **Isolation (Optional)**”

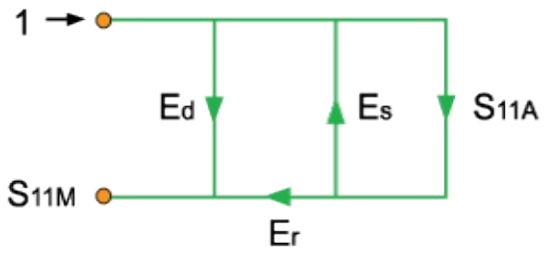
c. “ **Return**”

15. “ **Done**”

1

-
-

1



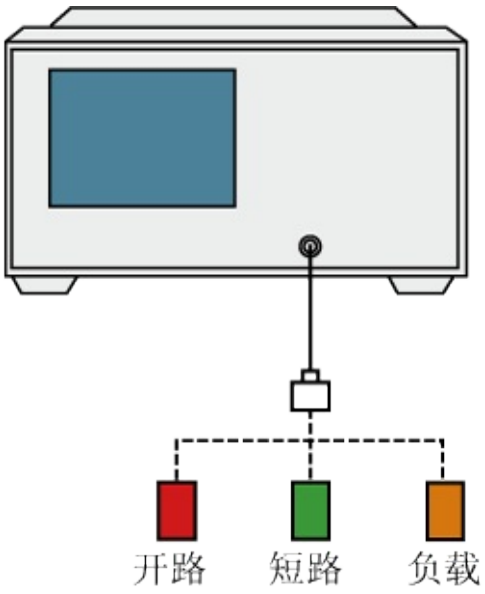
Er：反射跟踪误差
Ed：方向性误差
Es：源匹配误差

e5071c402

1 1

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”>“ **1-Port Cal**”1 >“ **Select Port**”
4. 1 S
5. DUT
6. “ **Open**”
7. DUT
8. “ **Short**”
9. DUT
10. “ **Load**”
11. “ **Done**” 1

1

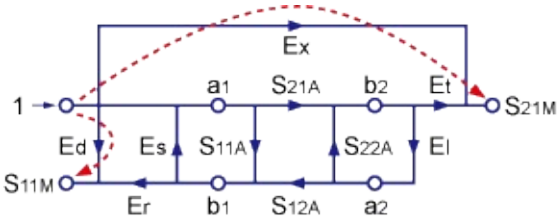


e5071c300

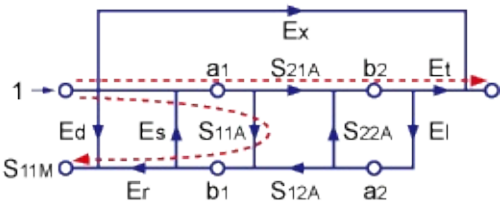
2

-
-

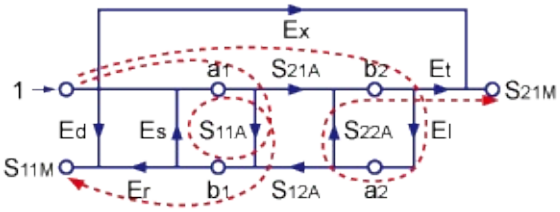
2



隔离



频率响应



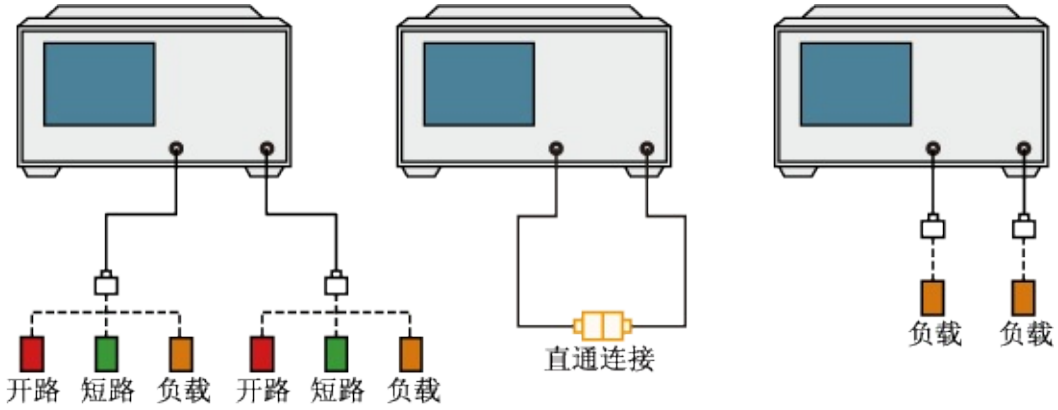
反射

Ed：方向性误差
Es：源匹配误差
Er：反射跟踪误差
Et：传输跟踪误差
Ex：隔离（串扰）误差
El：负载匹配误差

e5071c303

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”>“ **2-Port Cal**”2
4. “ **Select Ports**” 2 x y
5. “ **Reflection**”
6. xDUT
7. “ **Port x Open**” x **x**
8.
9. “ **Port x Short**” x **x**
10.
11. “ **Port x Load**” x **x**
12. y
13. “ **Return**”
14. “ **Transmission**”
15. x y DUT
16. “ **Port x-y Thru**” x-y **x** **y**
17. “ **Return**”
18.
19. “ **Isolation (Optional)**”
20. DUT
21. “ **Port x-y Isol**” x-y **x** **y**
22. “ **Return**”
23. “ **Done**” 2

2

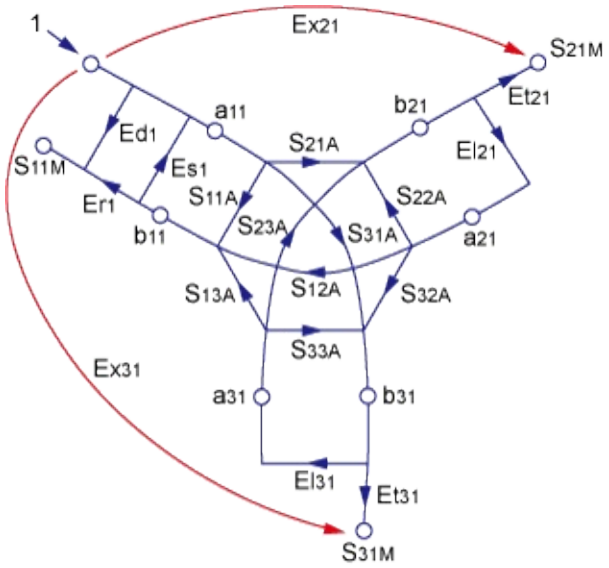


e5071c331

3

-
-

3 2 3 × 3 =
93 × 6 = 18 3 27



方向性误差：	Ed1、Ed2、Ed3
隔离误差：	Ex21、Ex31、Ex12、Ex32、Ex13、Ex23
源匹配误差：	Es1、Es2、Es3
负载匹配误差：	El21、El31、El12、El32、El13、El23
反射跟踪误差：	Er1、Er2、Er3
传输跟踪误差：	Et21、Et31、Et12、Et32、Et13、Et23

e5071c304

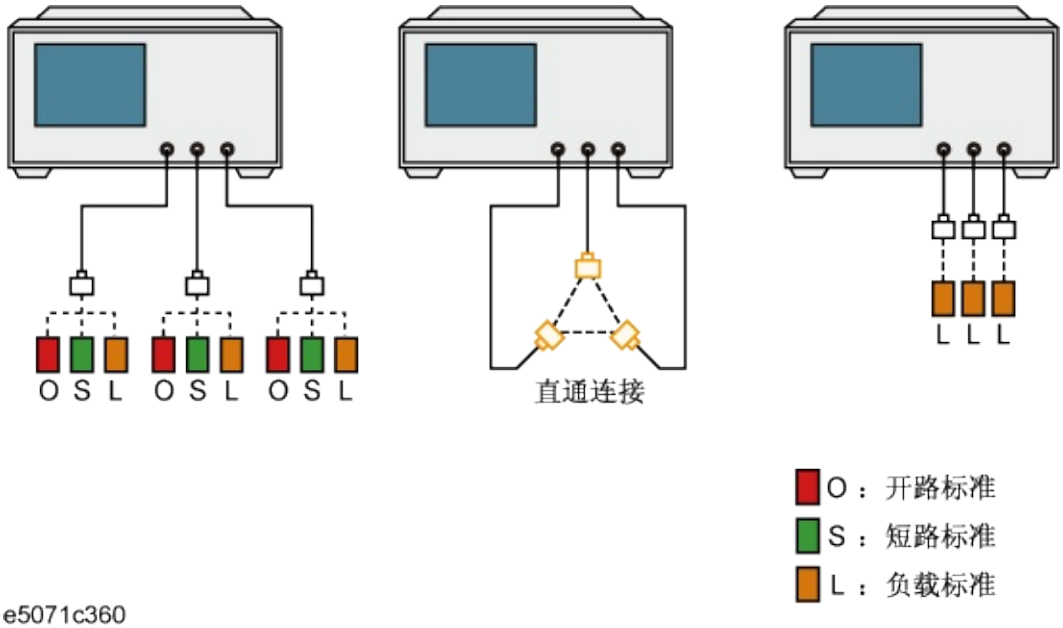
1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”>“ **3-Port Cal**”3
4. “ **Select Ports**” 3

1. “ **Reflection**”
2. xDUT
3. “ **Port x Open**” x **x**
4.
5. “ **Port x Short**” x **x**
6.
7. “ **Port x Load**” x **x**
8. y z
9. “ **Return**”

1. “ **Transmission**”
2. x y DUT
3. “ **Port x-y Thru**” x-y **x** **y**
4. x z y z
5. “ **Return**”
6.
7. “ **Isolation (Optional)**”

8. DUT
9. “ **Port x-y Isol**” x-y **x** y
10. “ **Return**”
11. “ **Done**” 3

3

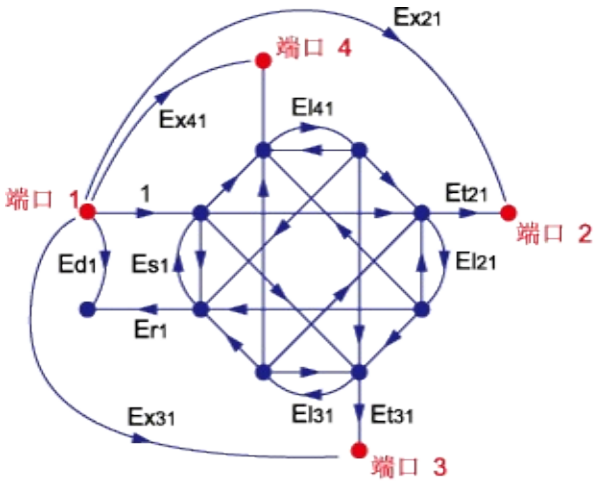


e5071c360

4

-
-

4 2 3 × 4 =
123 × 12 = 36 4 48



方向性误差： E_{d1}、E_{d2}、E_{d3}、E_{d4}
隔离误差： E_{x21}、E_{x31}、E_{x41}、E_{x12}、E_{x32}、E_{x42}、E_{x13}、E_{x23}、E_{x43}、E_{x14}、E_{x24}、E_{x34}
源匹配误差： E_{s1}、E_{s2}、E_{s3}、E_{s4}
负载匹配误差： E_{l21}、E_{l31}、E_{l41}、E_{l12}、E_{l32}、E_{l42}、E_{l13}、E_{l23}、E_{l43}、E_{l14}、E_{l24}、E_{l34}
反射跟踪误差： E_{r1}、E_{r2}、E_{r3}、E_{r4}
传输跟踪误差： E_{t21}、E_{t31}、E_{t41}、E_{t12}、E_{t32}、E_{t42}、E_{t13}、E_{t23}、E_{t43}、E_{t14}、E_{t24}、E_{t34}

e5071c306

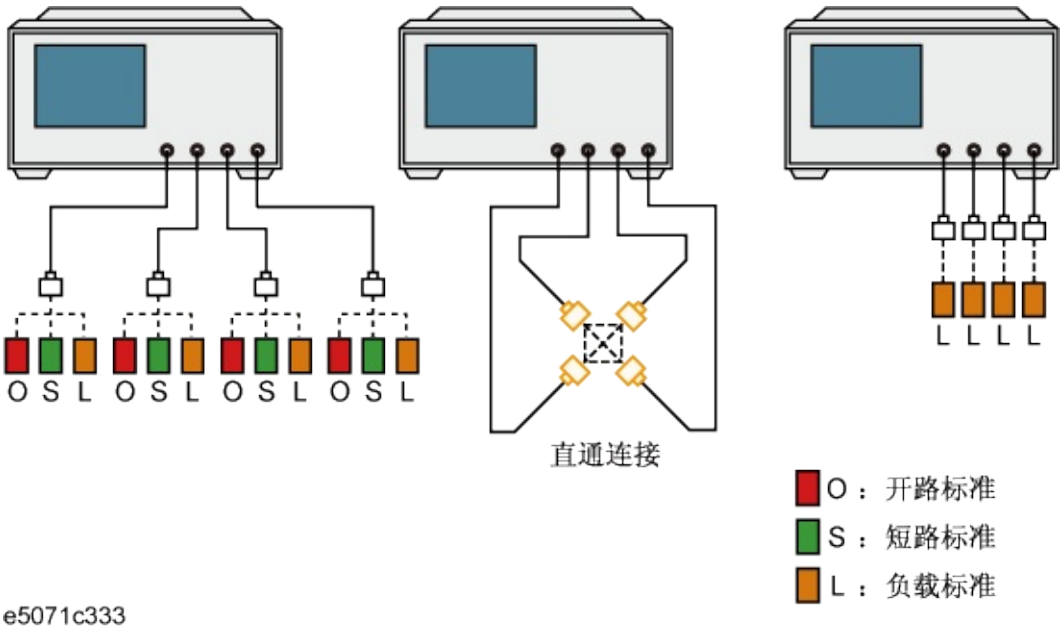
1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”>“ **4-Port Cal**”4

1. “ **Reflection**”
2. 1
3. “ **Port 1 Open**” 1
4.
5. “ **Port 1 Short**” 1
6.
7. “ **Port 1 Load**” 1
8. 23 4
9. “ **Return**”

1. “ **Transmission**”
2. 1 2
3. “ **Port 1-2 Thru**” 1-2
4. 1 3
5. 1 4
6. 2 3
7. 2 4
8. 3 4

9. “ **Return**”
10.
11. “ **Isolation (Optional)**”
12. DUT
13. “ **Port 1-2 Isol**” 1-2
14. “ **Port 1-3 Isol**” 1-3 “ **Port 1-4 Isol**” 1-4 “ **Port 2-3 Isol**” 2-3 “ **Port 2-4 Isol**” 2-4
15. “ **Return**”
16. “ **Done**” 4

4



ECal

ECal ECal

-
-
-
- ECal PIN FET

ECal

ECal

- ECal
- 2 ECal 1
- 4 ECal
- 2 ECal 2
- 2 ECal 3 4
- ECal
- ECal
- ECal
- ECal

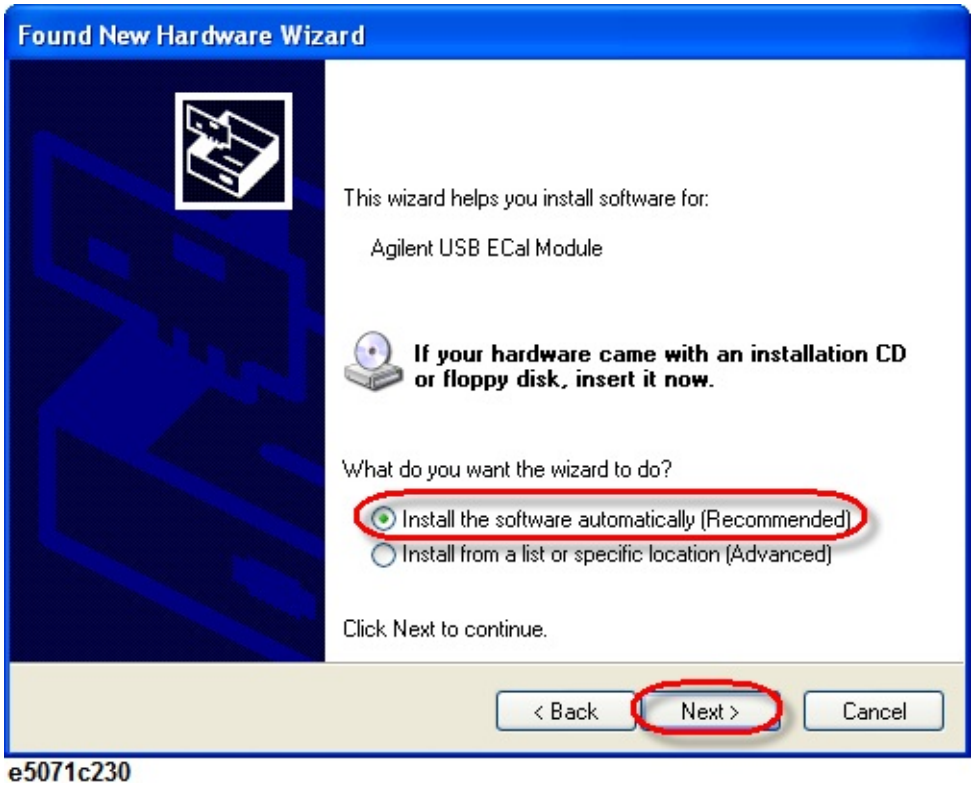
ECal

ECal

ECal USB ECal

1. Ecal E5071C USB
2. “ **No, not this time**”“ **Next**”

3. “ **Install the software automatically (Recommended)**”
   “ **Next**”



e5071c230

4. “ **Finish**”

e5071c231

USB E5091A USB

ECal

2 ECal 1

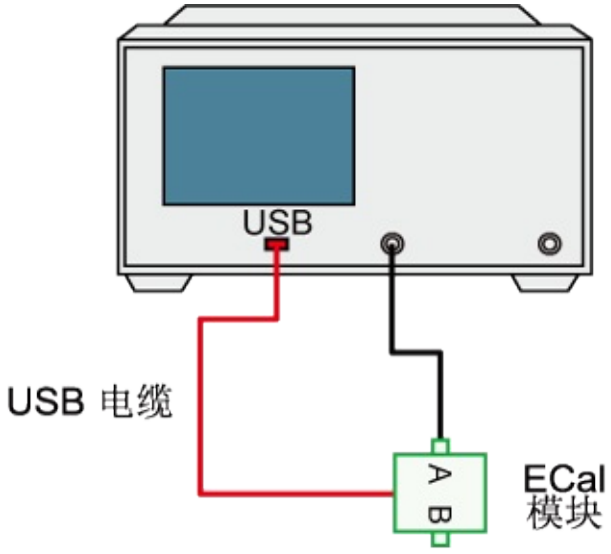
2 ECal 1

1. USB ECal USB E5071C USB E5071C

2. ECal 15 “WAIT”“READY”
3. ECal
4. ECal
5. “ **Channel Next**”/“ **Channel Prev**”
6. “ **Cal**”
7. “ **ECal**”>“ **1-Port Cal**”1
8. 1

ECal E5071C ECal

ECal 1



e5071c325

ECal

4 ECal

-
-

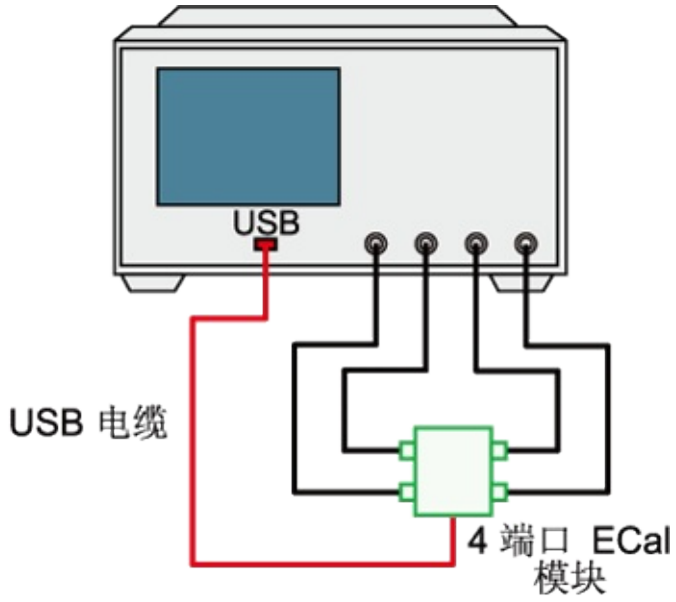
ECal

E5071C 4 ECal 2 ECal

1. 4 ECal USB E5071C USB USB E5071C

2. ECal 20 “WAIT”“READY”
3. 4 ECal
4. ECal
5. “ **Channel Next**”/“ **Channel Prev**”
6. “ **Cal**”
7. “ **Ecal**”
8. “ **Isolation**”“ **ON**”
9. .
10.
11. E5071C ECal ECal

4 ECal 4



e5071c326

ECal E5071C ECal

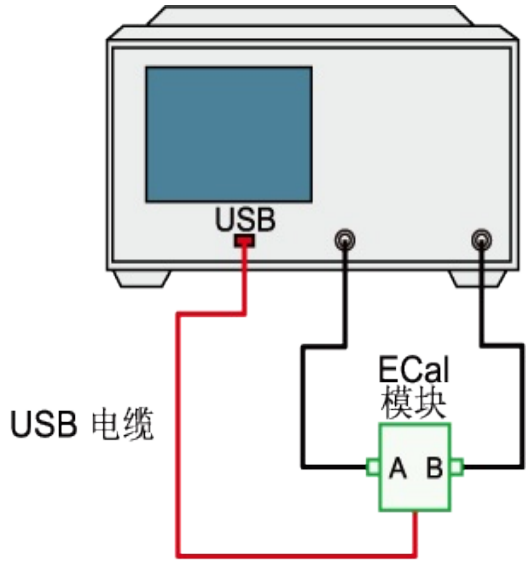
2 ECal 2

2 ECal 2

1. USB ECal USB E5071C USB E5071C

2. ECal 15 “WAIT”“READY”
3. ECal A B
4. “ **Channel Next**”/“ **Channel Prev**”
5. “ **Cal**”
6. “ **ECal**”
7. “ **Isolation**”“ **ON**”
8. “ **2 Port ECal**”2 2 E5071C 2 3 4
E5071C 2

ECal 2



e5071c322

ECal E5071C ECal

ECal

2 ECal 3 4

-
-

ECal

E5071C VBA ECal 2 ECal 3 4

ECal

1. USB ECal USB E5071C USB E5071C

2. ECal 15 “WAIT”“READY”

3. “ **Macro Setup**”

4. “ **Load Project**”

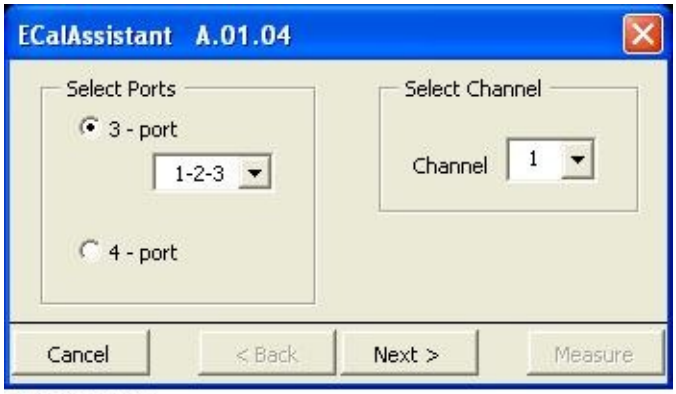
5. “Open” VBA **D:\Agilent\ECalAssistant.vba**
   “ **Open**”

6. “ **Macro Run**”

7.

e5071c085

8. “ **Next**”

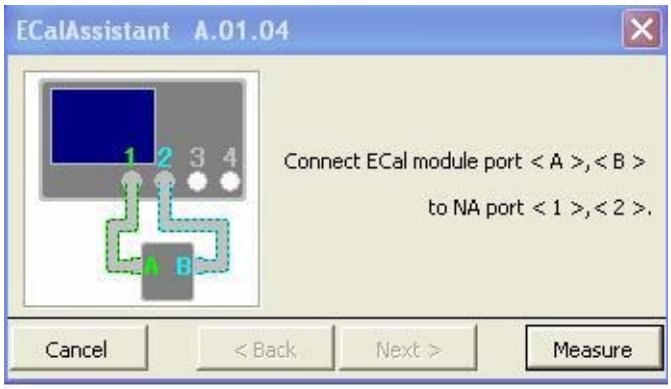
9. 

e5071c086

10. “Select Ports” 3 “ **3-Port**”3 4

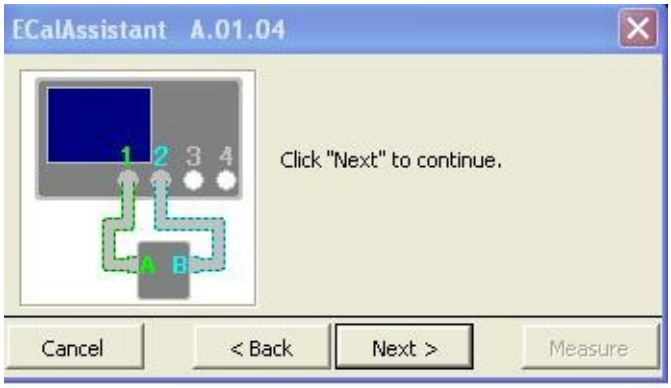
“ **4-Port**”4

11. E5071C 3 “ **3-Port**”3 “ **1-2-3**”“ **1-2-4**”“ **1-3-4**”“ **2-3-4**”
12. “Select Channel”“ **1**”“ **9**”
13. “ **Next**”
14.



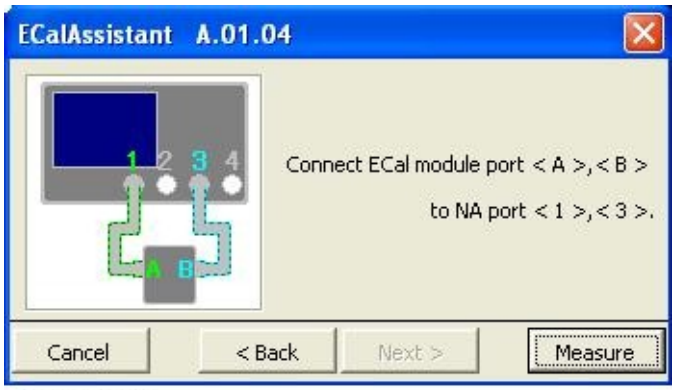
e5071c087

15. ECal A B E5071C E5071C
16. “ **Measure**”
17.



e5071c088

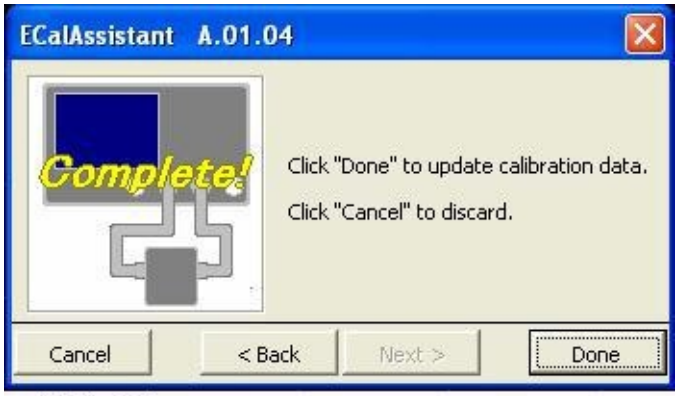
18. “ **Next**”
19.



e5071c089

20. ECal

21. “ **Complete!**”“ **Done**”
“ **Cancel**”



e5071c099

ECal

ECal

-
-

ECal

2

1. ECal 2
2.

E5071C ECal ECal ECal E5071C

1. “ **Cal**”“ **ECal**”
2. “ **Unknown Thru**”
3. ECal

ECal

-
-

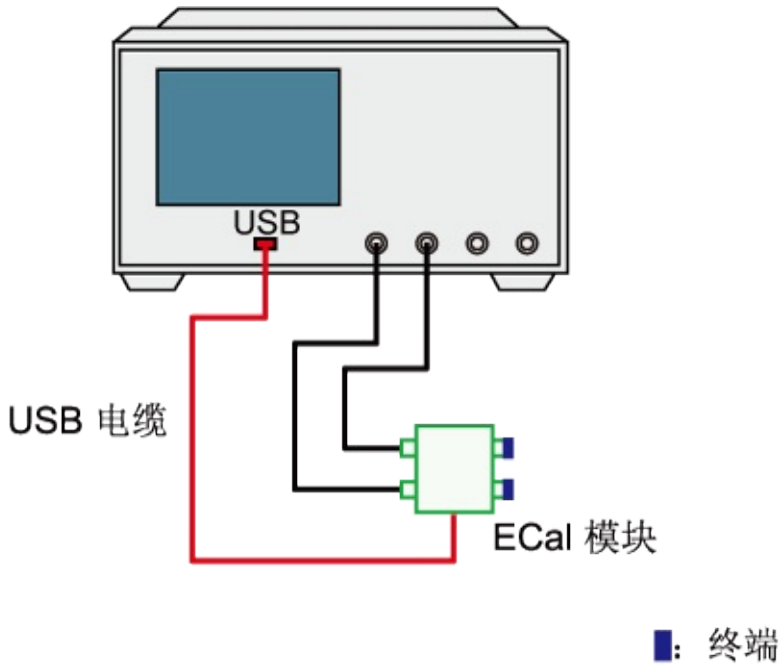
ECal

ECal E5071C

E5071C ECal ECal ECal E5071C
ECal

1. ECal USB E5071C USB USB E5071C
2. ECal 15 20 “WAIT”“READY”
3. “ **Channel Next**”/“ **Channel Prev**”
4. “ **Meas**”
5. S S
6. S E5071C S S21 1 2 ECal
7. ECal
8. “ **Cal**”
9. “ **ECal**”
10. ECal “ **Characterization**”
11. “ **Confidence Check**”
12.
13.
    a. “ **Display**”
    b. “ **Display**”>“ **Data & Mem**”
    c. “ **Scale**”
    d. “ **Auto Scale**”
    e.
14.

ECal S21



e5071c328

ECal

ECal E5071C ECal

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **ECal**”>“ **Orientation**”“ **Manual**”
4. E5071C
5. E5071C Ecal

ECal

ECal

-
- VBA
- ECal
- ECal /
- ECal

ECal

E5071C ECal ECal ECal ECal ECal

ECal ECal ECal

VBA ECal

	VBA
D:\Agilent	**EcalCharacterization.vba**

VBA

- VBA / USB

VBA ECal USB ECal

- VBA ECal ECal

ECal

ECal ECal

2/3 E5071C VBA 4 ECal

1. ECal

ECal USB E5071C USB USB E5071C

2.

IF 1 kHz

3.

2/4 ECal 2/4 ECal

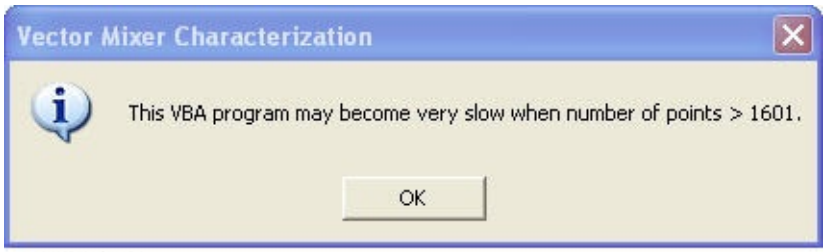
4. VBA

a. “ **Macro Setup**”

b. “ **Load Project**”

c. “Open” **D:\Agilent\EcalCharacterization.vba**
“ **Open**”

d. “ **Macro Run**” 1 4 1601 E5071C VBA
1 4 1601 “



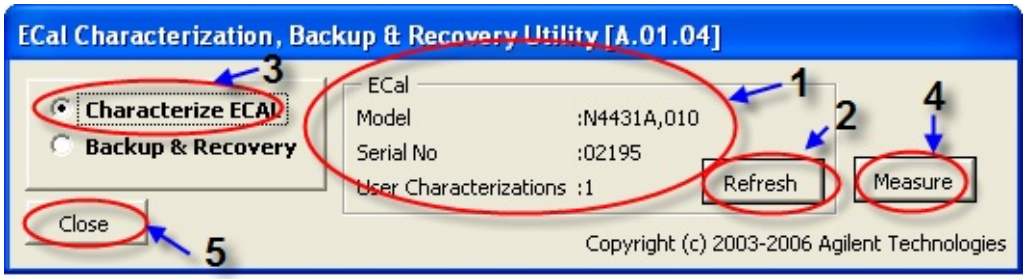
e5071c136

e. “ECal” 1 E5071C ECal

f. ECal “ **Refresh**” 2

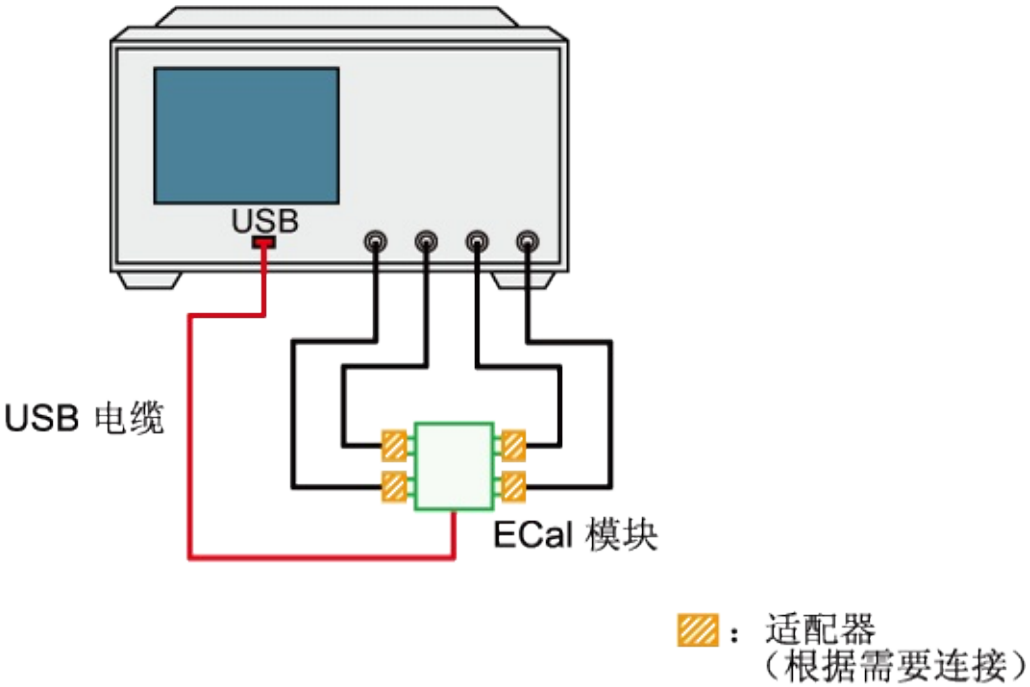
5.

a. “Characterize ECAL” ECAL 3“User Characteristic Measurement”



e5071c232

b. ECal ECal E5071C



e5071c329

c. “ **Measure**” 4

ECal E5071C E5071C

6.

a.

b.

1		“Characterization”“Connectors” “Adapter Description”
2		
3		ECal “Male”“Female”“No adapter”
4		

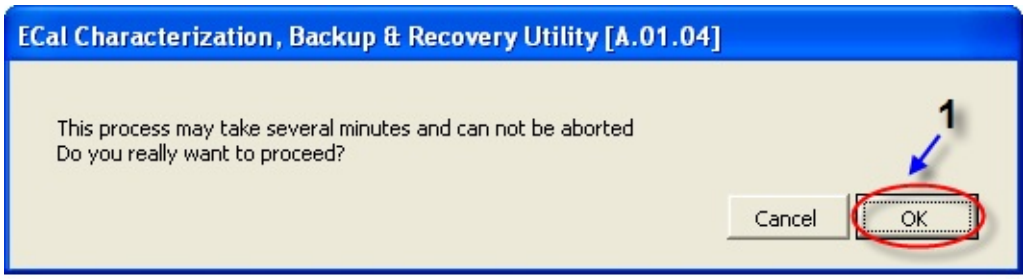
“ **Cal**”>“ **Ecal**”>“ **Characterization Info**”

d. “ **Write**”

e. “ **OK**”

注意 ECal 5“ **Write**”

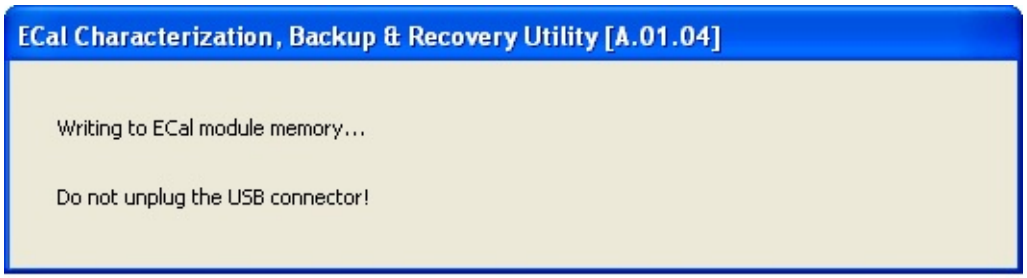
f. “ **OK**”



e5071c234

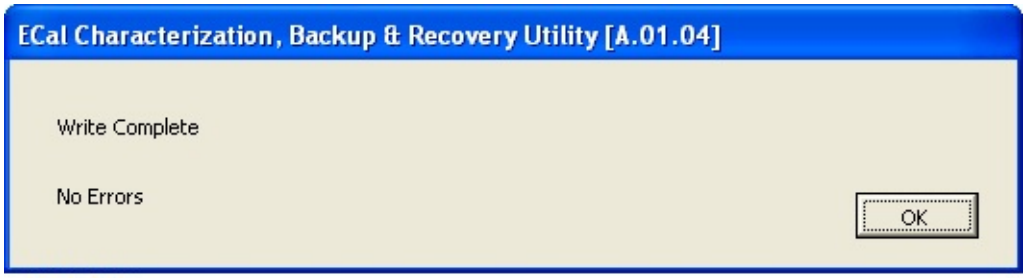
注意 VBA ECal USB VBA ECal

g. VBA



e5071c236

h. “ **OK**”



e5071c237

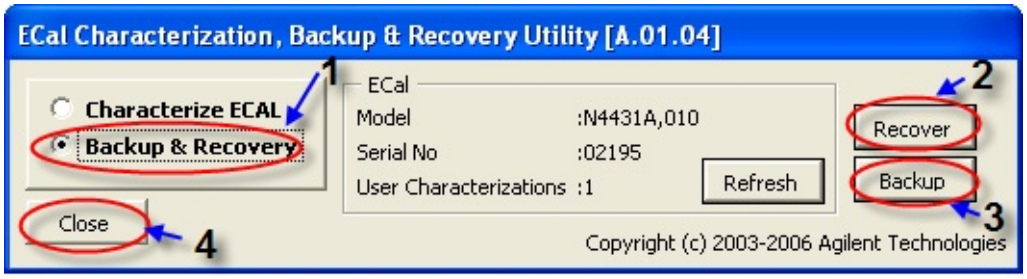
8. VBA

a. “ **Close**” 5

ECal

ECal

1. ECal USB E5071C USB USB E5071C
2. VBA VBA
3. “ **Backup Flash ROM**” Flash ROM 1“Backup”



e5071c235

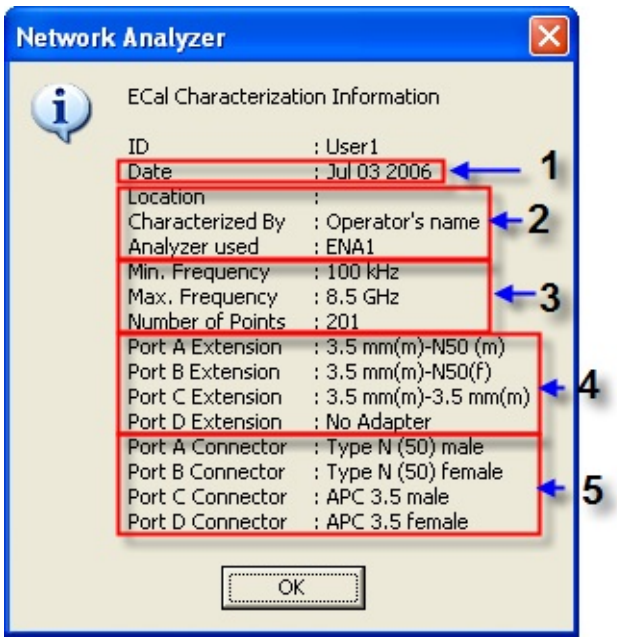
1. “ **Recover**” 2
2. “Open”“ **Open**” E5071C ECal
   “ **OK**”
3. “ **OK**” VBA
4. “Completion”“ **OK**”

1. “ **backup**”
2. “Save As”“ **Save**”
3. “ **Close**”

ECal

ECal ECal

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **ECal**”>“ **Characterization**”
4. .
5. “ **Characterization Info**”



1:	
2:	“Characterization” 1
3:	
4:	“Adapter Description” 4
5:	“Connectors” 3

-
-

“Internal”“System”“System”

“Internal”“System” PC

“System”“External”“Manual”“ON”
“ON”

- ECal
-
-
- /

1. “ **Cal**”
2. “ **Cal Trig Source**”

Internal	“Internal”
System	“System”

-
-
-
- TRL

(f) (m)

- E5071C SMA
- E5071C
- 7-mm 50.4Ω 50.0Ω

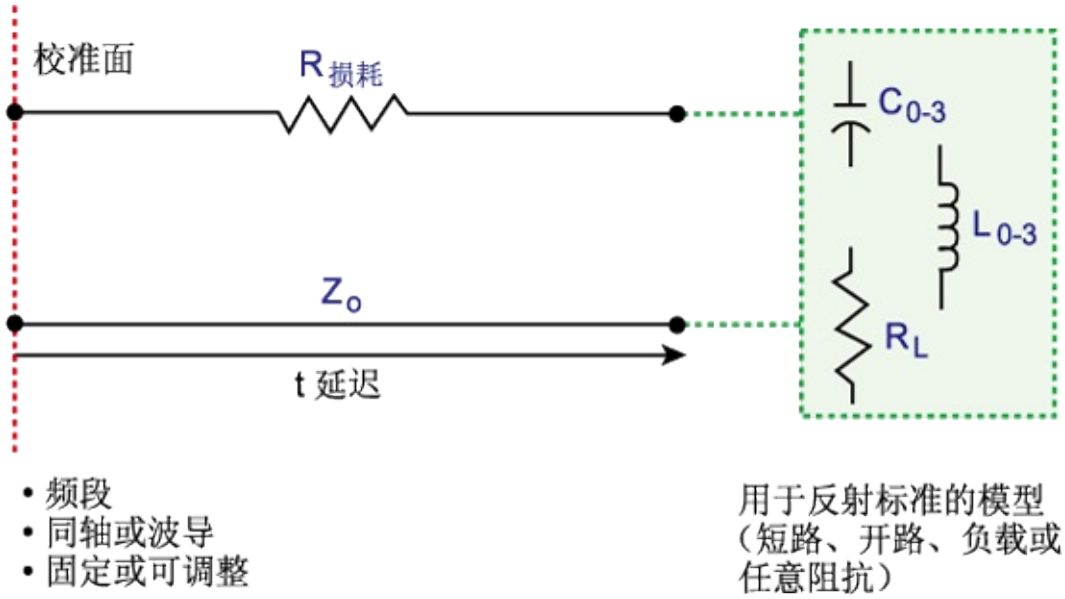
E5071C 21 1 21 85033E 3.5-mm 1

/

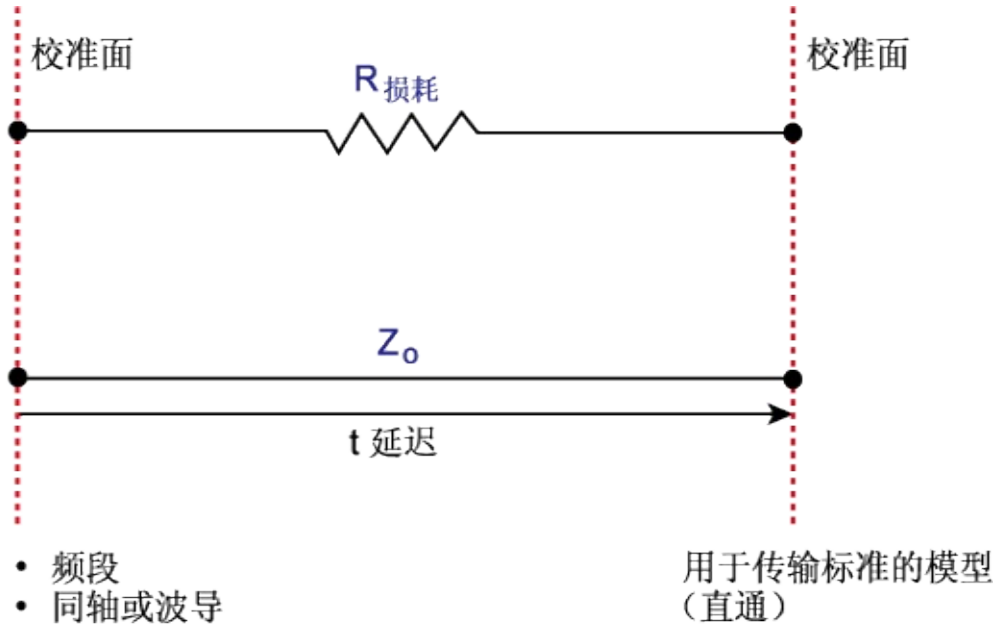
3.5-mm (32 ps)

21

8


校准面
R 损耗
C0-3
L0-3
Zo
RL
t 延迟
• 频段
• 同轴或波导
• 固定或可调整
用于反射标准的模型
（短路、开路、负载或
任意阻抗）

e5071c373



e5071c374

Z0

Z0

1 GHz Ω/s 0 1 GHz

$$Loss\left(\frac{\Omega}{s}\right) = \frac{loss(dB) \times Z_0(\Omega)}{4.3429(dB) \times delay(s)}$$

C0C1C2C3

$$C = (C0) + (C1 \times F) + (C2 \times F^2) + (C3 \times F^3)$$

F
C0 F

C1 F/Hz

C2 F/Hz ^{2}

C3 F/Hz ^{3}

L0L1L2L3

$$L = (L0) + (L1 \times F) + (L2 \times F^2) + (L3 \times F^3)$$

F

L0 [H]

L1 [H/Hz]

L2 [H/Hz ^{2}*]*

L3 [H/Hz ^{3}*]*

“”“0”

1.
2. /
3.
4.
5.

(f) (m)

1.
2.
3.

1. “ **Cal**”
2. “ **Cal Kit**”
3. “ **Modify Kit**” (f) (m)
4. “ **Define STDs**”
5. 1 21
6. “ **STD Type**”
7.
8. “ **Label**”“ **Return**”
9. “ **Return**”
10. “ **Specify CLSs**” CLS

11. “TRL Thru”TRL “ **Set All**”
12. “ **Set All**”
13. 1 21 (f) (m)
14. “ **Return**”
15. “ **Return**”
16. “ **Label Kit**”

TRL

TRL TRL

- THRU 0 ps 1.3 GΩ/s
- REFLECT 0 ps
- MATCH 0 2GHz
- LINE150-Ω 2G 7GHz 54.0 ps
- LINE250-Ω 7G 32GHz 13.0 ps

1. “ **Cal**”
2. “ **Cal Kit**”
3. “ **Modify Kit**”>“ **Label Kit [User]**”[]

1. “ **Define STDs**”
2. “ **1:No Name**”1:>“ **Label**”“ **THRU**”
3. “ **STD Type**”>“ **Delay/Thru**”/
4. “ **Offset Loss**” 1.3GΩ/s“ **Offset Delay**” 0
5. “ **Return**”“Define STD”
6. No. 2 REFLECT

7. “ **SHORT**”

1. “ **3:No Name**”3:>“ **Label**”
2. “ **MATCH <2G**”
3. “ **STD Type**”>“ **Load**”
4. “ **Max Frequency**” 2GHz
5. “ **Return**”“Define Std”

1/2

1. “ **4:No Name - Label**”4: -
2. “ **LINE <7G**”
3. “ **STD Type - Thru**” -
4. “ **Offset Delay**” 54 ps
5. “ **Min Frequency**” 2 GHz
6. “ **Max Frequency**” 7 GHz
7. “ **Return**”“Define Std”
8. “ **5:No Name - Label**”5: -
9. “ **LINE >7G**”
10. “ **STD Type**”>“ **Thru**”
11. “ **Offset Delay**” 13 ps
12. “ **Min Frequency**” 7 GHz
13. “ **Return**”“Define Std”
14. “ **Return**”“Modify Cal Kit”

// 1

1. “ **Specify CLSs**” CLS>“ **Sub Class**”

2. “ **Sub Class1**” 1
3. “ **TRL Thru**”TRL >“ **Set All**”>“ **THRU**” >“**Return**”
4. “ **TRL Reflect**”TRL >“ **REFLECT**”
5. “ **TRL Line/Match**”TRL />“ **Set All**”>“ **MATCH <2G**” <2G>“ **Return**”

1/2 2/3

1. “ **Sub Class2**” 2
2. “ **TRL Line/Match**”TRL />“ **Set All**”>“ **LINE <7G**” <7G>“ **Return**”
3. “ **Sub Class3**” 3
4. “ **TRL Line/Match**”TRL />“ **Set All**”>“ **LINE >7G**” >7G>“ **Return**”
5. “ **Cal**”

1/2 1/2/3 TRL / 3

TRL

TRL

1. “ **Cal**”
2. “ **Cal Kit**”
3. “ **Modify Cal Kit**”>“ **TRL Option**”TRL
4. “ **Impedance**”
5. “ **Reference Plane**”

1. “ **Cal**”
2. “ **Cal Kit**”
3. “ **Modify Cal Kit**”>“ **Modify Cal Kit**”
4. “ **Define STDs**”
5. “ **Media**”

- 1 Ω
-

1. “ **Cal**”
2. “ **Cal Kit**”
3. “ **Modify Cal Kit**”>“ **Export Cal Kit...**”...
4. “ **Save**”

- (.ckx)

D D

1. “ **Cal**”
2. “ **Cal Kit**”
3. “ **Modify Cal Kit**”>“ **Import Cal Kit...**”...
4. “ **Open**”

-
-
-
-

	Open 3G	1 GHz - 3 GHz
	Open 6G	3 GHz - 6 GHz

1. “ **Cal**”
2. “ **Cal Kit**”
3. .
4. “ **Modify Kit**”>“ **Define STDs**”
5. “#1”“ **Label**”“ **Open 3G**”
6. “ **Open**”“ **STD Type**”
7.
8. “ **Min Frequency**”“ **1G**”
9. “ **Max Frequency**”“ **3G**”
10. “ **Return**”
11. “#2”“ **Label**”“ **Open 6G**”
12. “ **Open**”“ **STD Type**”
13.
14. “ **Min Frequency**”“ **3G**”
15. “ **Max Frequency**”“ **6G**”
16. “ **Return**”

#1 #2

1. “ **Cal**”
2. “ **Cal Kit**”
3. .
4. “ **Modify Kit**”>“ **Specify CLSs**”
5. “ **Sub Class**”>“ **Sub Class 1**” 1
6. “ **Open**”“ **Set All**”
7. “#1”(**Open 3G**
8. “ **Return**”
9. “ **Sub Class**”“ **Sub Class 2**” 2
10. “ **Open**”“ **Set All**”
11. “#2”(**Open 6G**
12. “ **Return**”

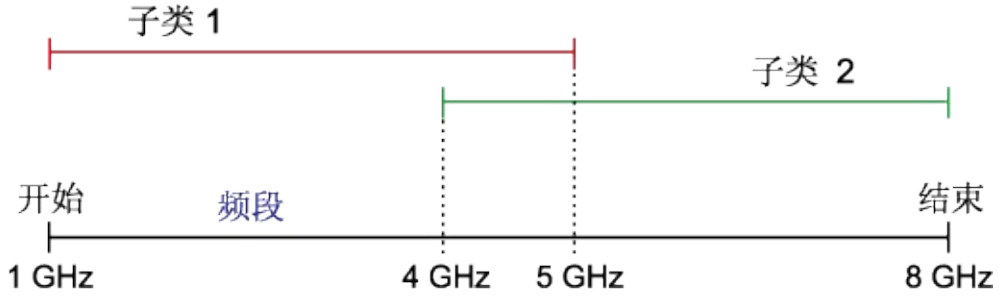
1 2

1. “ **Cal**”
2. “ **Cal Kit**”
3. .
4. “ **Modify Kit**”>“ **Specify CLSs**”
5. “ **Sub Class**”>“ **Sub Class 2**” 2
6. “ **Open**”“ **Set All**”
7. “ **None**”
8. “ **Return**”

1	
2	
3	
4	

1

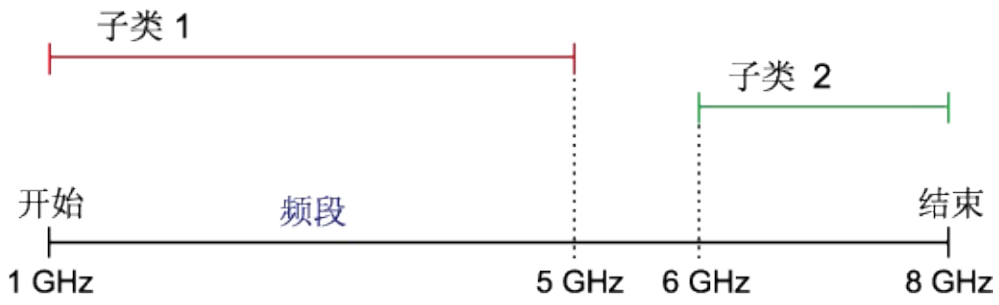
1 GHz 8 GHz 11 GHz 5 GHz 24 GHz 8 GHz
4 GHz 5 GHz



e5071c470

2

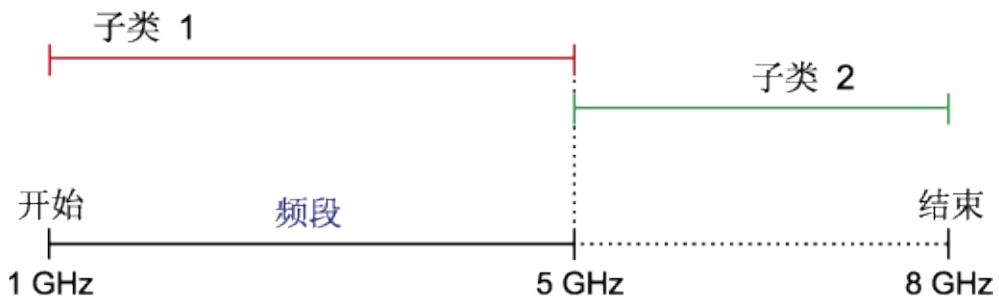
1 GHz 8 GHz 11 GHz 5 GHz 26 GHz 8 GHz
5 GHz 6 GHz



e5071c468

3

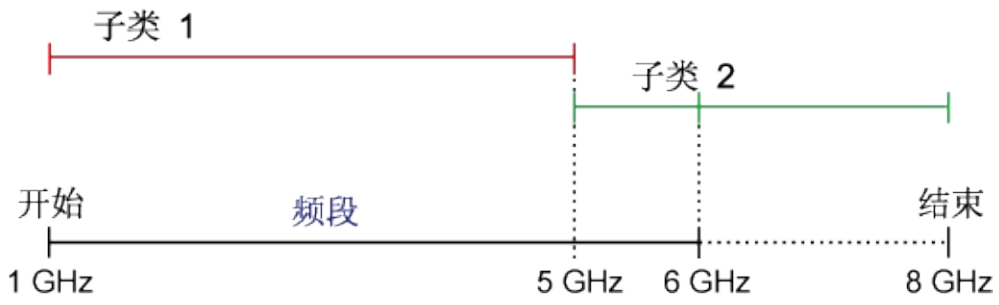
1 GHz 5 GHz 11 GHz 5 GHz 25 GHz 8 GHz 2
1



e5071c467

4

1 GHz 6 GHz 11 GHz 5 GHz 25 GHz 8 GHz
26 GHz 8 GHz5 GHz 6 GHz



e5071c469

-
-
-
-

E5071C

/

//

RC	
RC	
RC?	
RC?	
RC!	
RC!	

/

/

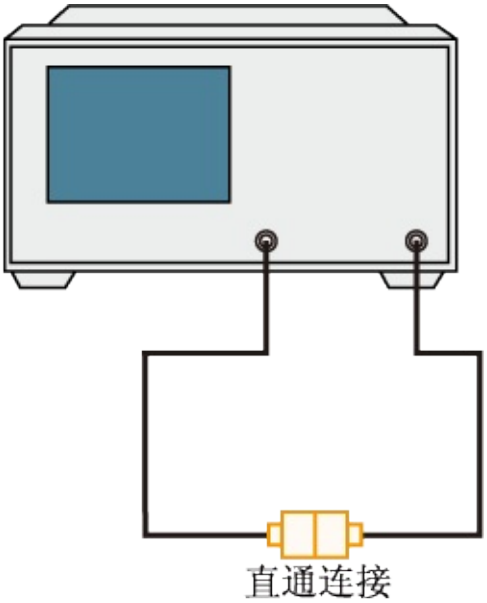
1. “ **Cal**”
2. “ **Receiver Calibration**”
3. “”
4. “ **Correction**”

/

-
-

1. “ **Cal**”
2. “ **Receiver Calibration**”
3. “ **Channel Next**”/“ **Channel Prev**”
4. “ **Select Port**”
5.

1. “ **Cal**”
2. “ **Receiver Calibration**”
3.
4. “ **Source Port**”
5.
6.



e5071c321

7. “ **Take Cal Sweep**”

-
-
-
-
-
-

E5071C

/

///

PC	
PC	
PC?	
PC?	

/

1. “ **Cal**”
2. “ **Power Calibration**”
3.
4. “ **Correction**”/

	Agilent 437B
	Agilent 438A
	Agilent E4416A
	Agilent E4417A
	Agilent E4418A
	Agilent E4418B
	Agilent E4419A
	Agilent E4419B
	HP EPM-441A
	HP EPM-442A
	Agilent 8482A
	Agilent E4412A
	HP ECP-E18A

E5071C GPIB

E5071C USB/GPIB E5071C USB GPIB
E5071C GPIB

USB/GPIB GPIB

GPIB

GPIB

1. “ **System**”
2. “ **Misc Setup**”>“ **GPIB Setup**”GPIB >“ **Power Meter Address**”
3. GPIB

E5071C 100%

437B 438A E5071C

437B 438A

8482A	E5071C E5071C
E4412A ECP-E18A	E5071C E5071C

Ax.02.00 E4418AE4419AEPM-441A EPM-442A E5071C E5071C E5071C

1. “ **Cal**”
2. “ **Power Calibration**”
3. A “ **Sensor A Settings**” A B “ **Sensor B Settings**” B
4. “ **Ref Cal Factor**”
5. 50 MHz

6. “ **Frequency**”“ **Factor**”

- “ **Preset**”>“ **OK**”
- 2
- “ENTRY”“ **Focus**”?

CSV

1. “ **Cal**”
2. “ **Power Calibration**”
3. “ **Sensor A Settings**” A “ **Sensor B Settings**” B
4. “ **Export to CSV File**” CSV “Save As”
   CSV *.csv
5. “ **File Name**”“ **Save**”

“” CSV

1. “ **Cal**”
2. “ **Power Calibration**”
3. “ **Sensor A Settings**” A “ **Sensor B Settings**” B
4. “ **Import from CSV File**” CSV “Open”
   CSV *.csv
5. CSV “ **Open**”

- 1) / CSV
- 2) CSV

//

- /
-
-
-
-

/

1. “ **Cal**”
2. “ **Power Calibration**”
3. “ **Channel Next**”/“ **Channel Prev**”
4. “ **Select Port**”
5.

DUT DUT

/

/

1. “ **Cal**”
2. “ **Power Calibration**”
3. .
4. “ **Loss Compen**”>“ **Compensation**”/

1. “ **Cal**”
2. “ **Power Calibration**”
3. .
4. “ **Loss Compen**”
5. “ **Frequency**”“ **Loss**”

- “ **Preset**”>“ **OK**”
- 2
- “ENTRY”“ **Focus**”?

CSV

1. “ **Cal**”

2. “ **Power Calibration**”
3. .
4. “ **Loss Compen**”
5. “ **Export to CSV File**” CSV “Save As”
   CSV *.csv
6. “ **File Name**”“ **Save**”

CSV

//

1. “ **Cal**”
2. “ **Power Calibration**”
3. .
4. “ **Loss Compen**”
5. “ **Import from CSV File**” CSV “Open”
   CSV *.csv
6. CSV “ **Open**”

1) / CSV 2) . CSV

1. “ **Cal**”
2. “ **Power Calibration**”>“ **Tolerance**”

3.

E5071C 100%

1. “ **Cal**”
2. “ **Power Calibration**”
3.
4. “ **Use Sensor**” A B A
5. “ **Num of Readings**”
6.
7.
8. “ **Take Cal Sweep**”
9. “ **Abort**”
10.

GPIB

-
-

ErEsEd Et Ex

- E5071C
- 3 1 4

2

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”>“ **2-Port Cal**”2 >“ **Select Ports**”
4. 2 x y
5. “ **Transmission**”
6. x y DUT
7. “ **Port x-y Thru**” x-y **x** **y**
8. “ **Return**”
9. “ **Overwrite**” 2

.

-
- 3/4
- 3/4 TRL

3/4 3/4 TRL TRL

3/4

3/4 3/4 3/4

-
  -
  -
  -
  - N
-
  -
  -
  - IF
  -
  - /
  -

3

12 3	2-3
12 4	2-4
13 4	1-4
23 4	2-4

3 “(Optional)” 2-3 “ **2-3 (Optional)**”2-3

4

1-21-33-4	1-42-32-4

4 “(Recommended)” 2-3 “ **2-3**
(Recommended)”2-3“ **2-3 (Optional)**”2-3

4

3/4 TRL

3/4 TRL // 3/4 TRL 3/4 3/4 TRL

////

3 TRL

12 3	2-3
12 4	2-4
13 4	1-4
23 4	2-4

3 TRL /“(Optional)”/ 2-3 “ **2-3 (Optional)**”2-3/“ **2-3 Thru/Line**”2-3 /

//“(Optional)”/ 2-3 “ **2-3 (Optional)**”2-3 /“ **2-3 Line/Match**”2-3 /

/Fwd Rvs

4 TRL

1-21-33-4	1-42-32-4

4 TRL /“(Recommended)”/ 2-3 “ **2-3 (Recommended)**”2-3/“ **2-3 Thru/Line**”2-3 /“ **2-3 (Optional)**”2-3

//“(Recommended)”/ 2-3 “ **2-3 (Recommended)**”2-3/“ **2-3 Line/Match**”2-3 /“ **2-3 (Optional)**”2-3

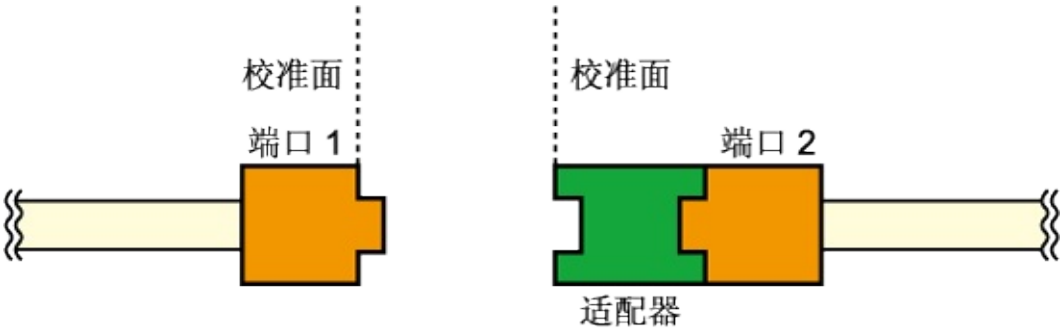
/Fwd Rvs

4 TRL //

- .
- .
- /

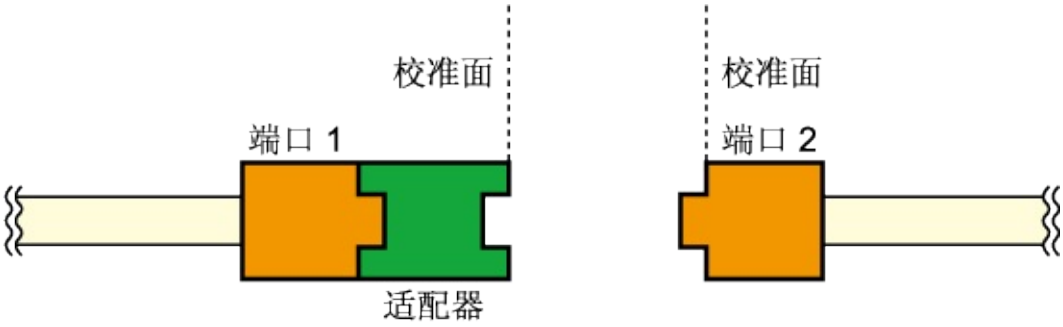
2

2

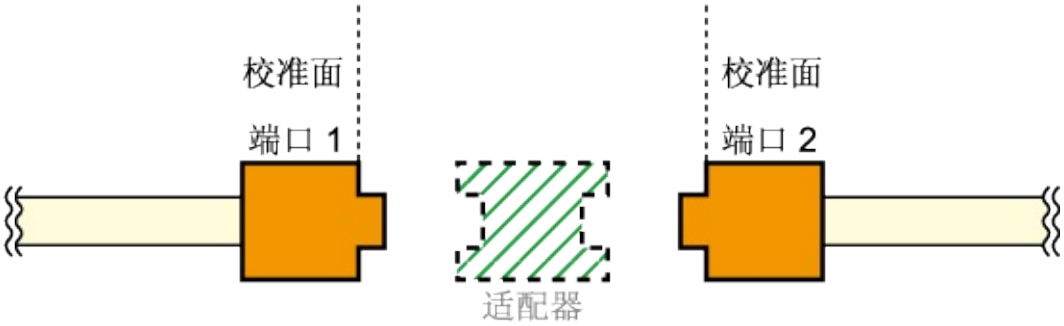


e5071c200

1



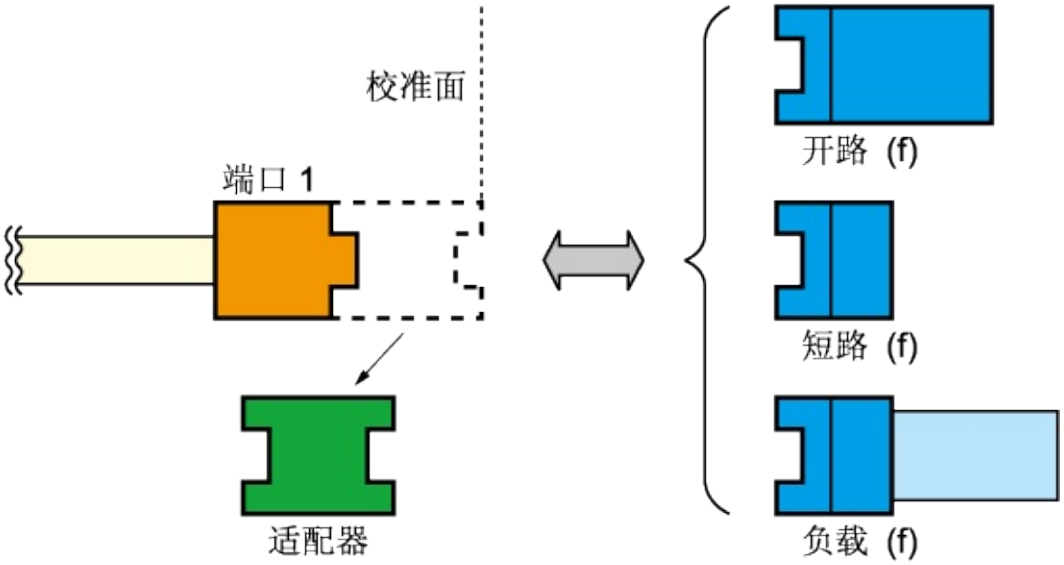
e5071c201



e5071c202

2 E5071C

1.
2.
3.



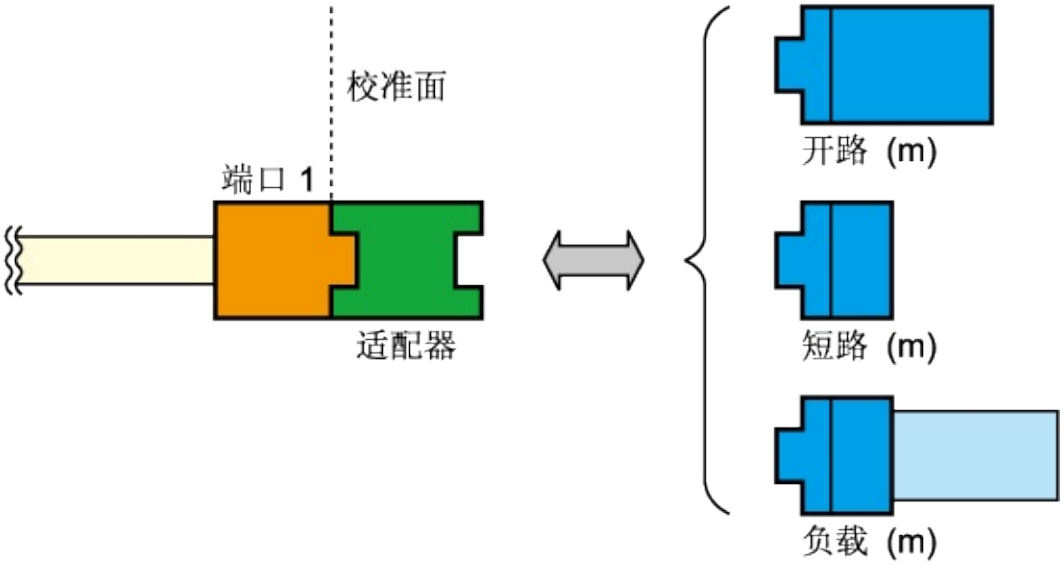
e5071c203

n E5071C

1.

2.

3.



e5071c204

1. S21 S12
2. ±1/4

/

S

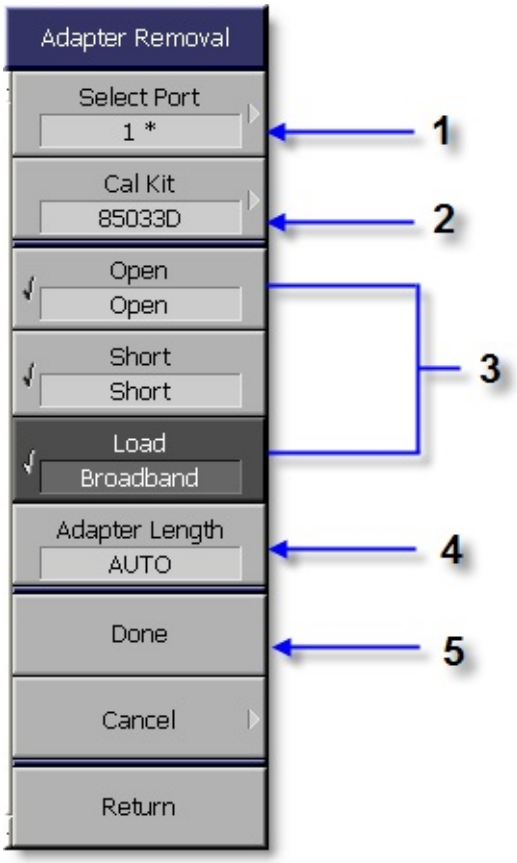
1.
2. “ **CalKit**”
3.
4.

S

(**+**) (-)

/

1. n
2. “Adapter Removal”
   “Cal”>“ **Calibrate**”>“ **Adapter Removal**”



e5071c194

3. “ **Select Port**” 1 **1** *
4. “ **CalKit**” 85033D **2**
5. “ **Adapter Length**”“ **Auto**”/ **4** (**+**) (-)
6. “ **Calkit**” 85033D/“ **Open**”“ **Short**” “ **Load**”

“ **Open**”“ **Short**”“ **Load**”

6. “ **Done**” **5**

-
-

ENA

-
-
-
- DUT DUT THRU

1. S21 S12
2. ±1/4

TRL

S

1.

2. “ **CalKit**”

1. “ **Cal**”“ **Calkit**”
2. 85033D
3. “ **Cal**”>“ **Modify Calkit**”>“ **Define STDs**”>“ **No Name**”>“ **Label**”
4. {Unknown Thru1}

注意 Unknown Thru1“”

5. “ **Cal**”>“ **Modify Calkit**”>“ **Define STDs**” >“**{Defined Name}**”{}>“ **STD Type**”>“ **Unknown Thru**”
6. “ **Offset Delay**”
7. “ **Cal**”>“ **Modify Calkit**”>“ **Specify CLSs**” >“**Thru**”>“ **Set All**”>“ **{Defined Name}**”{}

注意

8. “ **Cal**”>“ **Calibrate**”>“ **n-Port Cal**”n (n>1) >“**Transmission**”>“ **{Defined Name}**”{}
9. n

-
-
-
- ECal

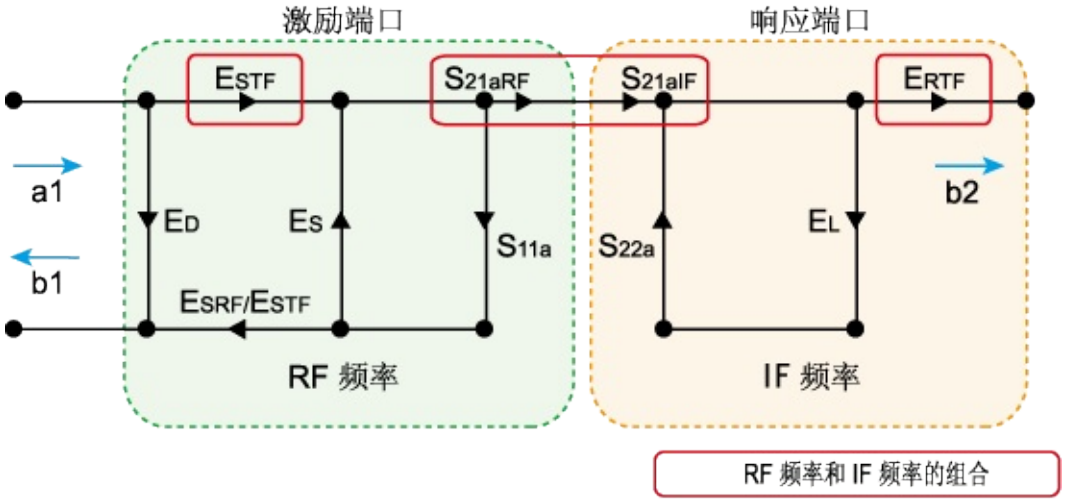
E5071C

//

2 2

-
-
-

(EXF) (ETF) EXF ETF (ETF)



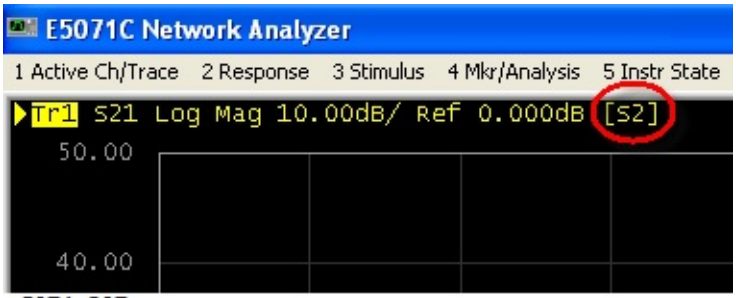
ED 方向性误差
ES 源匹配误差
EL 负载匹配误差
ETF 传输跟踪误差 = (ESTF x ERTF)
ESTF 传输跟踪源误差
ERTF 传输跟踪接收机误差

ERF 反射跟踪误差 = (ESTF x ESRF)
ESRF 反射跟踪源误差
C21a 实际转换损耗 = (S21aRFF x S21aIF)
S21aRF RF 频率上的 S21aRF 转换
S21aIF IF 频率上的 S21aIF 转换

e5071c457

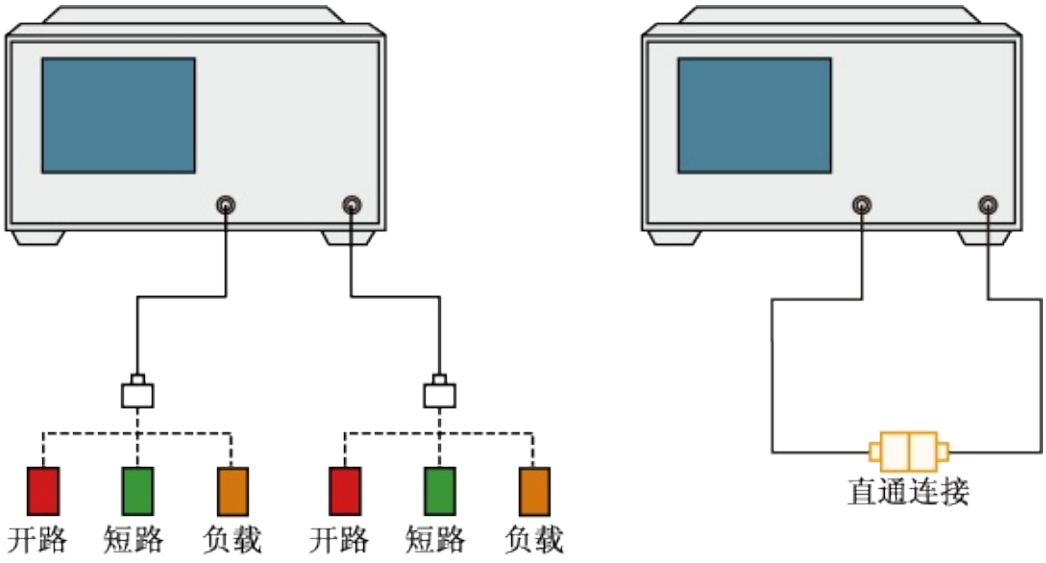
(ETF) (ESTF) (ERTF) (ESTF) 2 (ETF)
(ESTF) (ERTF) (ETF) ESTF ERTF

[S2]



e5071c207

1. “ **Sweep Setup**” >“ **Frequency Offset**”
2. “ **Channel Next**”/“ **Channel Prev**”
3. “ **Cal**”
4. “ **Mixer/Converter Calibration**”/>“ **Scalar Cal (Manual)**”>“ **Select Ports**”
5. .



e5071c330

2 “” 2

x y

6. “ **Reflection**”
7. **x**DUT

8. “ **Port x @Freq x Open**” x x
9. “ **Port x @Freq y Open**” x y
10. “”“”
11. “ **Port x@Freq x Short**” x x
12. “ **Port x@Freq y Short**” x y
13. “”“”
14. “ **Port x@Freq x Load**” x x
15. “ **Port x@Freq y Load**” x y
16. y
17. “ **Return**”

18. “ **Transmission**”
19. x y DUT
20. “ **Port y-x@Freq y Thru**” y-x y
21. “ **Port x-y@Freq x Thru**” x-y x
22. “ **Return**”

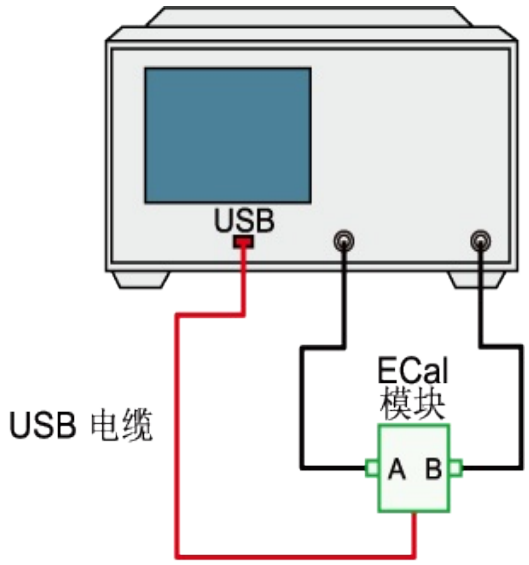
23. “ **Power Meter**”
24. “ **Use Sensor**” A B A
25.
26. “ **Port x@Freq x**” x x
27. “ **Port x@Freq y**” x y
28. “ **Port y@Freq x**” y x
29. “ **Port y@Freq y**” y y

30. “ **Return**”

31. “ **Done**”

ECal

1. “ **Sweep Setup**” >“ **Frequency Offset**”
2. “ **Channel Next**”/“ **Channel Prev**”
3. “ **Cal**”
4. “ **Mixer/Converter Calibration**”/>“ **Scalar Cal (Manual)**”>“ **Select Ports**”
5.
6. “ **Power Meter**”
7. “ **Use Sensor**” A B A
8.
9. “ **Port x@Freq x**” x x
10. “ **Port x@Freq y**” x y
11. “ **Port y@Freq x**” y x
12. “ **Port y@Freq y**” y y
13. “ **Return**”
14. ECal



e5071c322

15. “ **ECal & Done**”ECal

ECal E5071C“ **ECal & Done**”ECal
ECal

-
- IF
-
- IF
-

E5071C

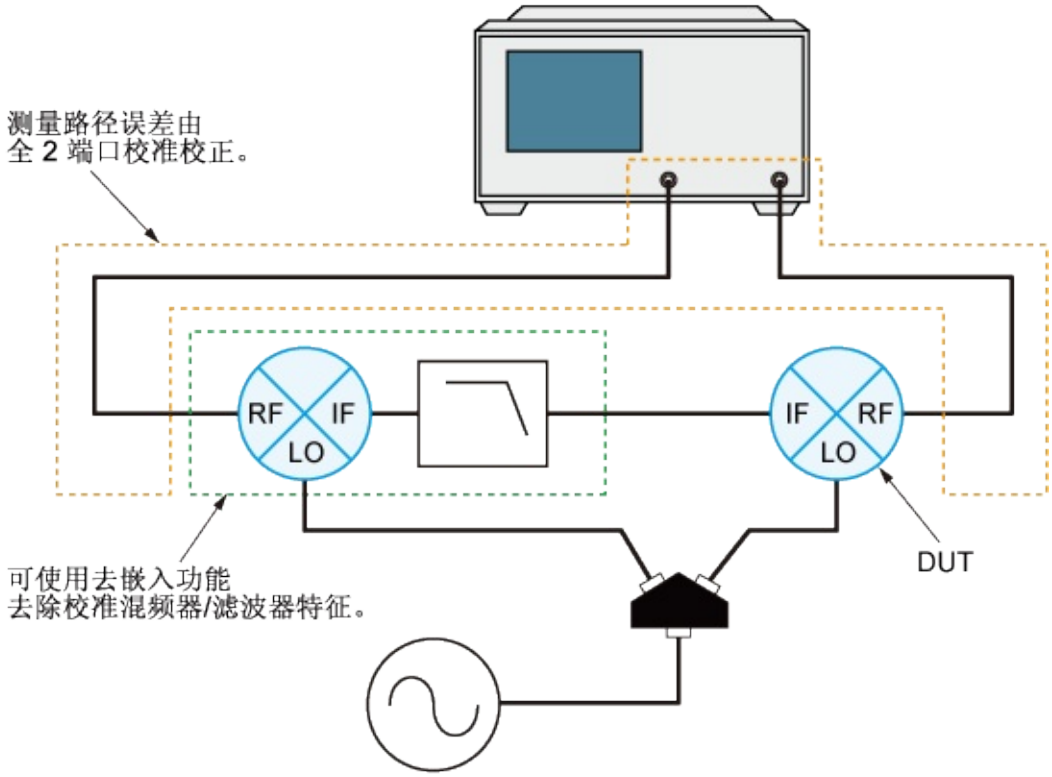
// IF E5071C

IF

IF IF RF

2 IF / 2 IF (DUT)

/



e5071c340

IF

(DUT)

IF

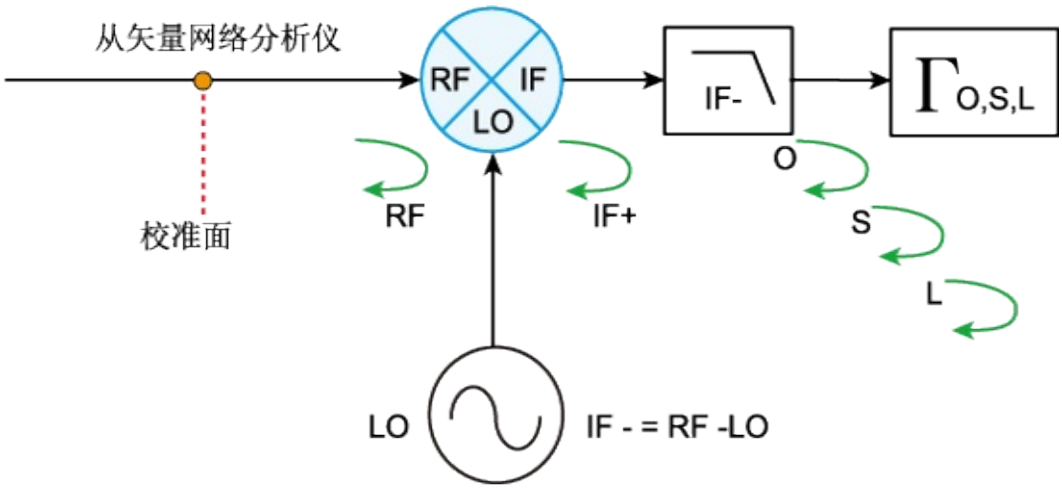
/ IF RF+LORF-LO LO-RF IF

IF

IF IF IF RF IF IF
(IF+) IF IF (IF-)

1 Γ OΓS Γ L

IF



e5071c458

- “” RF IF IF RF
- IF 10 dB 15 dB

IF

E5071C VBA IF

	VBA
D:\Agilent	**MixerCharacterization.vba**

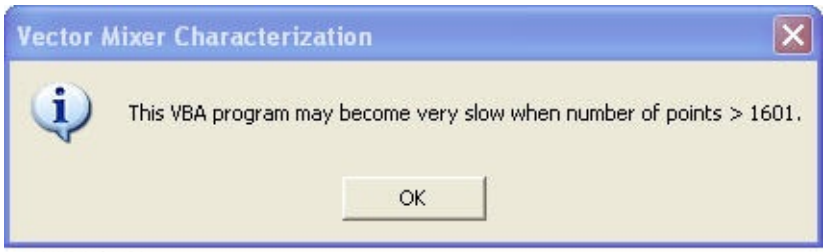
1.

2. VBA

IF 1 2 2 2 2

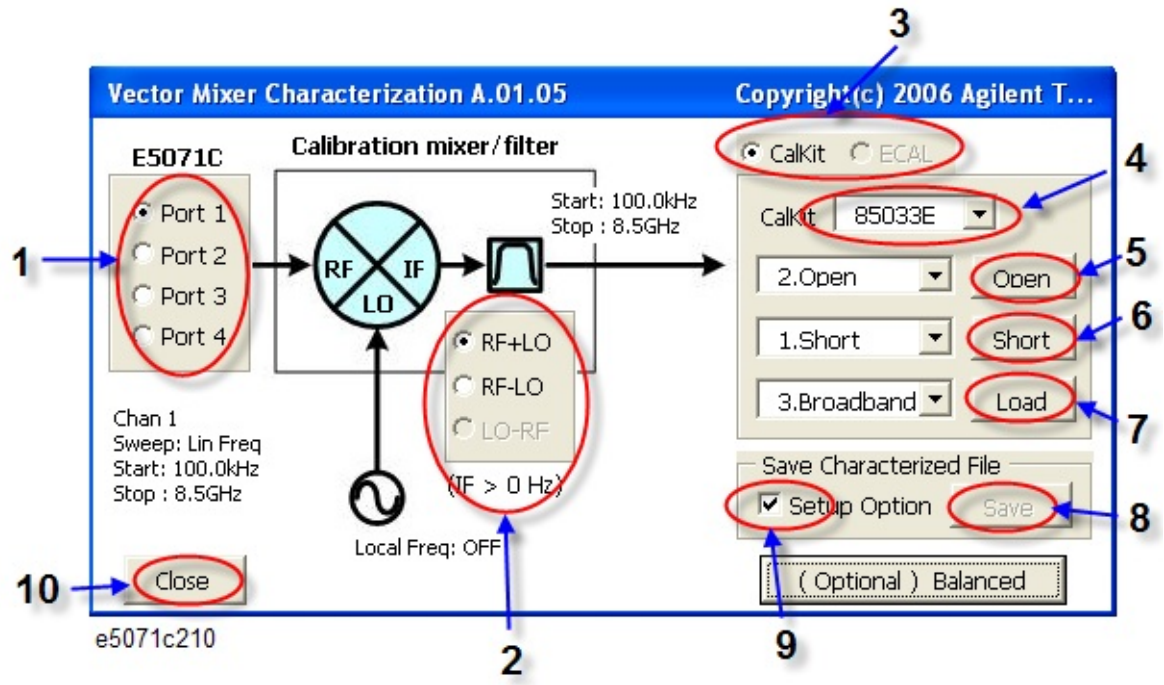
VBA

1. “ **Macro Setup**”
2. “ **Load & Project**”
3. “Open” **D:\Agilent\MixerCharacterization.vba**
   “ **Open**”
4. “ **Macro Run**”
5. “ **OK**”



e5071c136

“1 4 ” 1601 E5071C VBA



3.

1 1

■ 2

4. IF

IF RF+LORF-LO LO-RF IF 2

■ E5071C IF 0 kHz
IF BW IF

5.

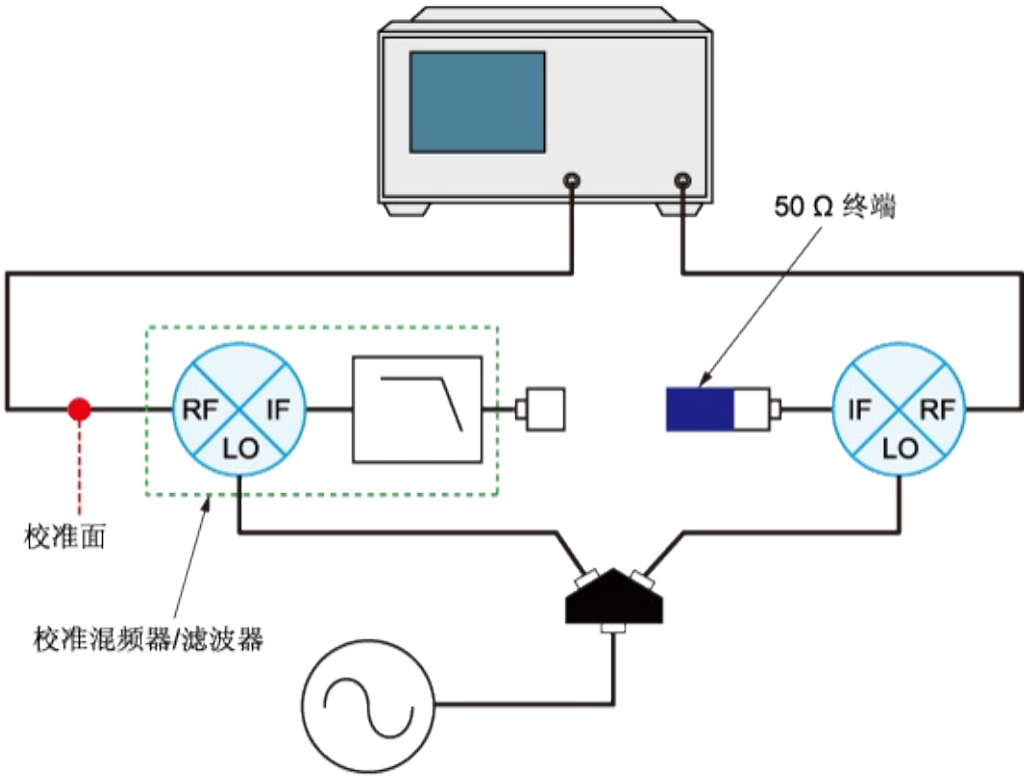
3

■ E5071C Ecal E5071C ECal

6. IF

1

IF



e5071c341

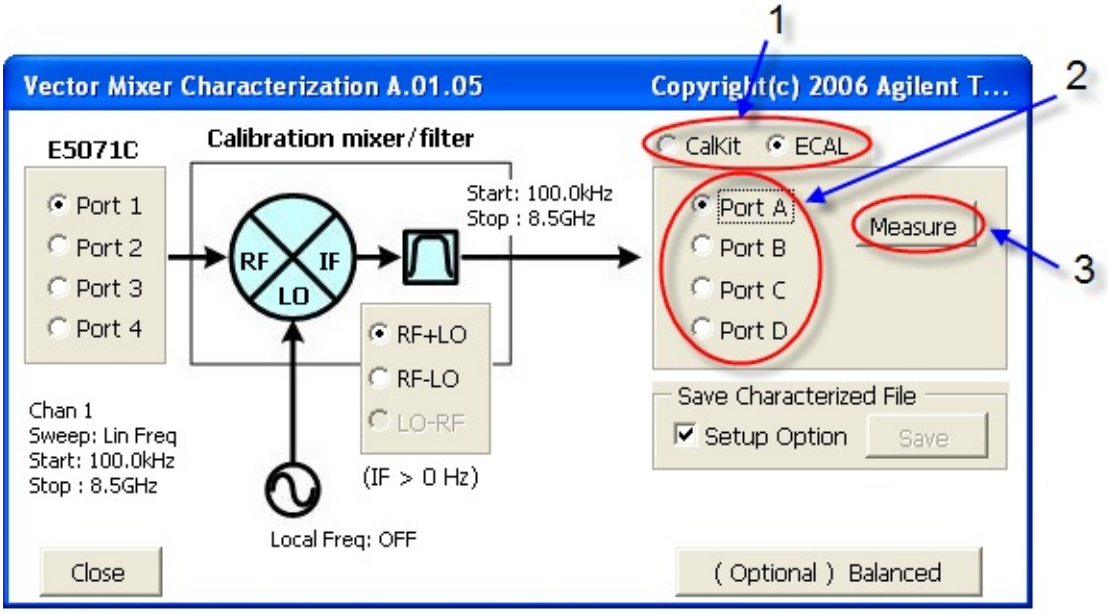
2

LO IF /LO LO LO
LO

1. “ **CalKit**” __
2. “ **CalKit**” __
3. “ **Open**” __
4. “ **Short**” __
5. “ **Load**” __

7. IF ECal

ECal



e5071c211

1. “ **ECal**” __
2. Ecal __
3. “ **Measure**” __

8. VBA

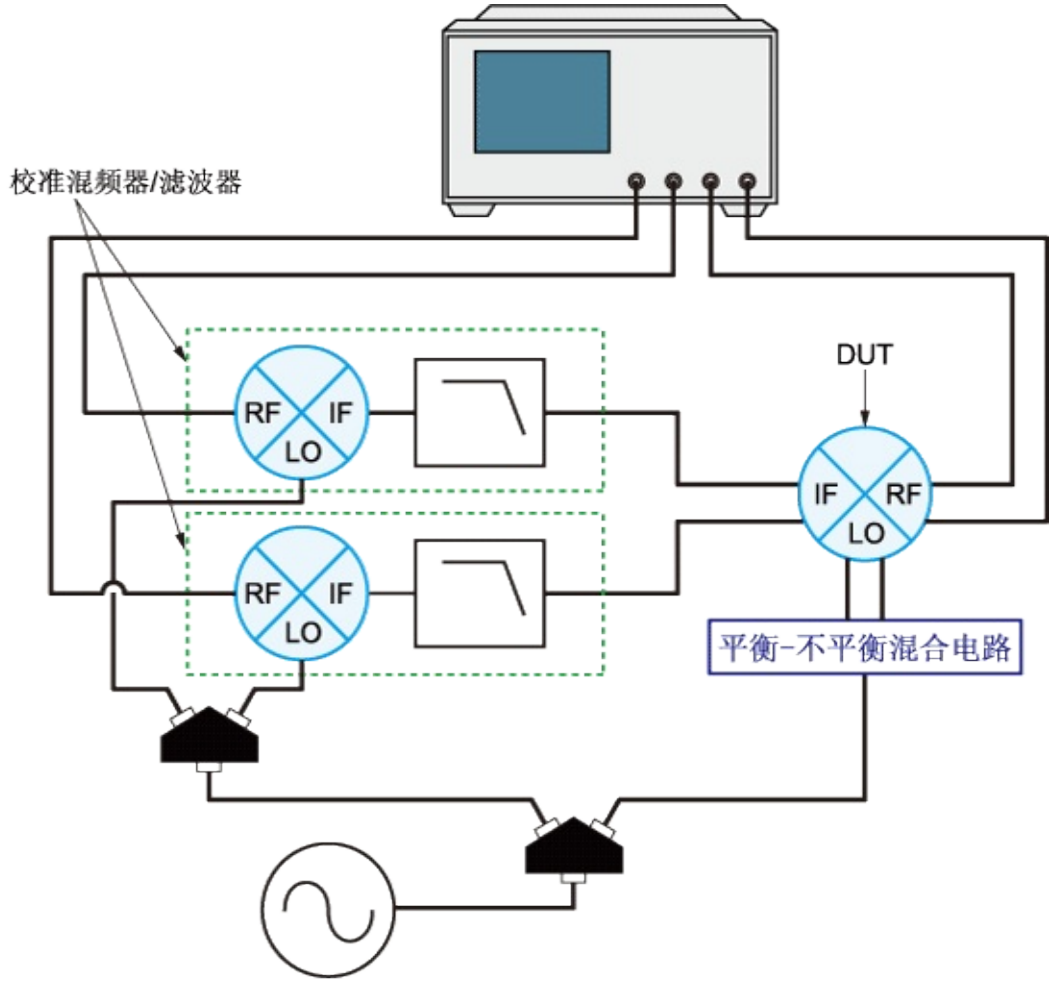
1. “Save” __“Save”
2. “Save” IF Touchstone “Setup” ____ ____
3. “ **Close**” 10 __

IF

E5071C VBA IF IF IF

IF IF IF LO IF
IF

IF



e5071c342

1. IF

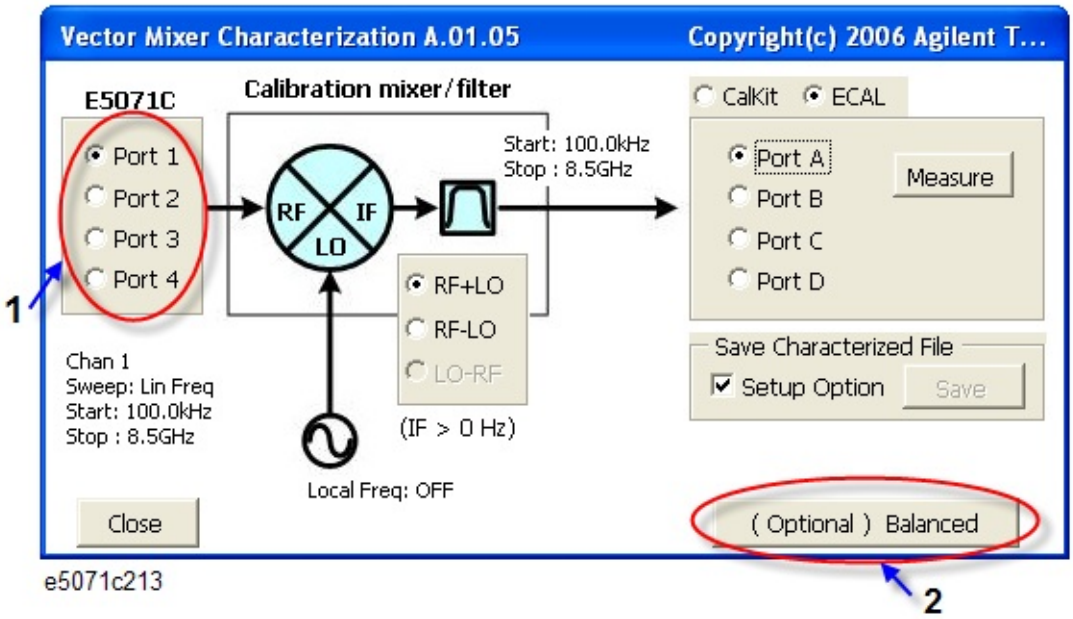
IF

IF 4 4 4

1. “ **Macro Setup**”
2. “ **Load & Project**”
3. “Open” **D:\Agilent\MixerCharacterization.vba**
   “ **Open**”
4. “ **Macro Run**”
5. “ **Port 1**” 1 1 IF 1 (MIXER_1.s2p)
6. “ **Port 2**” 2 1 IF 2 (MIXER_2.s2p)

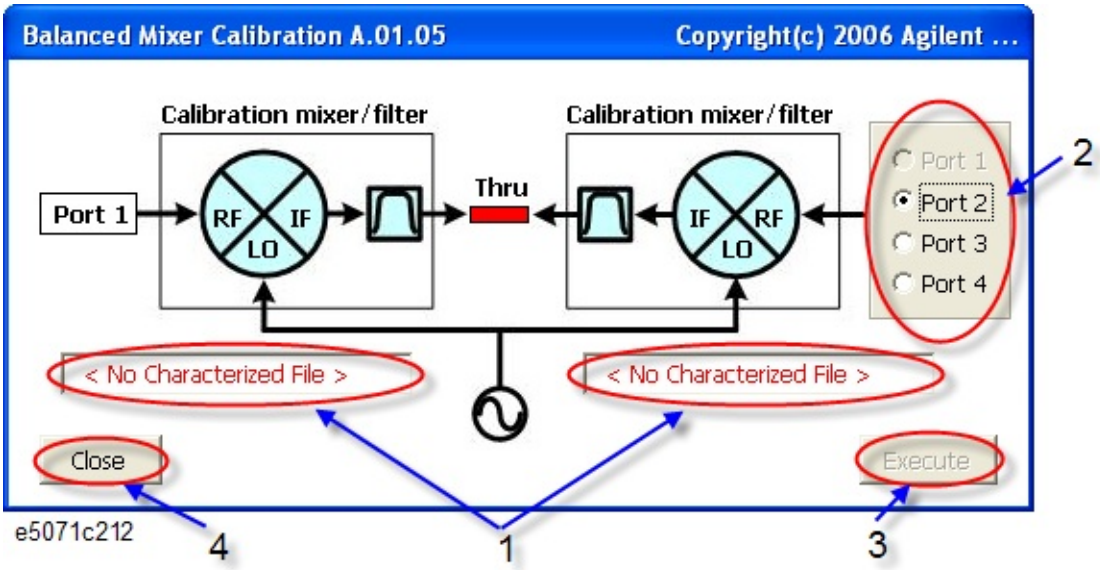
注意 IF

7. “ **(Optional) Balanced Mixer**” 2



8. IF (MIXER_1.s2pMIXER_2.s2p) 1

IF



1. 2 IF IF LO
2. " **Execute**" 3

IF (*.s2p)" **Execute**"

LO

3. “ **Close**” 4

2 TRL

-
-

TRL

2 TRL 2 2 12

TRL

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”>“ **2-Port TRL Cal**”2 TRL
4. “ **Select Ports**” TRL 2 x y
5.

/

1. x y DUT
2. “ **Thru/Line**”/
3. “ **Port x-y Thru**” x-y /
4. “ **Return**”

1. “ **Reflect**”
2. “ **Port x Reflect**” x x
3. “ **Port y Reflect**” y y
4. “ **Return**”

/

1. “ **Line/Match**”/
2. “ **softkey**” /
3. “ **Return**”
4. “ **Done**” TRL 2

3 TRL

-
-

TRL

3 TRL 3 3 27

TRL

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”>“ **3-Port TRL Cal**”3 TRL >“ **Select Ports**”
4. TRL 3 . xy z
5. “ **Thru/Line**”/
6. “ **Reflect**”
7. “ **Line/Match**”/

x-y Line/Matchx-y /	x y /
x-y Fwd (Syx) x-y (Syx)	x y /
x-y Rvs (Sxy) x-y (Sxy)	x y /
x-z Line/Matchx-z /	x z /
x-z Fwd (Szx) x-z (Szx)	x z /
x-z Rvs (Sxz)	

x-z (Sxz)	x z /
y-z Line/Matchy-z /	y z /
y-z Fwd (Szy) y-z (Szy)	y z /
y-z Rvs (Syz) y-z (Syz)	y z /

8. “ **Done**” TRL 3

4 TRL

- .
- .

TRL

4 TRL 4 4 48

TRL

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Calibrate**”>“ **4-Port TRL Cal**”4 TRL >“ **Select Ports**”
4. “ **Thru/Line**”/
5. “ **Reflect**”

Port 1 Reflect 1	1
Port 2 Reflect 2	2
Port 3 Reflect 3	3
Port 4 Reflect 4	4

6. “ **Line/Match**”/”

1-2 Line/Match1-2 /	1 2 /
1-2 Fwd (S21) 1-2 (S21)	1 2 /
1-2 Rvs (S12)	

1-2 Rvs (S11) **1-2** **(S12)**	1 2 /
1-3 Line/Match1-3 /	1 3 /
1-3 Fwd (S31) **1-3** **(S31)**	1 3 /
1-3 Rvs (S13) **1-3** **(S13)**	1 3 /
1-4 Line/Match1-4 /	1 4 /
1-4 Fwd (S41) **1-4** **(S41)**	1 4 /
1-4 Rvs (S14) **1-4** **(S14)**	1 4 /
2-3 Line/Match2-3 /	2 3 /
2-3 Fwd (S32) **2-3** **(S32)**	2 3 /
2-3 Rvs (S23) **2-3** **(S23)**	2 3 /

2-4 Line/Match2-4 /	2 4 /
2-4 Fwd (S42) 2-4 (S42)	2 4 /
2-4 Rvs (S24) 2-4 (S24)	2 4 /
3-4 Line/Match3-4 /	3 4 /
3-4 Fwd (S43) 3-4 (S43)	3 4 /
3-4 Rvs (S34) 3-4 (S34)	3 4 /

7. “ **Done**” TRL 4

-
-
-
-
-

-
-
-
-
-

E5071C 1 36 “”

“”“”

1	Port 1 1	S11S21S31S41
2	Port 2 2	S12S22S32S42
3	Port 3 3	S13S23S33S43
4	Port 4 4	S14S24S34S44

23 4

“ **Internal**”	
“ **External**”	(BNC) I/O 18
“ **Manual**”	“ **Trigger**”>“ **Trigger**”
“ **Bus**”	GPIB/LAN/USB

“ **Hold**”	“”
“ **Single**”	“”“”
“ **Continuous**”	“”“”

1.

1. “ **Trigger**”
2. “ **Trigger Source**”
3. .
4. “ **External**”“ **Ext Trig Input**”

Negative Edge	
Positive Edge	

I/O

2.

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Trigger**”
3.

Hold	
Single	
Continuous	

Hold All Channels	
Continuous Disp Channels	“ **Display**”>“ **Allocate Channels**”

3.

“ **Trigger**”>“ **Restart**”

-
-

On Point	
On Sweep	

1. “ **Trigger**”
2. “ **Trigger Event**”
3.

On Point	
On Sweep	

-
-
-

E5071C

“External”

ON	“ON”
OFF	“OFF”

“ON”

1.

“External”“On”

1. “ **Trigger**”
2. “ **Low Latency**”
3. “ **ON**”

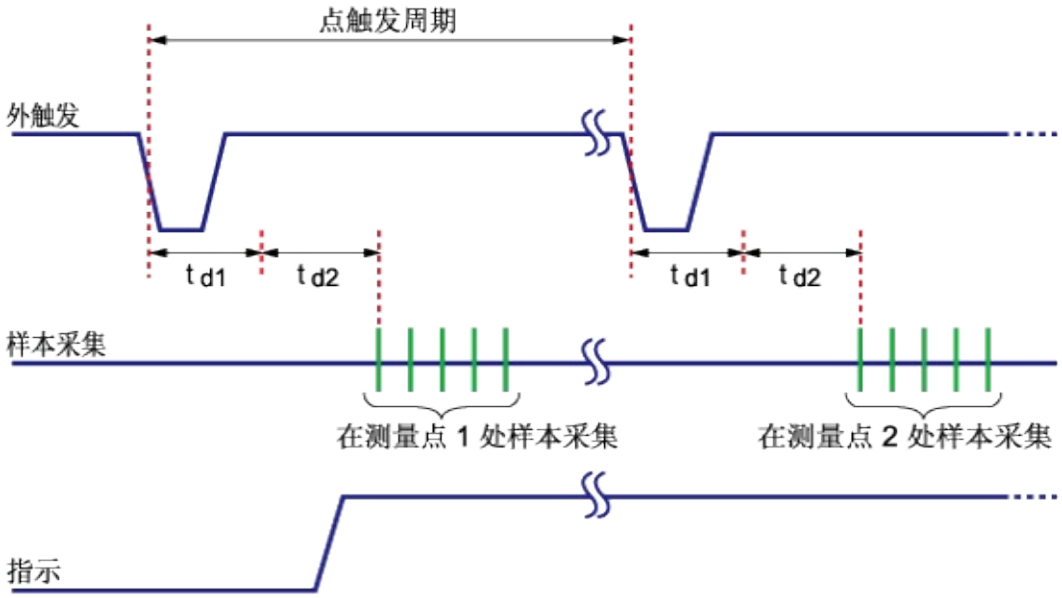
2.

10 μs

1. “ **Trigger**”
2. “ **Trig Delay**”
3.

IFBW

= = =



e5071c345

I	
External Trig	
Sampling	E5071C
Index	/ I/O “ON”“ON”
Point Trigger	

Period	E5071C E5071C
td1	
td2	E5071C IF td2 = 13±1 μs

-
-

“ **ON**”

ON	
OFF	

“× ”

1.

“ **ON**”

1. “ **Avg**”
2. “ **Avg Trigger**”
3. “ **ON**”

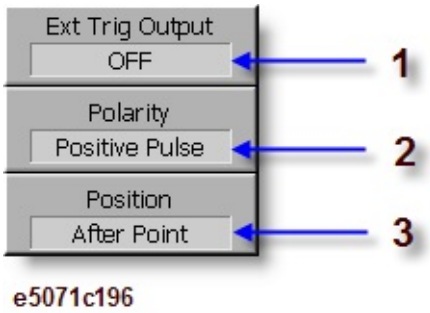
2.

1. “ **Trigger**”
2. “ **Single**”

-
-

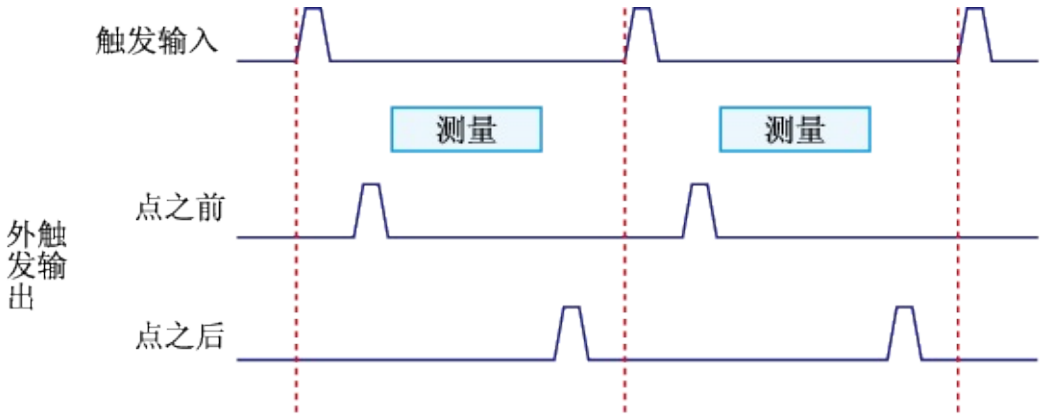
E5071C

1. “ **Trigger**” “ **Ext Trig Output**”(**1**) “ **ON**”



2. “ **Polarity**”“ **Positive Pulse**”“ **Negative Pulse**” (**2**)

3. “ **Position**”“ **After Point**”“ **Before Point**” (**3**)



e5071c208

- .
- .
- /
- _____
- -_____
- .
- .
- .
- .

- .
- .
- .
- /
- /
- .
- .
- .
- .

-
-
-
-

E5071C 10 X Y

Y

Smith - **Lin / Phase** - /		
Smith - **Log / Phase** - /		
Smith - **Real / Imag** - /		
Smith - **R + jX** - R + jX		
Smith - **G + jX** - G + jX		
Polar - **Lin / Phase** - /		
Polar - **Log / Phase** - /		
Polar - **Real / Imag** - /		

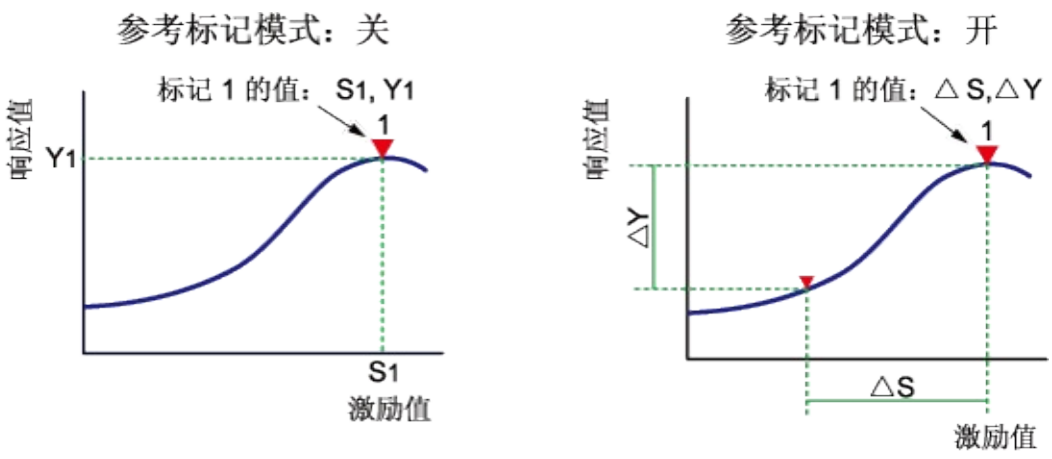
1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Trace Next**”/“ **Trace Prev**”

3. “ **Marker**” 1 1
4.

1.
2.

1. “ **Marker**”
2. “ **Clear Marker Menu**”

/



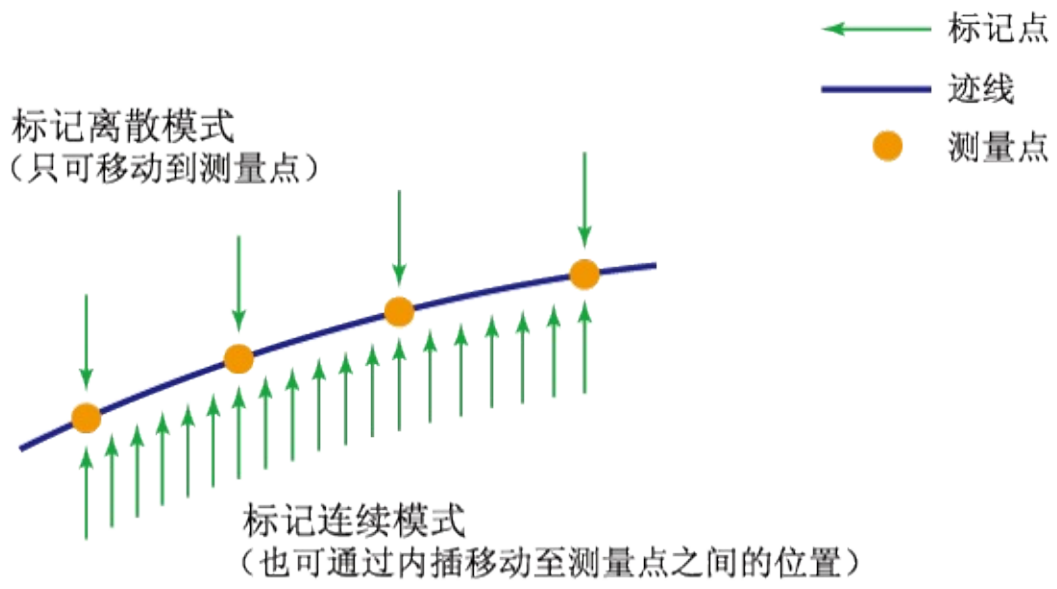
e5071c310

1. .
2.
3. “ **Ref Marker Mode**”
4.
5.

“ **Marker**”“ **Ref Marker**”

/

“ **Discrete ON**”	
“ **Discrete OFF**”	

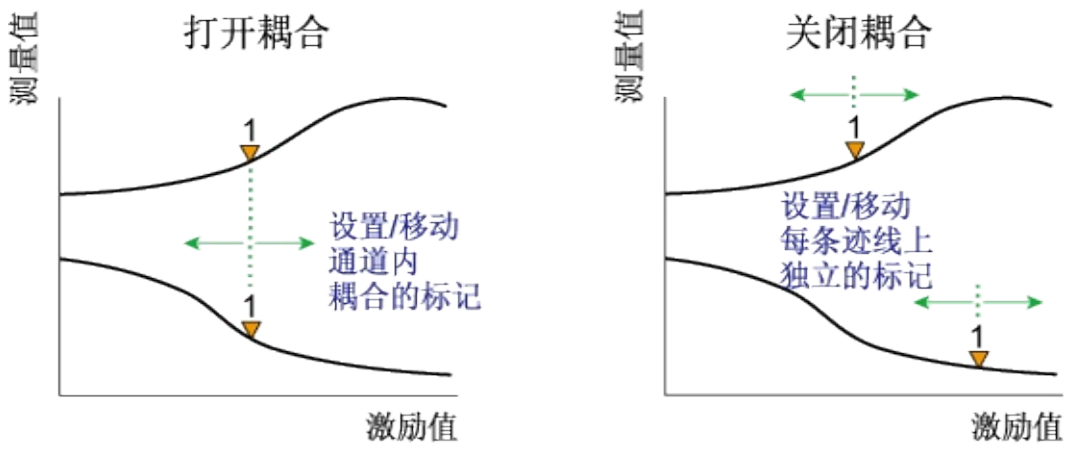


e5071c311

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”

2. “ **Marker Fctn**”
3. “ **Discrete**”

/



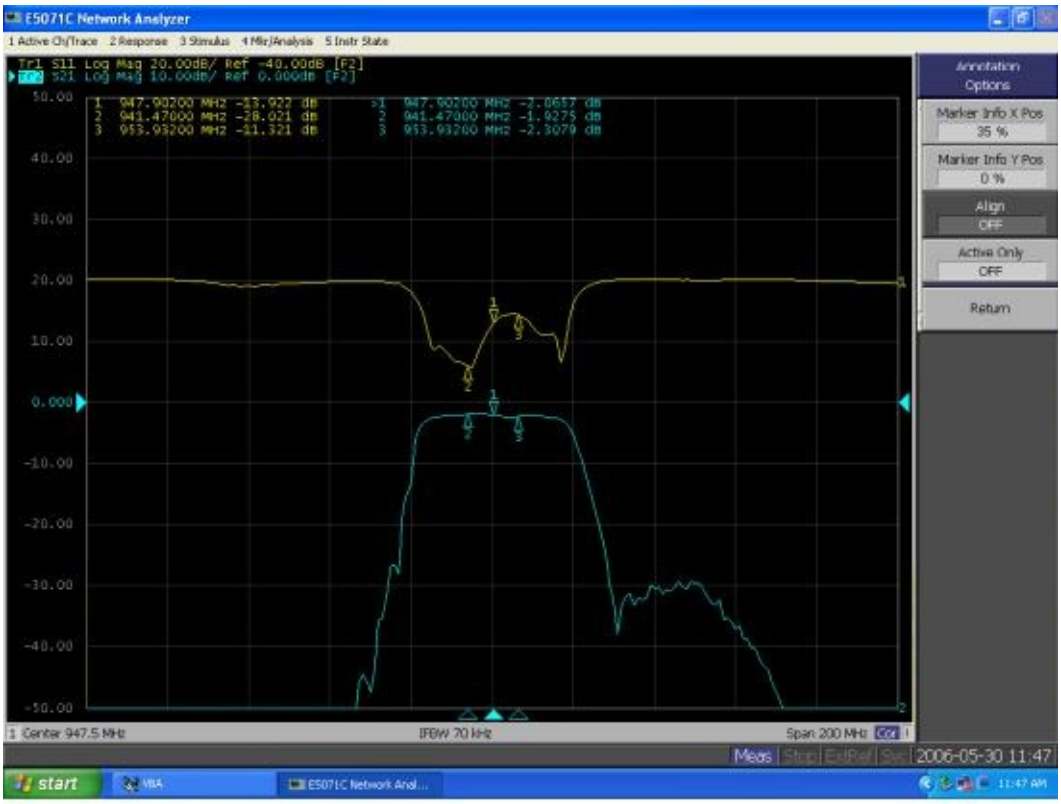
e5071c388

“ **Coupling ON**”	
“ **Coupling OFF**”	

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Marker Fctn**”
3. “ **Couple**”

1. “ **Marker Fctn**”
2. “ **Marker Table**”

e5071c063

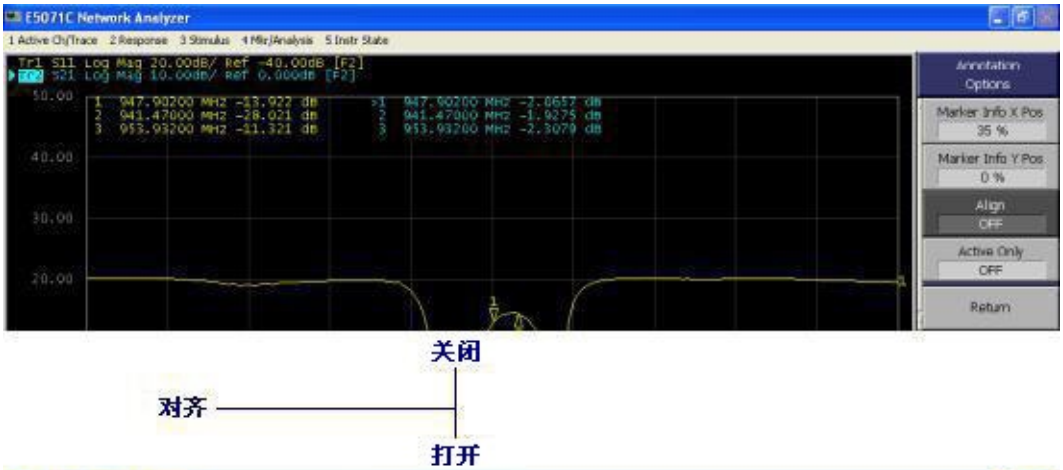


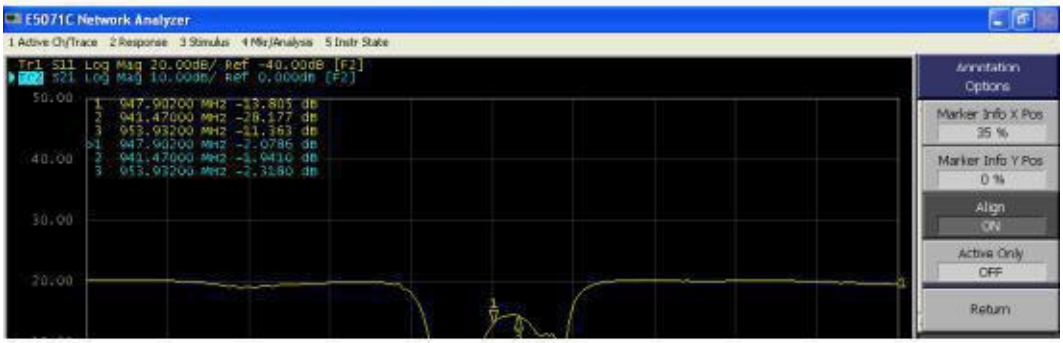
e5071c064

X	
Y	

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Marker Fctn**”

3. “ **Annotation Options**”
4. “ **Marker Info X Pos**”X
5. “ **Marker Info Y Pos**”Y

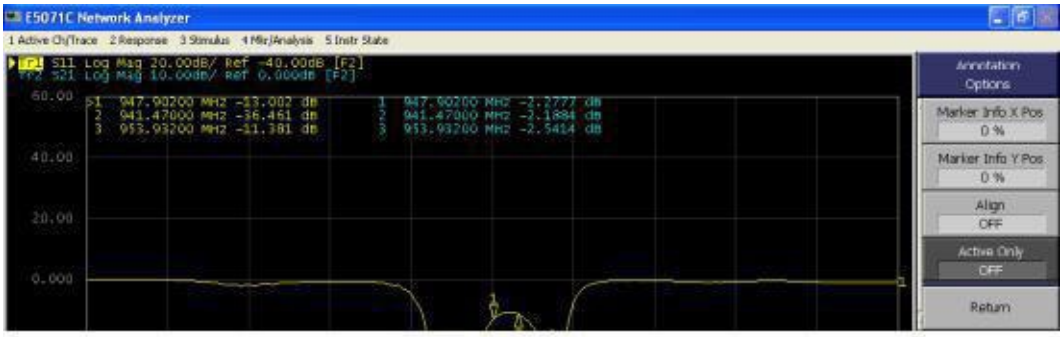




e5071c065

“ **Align ON**”	1
“ **Align OFF**”	

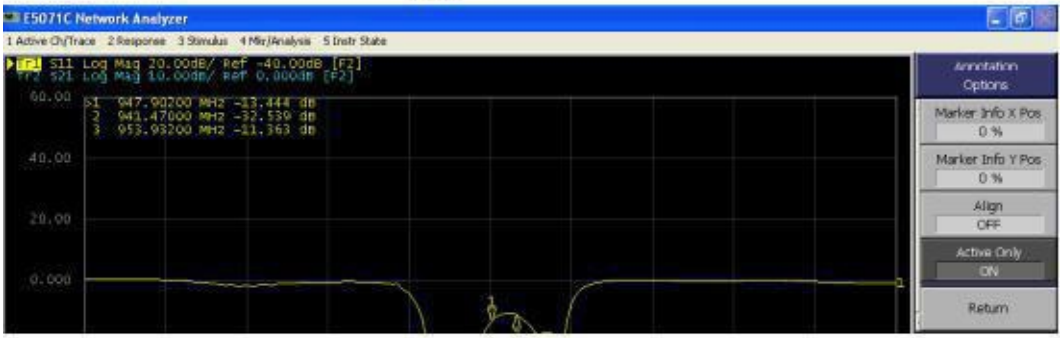
1. “ **Marker Fctn**”
2. “ **Annotation Options**”
3. “ **Align**”/



关闭

仅工作标记

打开

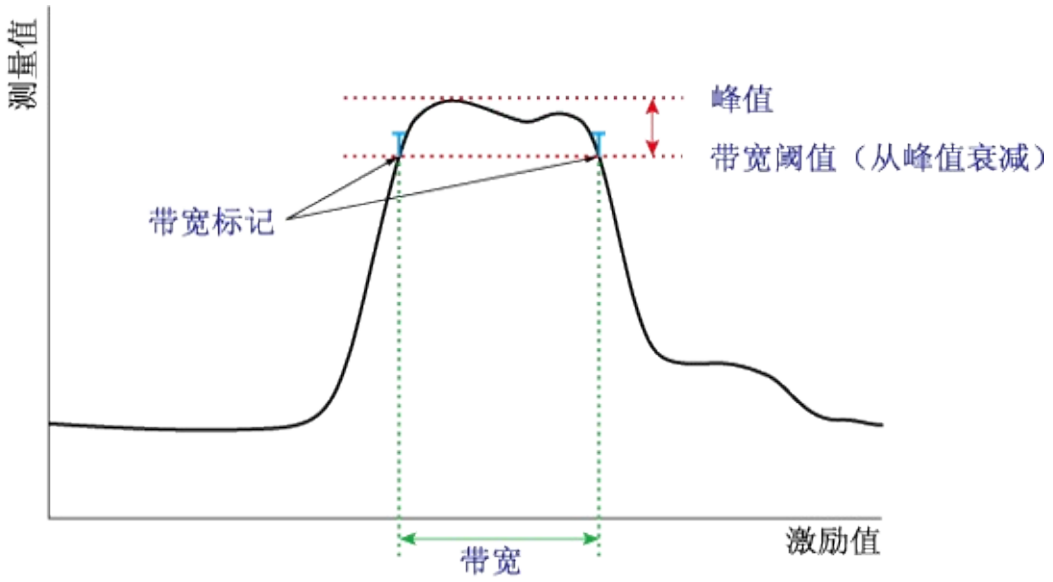


e5071c066

“ **Active Only OFF**”	
“ **Active Only ON**”	

1. “ **Marker Fctn**”
2. “ **Annotation Options**”
3. “ **Active Only**”/

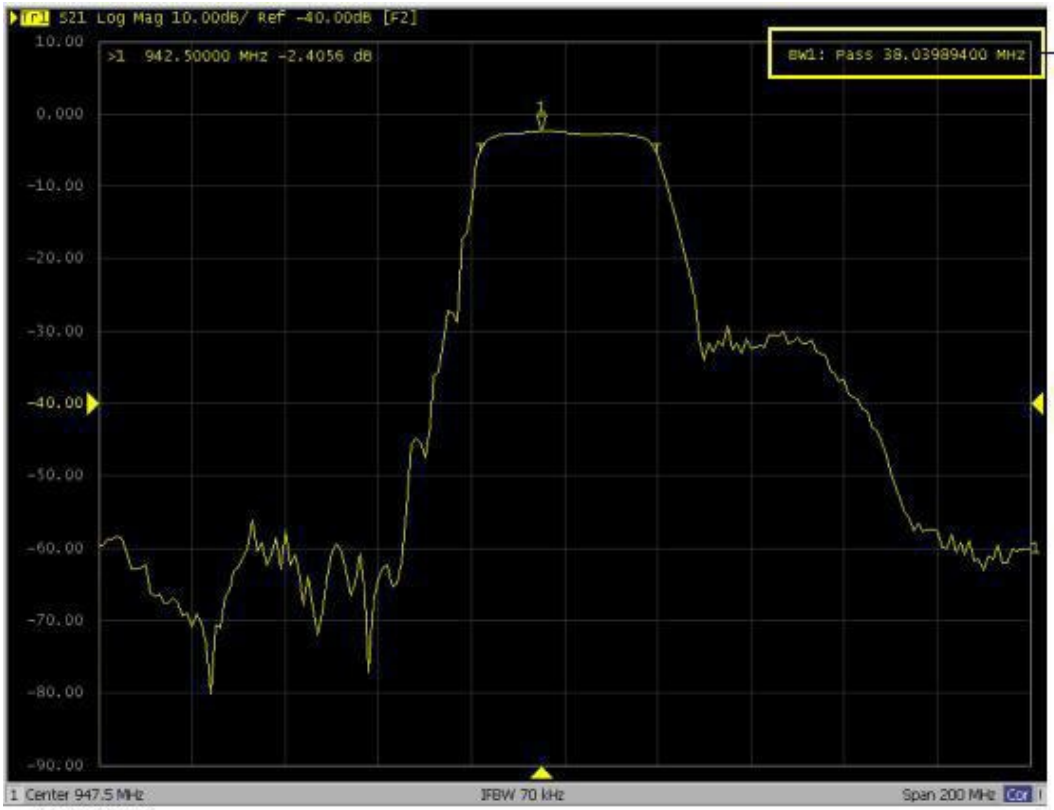
-
-
-
- /



e5071c450

BWn: “n”“Pass”“”Wide“Narrow”
“>Span”

/



e5071c076

“Fail Sign”//

1. “ **Analysis**”
2. “ **Bandwidth Limit**”
3. “ **Fail Sign**”

-
-

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“Analysis”
3. “ **Bandwidth Limit**”
4. “ **N dB Points**”N dB dB
5. “ **Min Bandwidth**”“ **Max Bandwidth**” Hz

/

/

/

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“Analysis”
3. “ **Bandwidth Limit**”
4. “ **BW Test**”“ *ON*”“ **BW Marker**” “ **BW Display**”

/

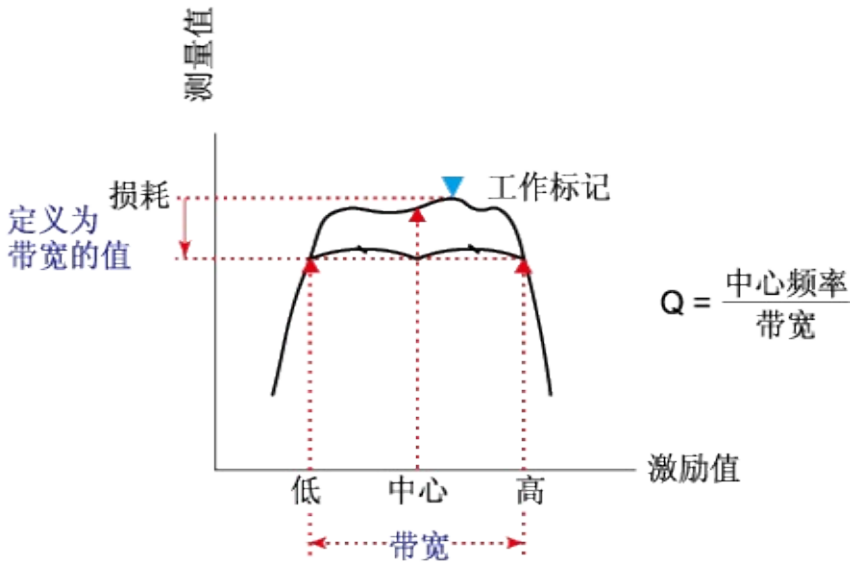
-
-

Data / Memory /	
Data * Memory *	
Data - Memory -	
Data + Memory +	

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”
2. “ **Display**”
3. “ **Data -> Mem**” ->
4. “ **Data Math**”
5. .
6. “ **Display**”
7. .
8.

-
-

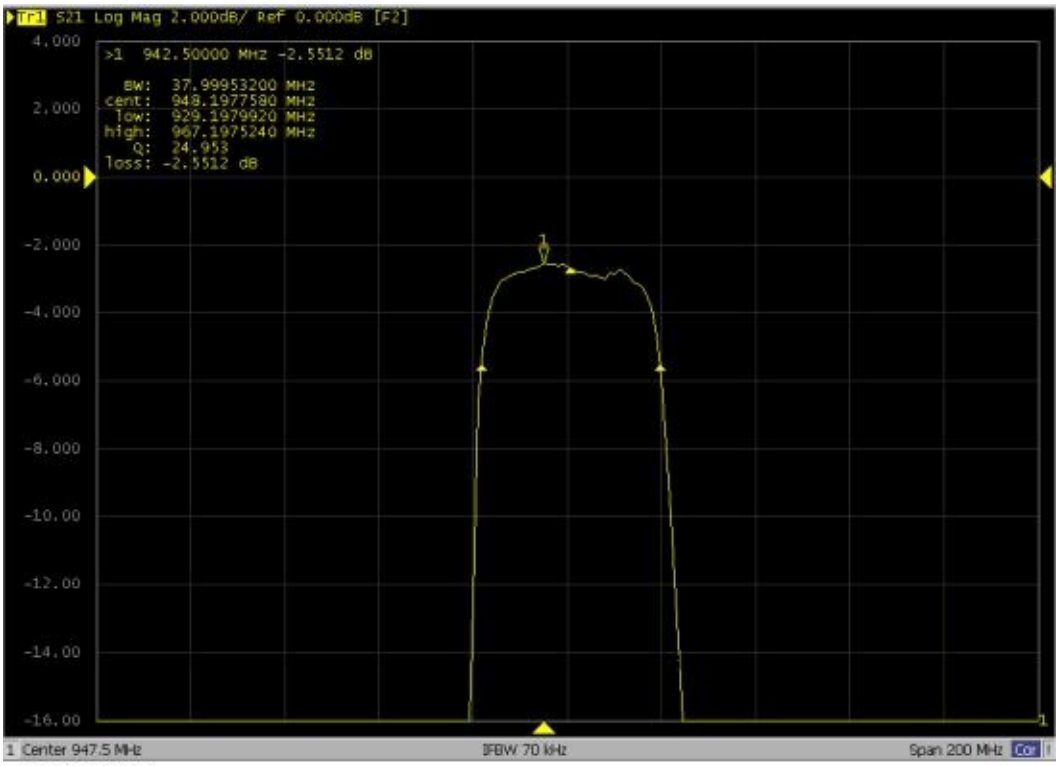
Q



e5071c392

	+ /2
(BW)	
Q	/

1. “ **loss**”
2. “ **Marker Search**”
3. “ **Bandwidth Value**”
4. “ **Bandwidth**”

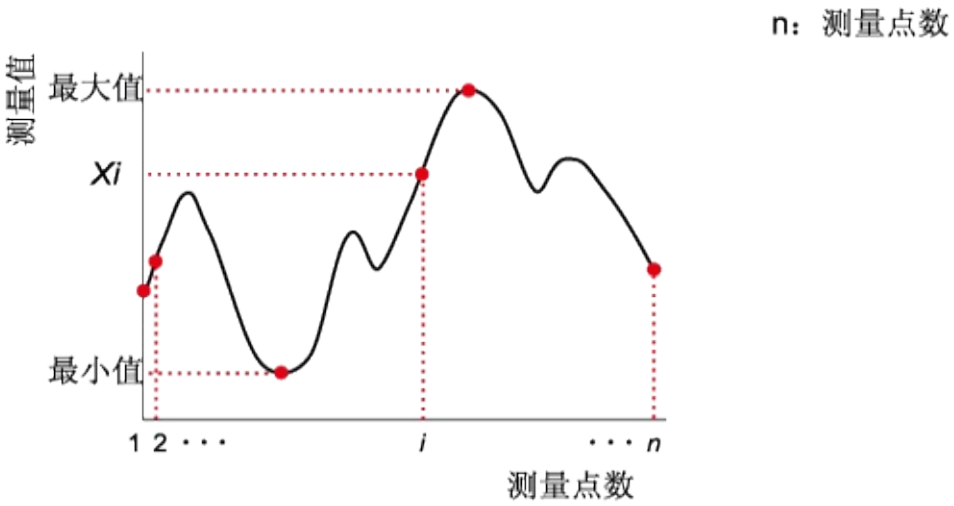


e5071c077

-

-
-

-



e5071c391

(*mean*)	$\frac{\sum_{i=1}^{n} x_i}{n}$ nx ¡ i
(*s. dev*)	$\sqrt{\frac{\sum_{i=1}^{n} (x_i - mean)^2}{n-1}}$ nx ¡ i mean

- (*p* - *p*)	Max - Min MaxMin

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”
2. “ **Marker Fctn**”
3. “ **Statistics**”

-
-
-
-

/

/

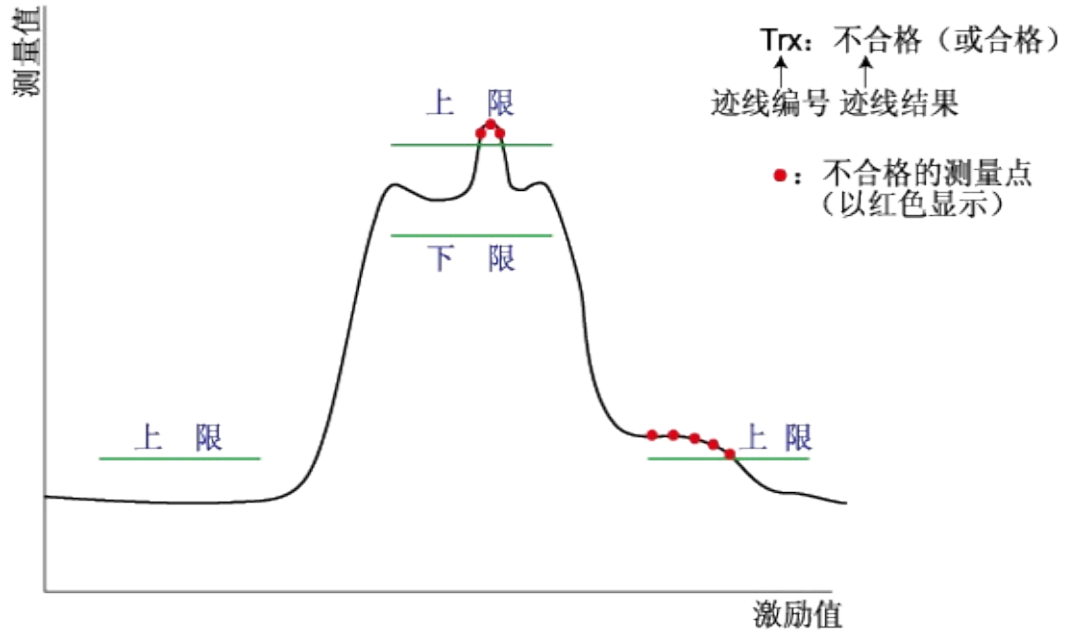
/

/

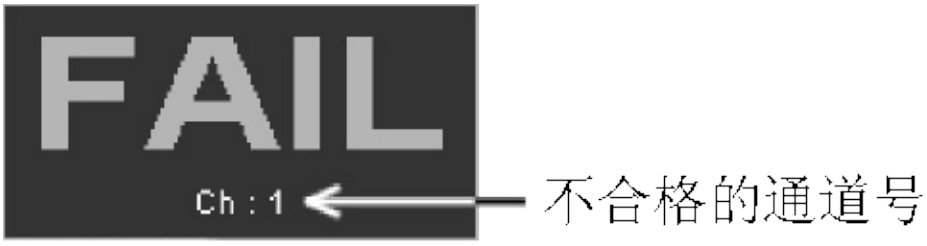
//

-
-

“Pass”“Fail”



e5071c446



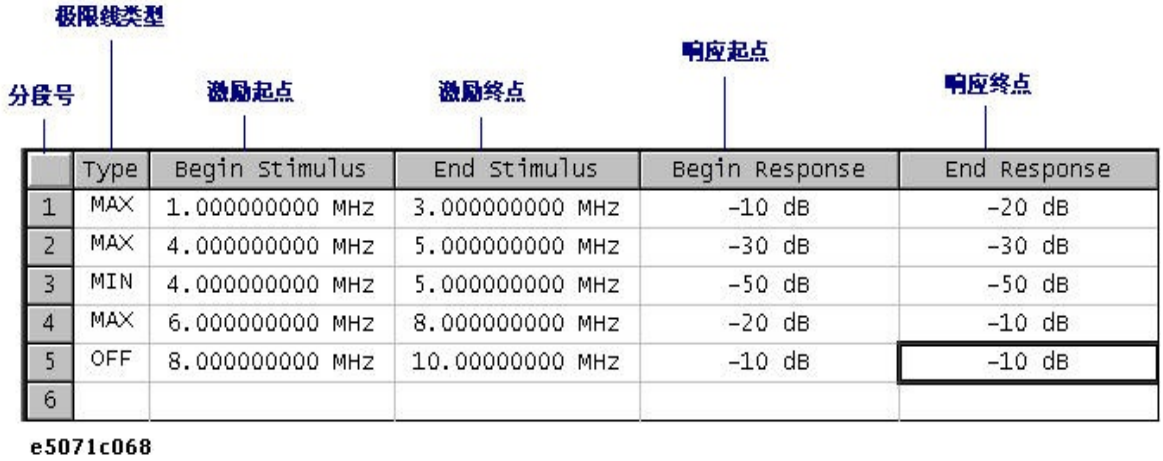
e5070bue065

/

1. “ **Analysis**”
2. “ **Limit Test**”

3. “ **Fail Sign**”/

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”
2. “ **Analysis**”“Analysis”
3. “ **Limit Test**”
4. “ **Edit Limit Line**”



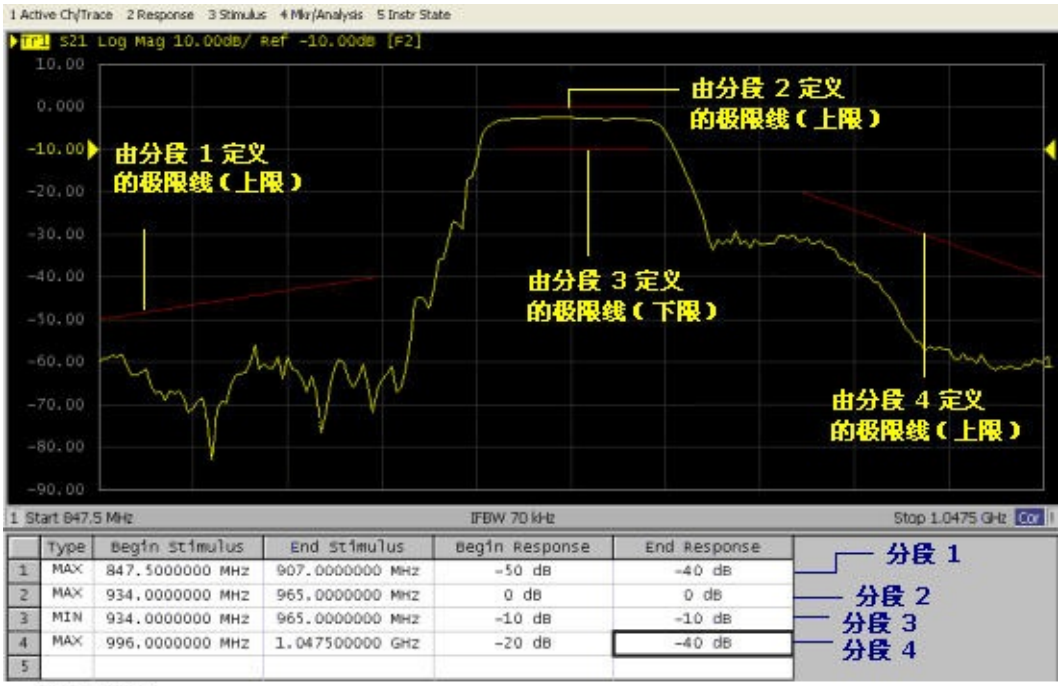
	Type	Begin Stimulus	End Stimulus	Begin Response	End Response
1	MAX	1.000000000 MHz	3.000000000 MHz	-10 dB	-20 dB
2	MAX	4.000000000 MHz	5.000000000 MHz	-30 dB	-30 dB
3	MIN	4.000000000 MHz	5.000000000 MHz	-50 dB	-50 dB
4	MAX	6.000000000 MHz	8.000000000 MHz	-20 dB	-10 dB
5	OFF	8.000000000 MHz	10.00000000 MHz	-10 dB	-10 dB
6					

5. / /
6. “ **Add**”

Type	**OFF** **MIN** **MAX**

Begin Stimulus	
End Stimulus	
Begin Response	
End Response	

- -500 G +500 G E5071C
- -500 M +500 M



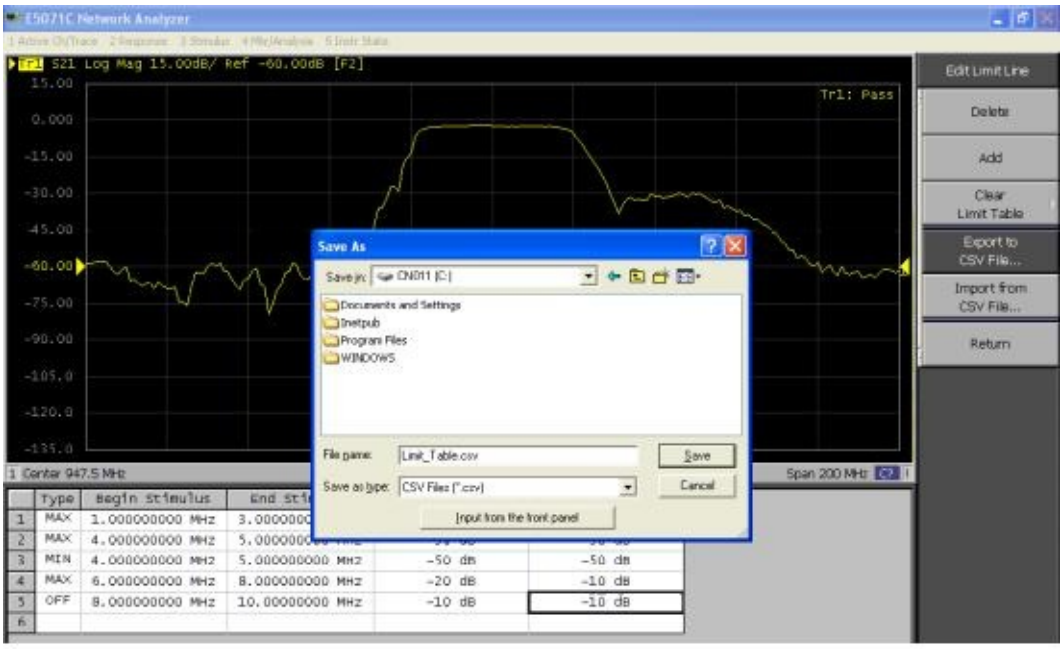
e5071c078

- (MAX)
- (MIN)

- E5071C 0 Begin Stimulus End Stimulus

/

CSV *.csv PC



e5071c069

1.
2. “Edit Limit Line”“ **Export to CSV File**” CSV “Save As”CSV*.csv
3. “ **Save**”
4. “Edit Limit Line”“ **Import from CSV File**” CSV “Open”CSV*.csv
5. “ **Open**”

CSV

-
-
-

•

"# Channel 1"

"# Trace 1"

Type, Begin Stimulus, End Stimulus, Begin Response, End Response

MAX, 200.0000000 MHz, 400.0000000 MHz, -100 dB, -100 dB

MAX, 490.0000000 MHz, 510.0000000 MHz, -10 dB, -10 dB

MIN, 490.0000000 MHz, 510.0000000 MHz, -20 dB, -20 dB

MIN, 600.0000000 MHz, 800.0000000 MHz, -100 dB, -100 dB

/

/

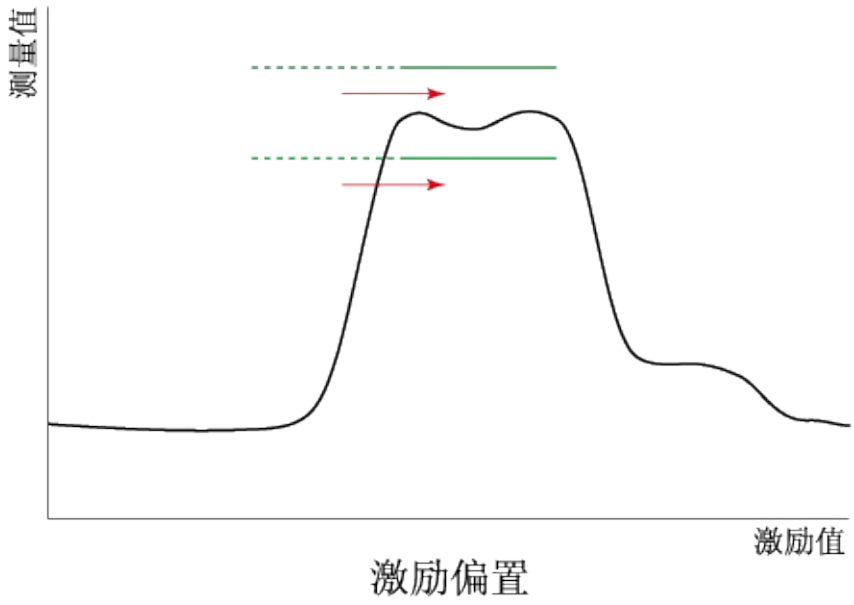
1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”
2. “ **Analysis**”“Analysis”
3. “ **Limit Test**”“Limit Test”

Limit Test	/
Limit Line	/
Edit Limit Line	

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”
2. “ **Analysis**”“Analysis”
3. “ **Limit Test**”
4. “ **Limit Line Offsets**”

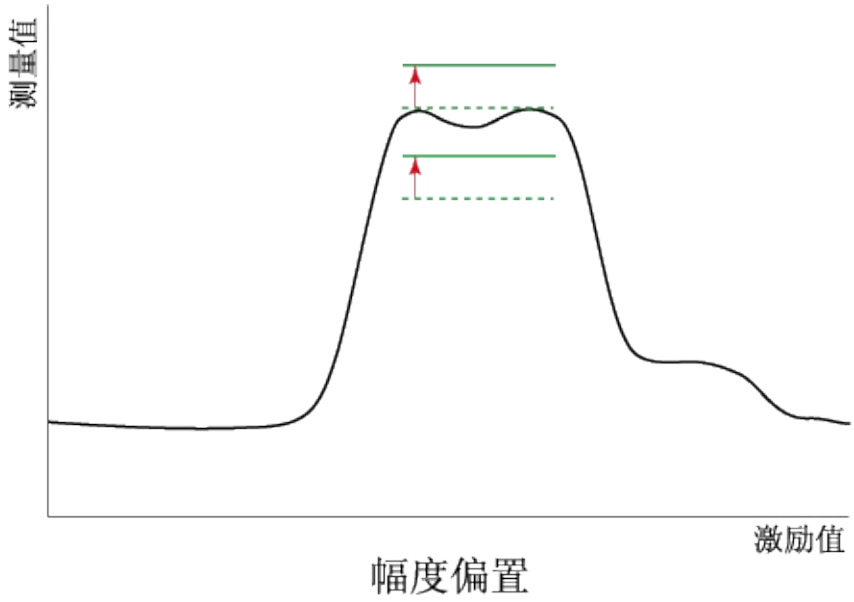
Stimulus Offset	“Stimulus offset”
Amplitude Offset	“Amplitude offset”
“Marker”->“Amp. Ofs.”	“ **Amplitude Offset**” “Marker amplitude offset”
Return	

Stimulus offset



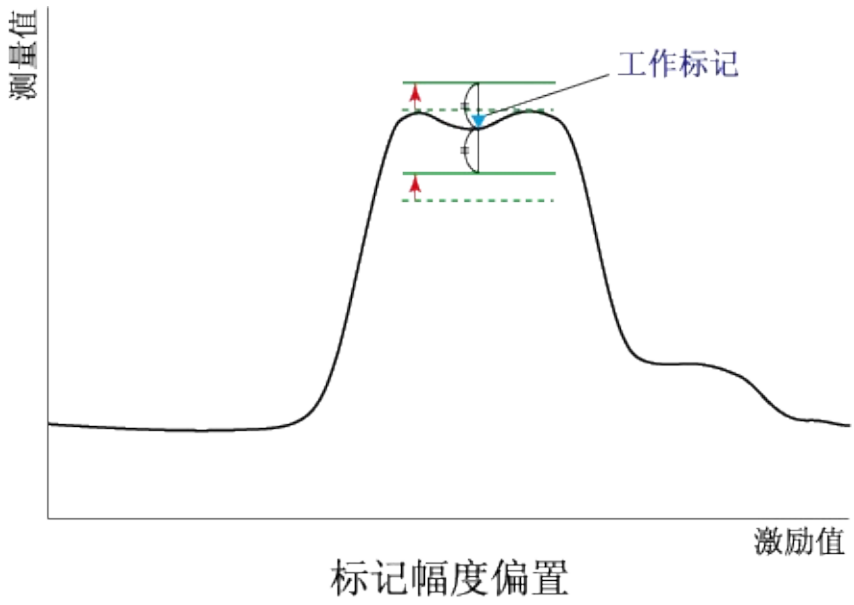
e5071c447

Amplitude offset



e5071c448

Marker amplitude offset



e5071c449

-
-
-
- “Edit Limit Line”“ **Clear Limit Table**”-“ **OK**”

-
-

S (S_{ab})

- (Z_r) (Y_r)

$$Z_r = Z_{0a} \times \frac{1 + S_{ab}}{1 - S_{ab}}, Y_r = \frac{1}{Z_r}$$

- (Z_t) (Y_t)

$$Z_t = \frac{2 \times \sqrt{Z_{0a} \times Z_{0b}}}{S_{ab}} - (Z_{0a} + Z_{0b}), Y_t = \frac{1}{Z_t}$$

- S (1/ S_{ab})

Z_{0a} a

Z_{0b} b

“ON” Z050 Ω

ON/OFF/

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”
2. “ **Analysis**”
3. “ **Conversion**”
4. “ **Conversion**”

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”
2. “ **Analysis**”
3. “ **Conversion**”>“ **Function**”
4.

“ON”

- .
- .
- .
- .
- /
- /
- .

12

“Pass”“Fail”“Pass”

/

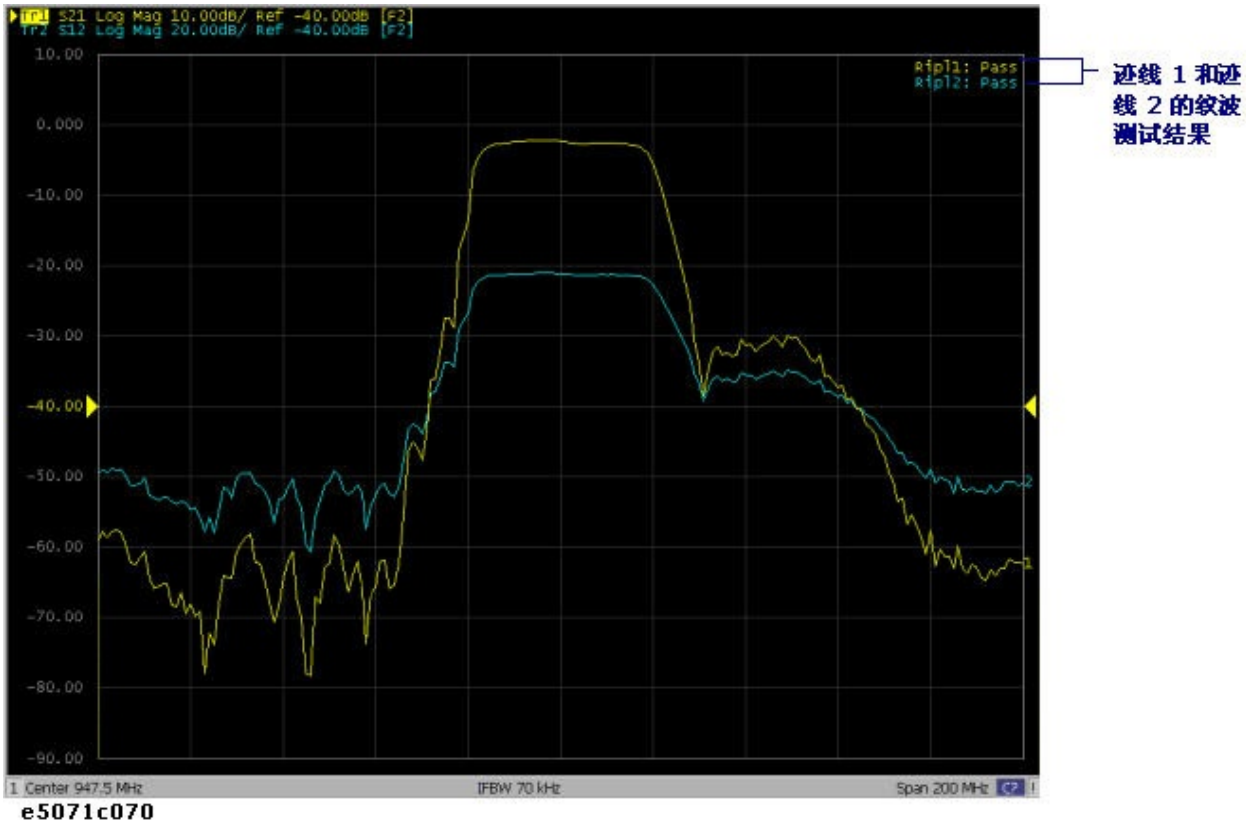
/

“Fail”“Fail” /“Fail”

“Pass”“Fail”

“Ripln:Pass”n“Ripln:Fail”nn Bn
Bn

“Ripl1:Pass”1 1 B3 2



e5071c070

“Fail”

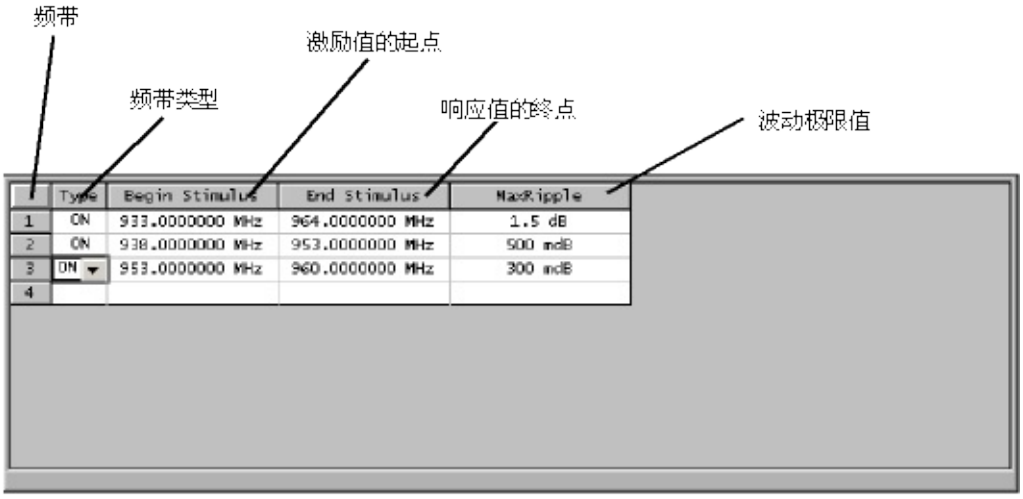
e5071c071

“Fail Sign”Fail //

1. “ **Analysis**”
2. “ **Ripple Limit**”
3. “ **Fail Sign**”

- “Fail”
-

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“Analysis”
3. “ **Ripple Limit**”
4. “ **Edit Ripple Limit**”



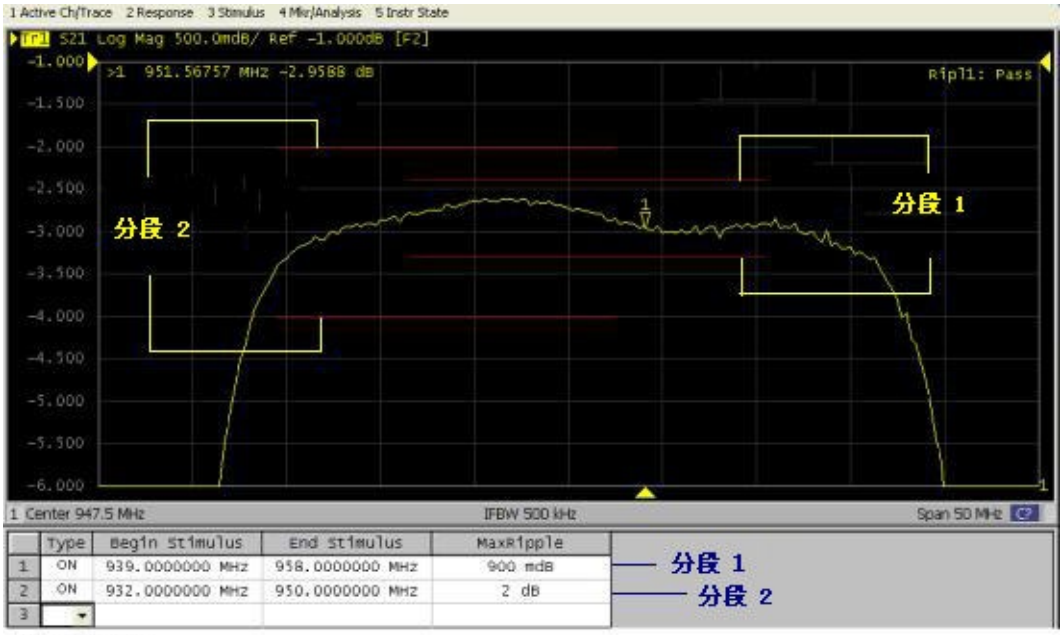
e5070bus7018

5. “Edit Ripple Limit”
6. “ **Add**”

Type	ON OFF **ON**

	OFF
Begin Stimulus	
End Stimulus	
Max Ripple	

- -500G +500G
- E5071C



e5071c079

-
- E5071C Begin Stimulus End Stimulus
-

/

csv *.csvcsv PC

//

1. .
2. “Edit Ripple Limit”“ **Export to CSV File**” CSV CSV *.csv
3. “ **Save**”
4. “Edit Ripple Limit”“ **Import from CSV File**” CSV “Open” CSV *.csv
5. “ **Open**”

-
-
-
-

"# Channel 1"

"# Trace 1"

Type, Begin Stimulus, End Stimulus, MaxRipple

ON, 933.0000000 MHz, 964.0000000 MHz, 1.5 dB

ON, 938.0000000 MHz, 953.0000000 MHz, 500 mdB

ON, 953.0000000 MHz, 960.0000000 MHz, 300 mdB

/

/

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“Analysis”
3. “ **Ripple Limit**”
4.

Ripple Test	/
Ripple Limit	/
Ripple Value	“off”“absolute value (difference between maximum and minimum values within the band) display”“margin (difference between absolute value of ripple and ripple limit) display”
Ripple Value Band	
Edit Ripple Limit	

-
-
-
- “Edit Ripple Line”“ **Clear Limit Table**”>“ **OK**”

-
-
-
-
-

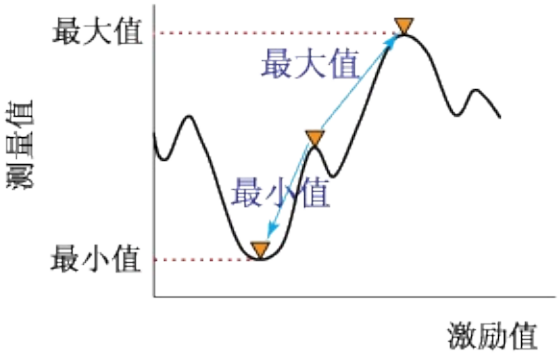
-
-
-
  -
  -
  -
-
  -
  -
  -

/

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Marker Search**”
3. “ **Search Range**”
4. “ **Couple**” /

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Marker Search**”
3. “ **Search Range**”
4. “ **Search Range**”
5. “ **Start**”
6. “ **Stop**”

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Marker Search**”
3. “ **Tracking**” /



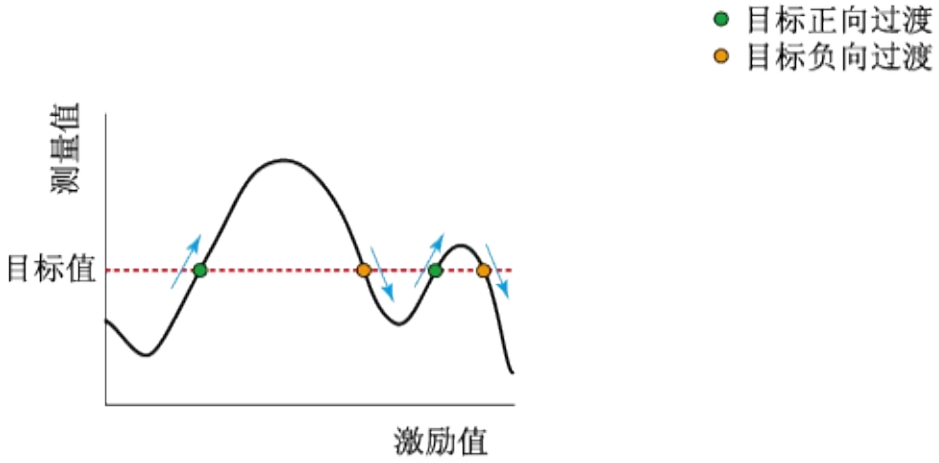
e5071c386

“ **Max**”	
“ **Min**”	

1. .
2. “ **Marker Search**”

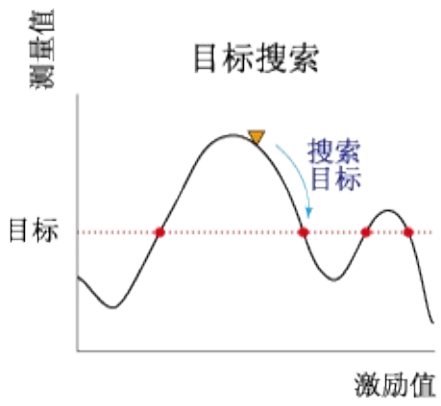
3.

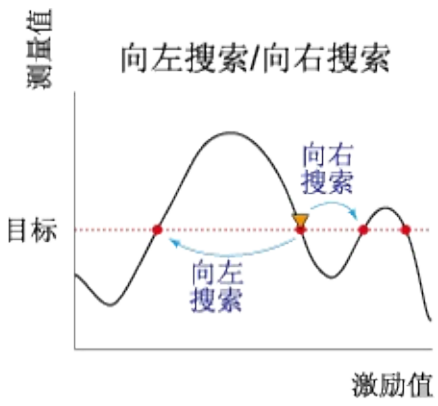
“ **Positive**”	
“ **Negative**”	



e5071c389

“ **Search Peak**”	“ **Positive**”“ **Both**” “ **Negative**”
“ **Search Left**”	
“ **Search Right**”	





e5071c387

1. .
2. “ **Marker Search**”
3. “ **Target**” >“**Target Value**”

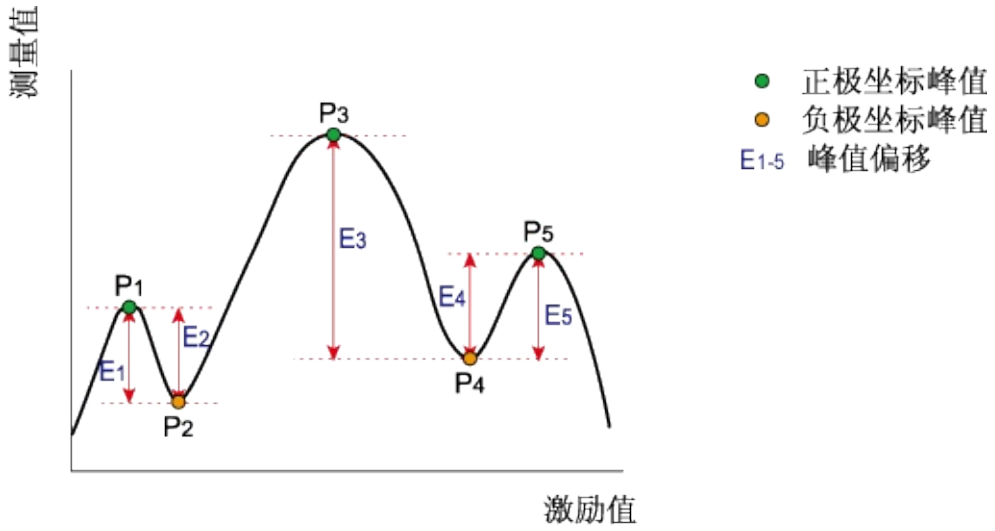
4.

5. “ **Target Transition**”

6.

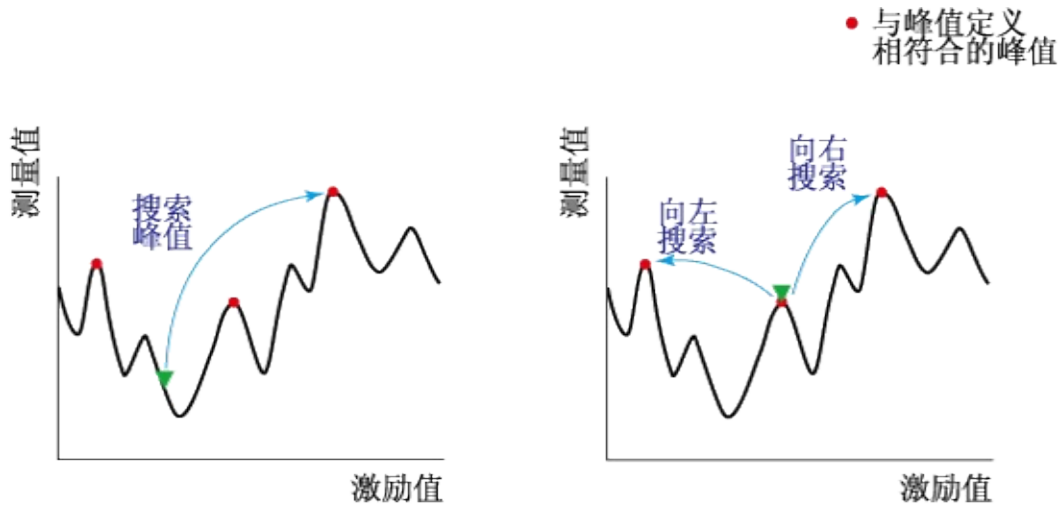
7.

“ **Positive**”	
“ **Negative**”	



e5071c390

“ **Search Peak**”	“ **Positive**”“ **Both**” “ **Negative**”
“ **Search Left**”	
“ **Search Right**”	



e5071c452

1. .
2. “ **Marker Search**”
3. “ **Peak**”>“ **Peak Excursion**”
4.
5. .
6.
7. .

-
-
-
-
-

- .
- _____
- .
- .
- “”
  _____
- .

E5071C

(State Only)	E5071C E5071C
(State & Cal)	E5071C E5071C
(State & Trace)	E5071C E5071C
(All)	E5071C E5071C

“”

E507xA/B E5071C /

			E5071C			
			230, 430	**235, 435**	**280, 480**	**285, 485**
	E5071C	**230, 430**	Y	N	Y	N
		235, 435	Y	Y	Y	Y
		280, 480	N	N	Y	N
		285, 485	N	N	Y	Y
	E5070A/B		Y	Y	Y	Y
	E5071A/B		N	N	Y	Y

Y N

- “ (**State Only**)”

		2	**4**

	2	Y	Y/c
	4	N	Y

Y

N

Y/c“ (**State Only**)”

/

- /“ **All**”
- //“ **Disp Only**”

-

“:SERV:SREV?”

-

1. “ **Save/Recall**”/
2. “ **Save Type**”
3. .

/

1. “ **Save/Recall**”/
2. “ **Channel/Trace**”//“ **All**”/“ **Disp Only**”

///

E5071C

1. “ **Save/Recall**”/
2. “ **Save State**”
3.

“**State01**” 01 “ **State08**” 08	
Autorec	E5071C “ **D:Autorec.sta**” D: **Autorec.sta**

	E5071C **Autore**
User Pres	
File Dialog.....	“ **Input from the front panel**”

- **D:\Autorec.sta** E5071C A: **A:\Autorec.sta** E5071C **A:\Autorec.sta** **Autor**
- (*) **backup.sta**

E5071C

“ **State &Trace**”“ **All**”“Manual”

1. “ **Save/Recall**”/
2. “ **Recall State**”

State01.sta State08.sta Autorec.sta

1. “ **State01**” 01“ **State08**” 08“ **Autorec**”

1. “ **File Dialog...**”...“Open”
2.
3. “ **Open**”

“ ”

“ **Recall by File Name**” D:\State “Open”

50 50 0 9 A Z 50

12 12 12 “...”

1. “ **Save/Recall**”/
2. “ **Recall by File Name**”
3. D:\State

E5071C

1	A
2	D
3	D “User” (**D:\UserPreset.sta**)

/

-
-
-
-

E5071C /A D

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Save/Recall**”/
3. “ **Save Channel**”
4. “ **State A**” A“ **State D**” D

*

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Save/Recall**”/
3. “ **Recall Channel**”
4.

1. “ **Save/Recall**”/
2. “ **Save Channel**”
3. “ **Clear States**”

- CSV
- Touchstone

CSV

E5071C CSV *.csv PC

"# Channel 1"

"# Trace 1"

Frequency,	Formatted Data,	Formatted Data
+3.00000000000E+005,	+1.41837599227E-002,	+1.43446459328E-006
+4.27985000000E+007,	+1.41275293412E-002,	+2.02407834551E-004
+8.52970000000E+007,	+1.41334093048E-002,	+4.00643331604E-004
+1.27795500000E+008,	+1.41240661092E-002,	+6.09250514670E-004
+1.70294000000E+008,	+1.41402155348E-002,	+8.05620003993E-004

E5071C

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”
2. “ **Save/Recall**”/
3. “ **Save Trace Data**”“Save As”
4.
5. “ **Save**”

Touchstone

1 4 E5071C Touchstone

Touchstone

“ - ”“ - ”“ - ”

“AUTO” (LogMag) (LinMag) (Real/Imag)
-

Touchstone PC Agilent Advanced Design System (ADS) ADS

s2p2 Touchstone E5071C
Touchstone E5071C

Touchstone

E5071C Touchstone s1ps2ps3p s4p Touchstone

Touchstone

Touchstone

*IDN? S

s1p s4p

s1p

!Agilent Technologies,E5071B,<ID>,<FW Revision>
!Date <Date>
!Data & Calibration Information
!Freq Sww:Method(Stat)
Hz S FMT R Z0

s4p

!Agilent Technologies,E5071B,<ID>,<FW Revision>
!Date <Date>
!Data & Calibration Information
!Freq Sww:Method(Stat) Sxw:Method(Stat) Syw:Method(Stat) Szw:Method(Stat)

.
.
Swz:Method(Stat) Sxz:Method(Stat) Syz:Method(Stat) Szz:Method(Stat)
Hz S FMT R Z0

Sww Szz	S w z
Method	S
Stat	S ONOFF -- ON = “ON” OFF = “OFF” -- =
FMT	RI = - MA = - DB = -
Z0	

S S Touchstone 0 - =
-200 dB

Touchstone

1 Touchstone

数据

Freq (1) Tab Saa. pri (1) Tab Saa. sec (1) ↙
Freq (2) Tab Saa. pri (2) Tab Saa. sec (2) ↙
⋮
Freq (N) Tab Saa. pri (N) Tab Saa. sec (N) ↙

a：选定的测试端口号
Freq (n)：测量点 n 的频率 [Hz]
Saa. pri (n)：在测量点 n 处测得的参数 Saa 的实部 (RI)、线性幅度 (MA) 或 dB (DB)
Saa. sec (n)：在测量点 n 处测得的参数 Saa 的实部 (RI) 或相位（MA、DB）
N：测量点号数
Tab：Tab
↙：换行

e5071c346

2 Touchstone

Freq (1) Tab Saa. pri (1) Tab Saa. sec (1) Tab Sba. pri (1) Tab Sba. sec (1) Tab Sab. pri (1) Tab Sab. sec (1) Tab Sbb. pri (1) Tab Sbb. sec (1) ↙
Freq (2) Tab Saa. pri (2) Tab Saa. sec (2) Tab Sba. pri (2) Tab Sba. sec (2) Tab Sab. pri (2) Tab Sab. sec (2) Tab Sbb. pri (2) Tab Sbb. sec (2) ↙
⋮
Freq (N) Tab Saa. pri (N) Tab Saa. sec (N) Tab Sba. pri (N) Tab Sba. sec (N) Tab Sab. pri (N) Tab Sab. sec (N) Tab Sbb. pri (N) Tab Sbb. sec (N) ↙

数据

a - b：选定的测试端口号（从 1 或 a 开始, 以升序排列）
Freq(n)：测量点 n 的频率 [Hz]
Sxy. pri(n)：在测量点 n 处测得的参数 Sxy 的实部 (RI)、线性幅度 (MA) 或 dB (DB)
Sxy. sec(n)：在测量点 n 处测得的参数 Sxy 的虚部 (RI) 或相位（MA、DB）
N：测量点号数
Tab：Tab
↙：换行

e5071c347

3 Touchstone

数据

```
Freq (1) Tab Saa. pri (1) Tab Saa. sec (1) Tab Sab. pri (1) Tab Sab. sec (1) Tab Sac. pri (1) Tab Sac. sec (1) ↙
         Tab Sba. pri (1) Tab Sba. sec (1) Tab Sbb. pri (1) Tab Sbb. sec (1) Tab Sbc. pri (1) Tab Sbc. sec (1) ↙
         Tab Sca. pri (1) Tab Sca. sec (1) Tab Scb. pri (1) Tab Scb. sec (1) Tab Scc. pri (1) Tab Scc. sec (1) ↙
Freq (2) Tab Saa. pri (2) Tab Saa. sec (2) Tab Sab. pri (2) Tab Sab. sec (2) Tab Sac. pri (2) Tab Sac. sec (2) ↙
         Tab Sba. pri (2) Tab Sba. sec (2) Tab Sbb. pri (2) Tab Sbb. sec (2) Tab Sbc. pri (2) Tab Sbc. sec (2) ↙
         Tab Sca. pri (2) Tab Sca. sec (2) Tab Scb. pri (2) Tab Scb. sec (2) Tab Scc. pri (2) Tab Scc. sec (2) ↙
                                                  :
Freq (N) Tab Saa. pri (N) Tab Saa. sec (N) Tab Sab. pri (N) Tab Sab. sec (N) Tab Sac. pri (N) Tab Sac. sec (N) ↙
         Tab Sba. pri (N) Tab Sba. sec (N) Tab Sbb. pri (N) Tab Sbb. sec (N) Tab Sbc. pri (N) Tab Sbc. sec (N) ↙
         Tab Sca. pri (N) Tab Sca. sec (N) Tab Scb. pri (N) Tab Scb. sec (N) Tab Scc. pri (N) Tab Scc. sec (N) ↙
```

a - c：选定的测试端口号（从 1 或 a 开始，以升序排列）
Freq(n)：测量点 n 的频率 [Hz]
Sxy. pri(n)：在测量点 n 处测得的参数 Sxy 的实部 (RI)、线性幅度 (MA) 或 dB (DB)
Sxy. sec(n)：在测量点 n 处测得的参数 Sxy 的虚部 (RI) 或相位（MA、DB）
N：测量点号数
Tab：Tab
↙：换行

e5071c348

4 Touchstone

```
Freq (1) Tab S11. pri (1) Tab S11. sec (1) Tab S12. pri (1) Tab S12. sec (1) Tab S13. pri (1) Tab S13. sec (1) Tab S14. pri (1) Tab S14. sec (1) ↙
         Tab S21. pri (1) Tab S21. sec (1) Tab S22. pri (1) Tab S22. sec (1) Tab S23. pri (1) Tab S23. sec (1) Tab S24. pri (1) Tab S24. sec (1) ↙
         Tab S31. pri (1) Tab S31. sec (1) Tab S32. pri (1) Tab S32. sec (1) Tab S33. pri (1) Tab S33. sec (1) Tab S34. pri (1) Tab S34. sec (1) ↙
         Tab S41. pri (1) Tab S41. sec (1) Tab S42. pri (1) Tab S42. sec (1) Tab S43. pri (1) Tab S43. sec (1) Tab S44. pri (1) Tab S44. sec (1) ↙
Freq (2) Tab S11. pri (2) Tab S11. sec (2) Tab S12. pri (2) Tab S12. sec (2) Tab S13. pri (2) Tab S13. sec (2) Tab S14. pri (2) Tab S14. sec (2) ↙
         Tab S21. pri (2) Tab S21. sec (2) Tab S22. pri (2) Tab S22. sec (2) Tab S23. pri (2) Tab S23. sec (2) Tab S24. pri (2) Tab S24. sec (2) ↙
         Tab S31. pri (2) Tab S31. sec (2) Tab S32. pri (2) Tab S32. sec (2) Tab S33. pri (2) Tab S33. sec (2) Tab S34. pri (2) Tab S34. sec (2) ↙
         Tab S41. pri (2) Tab S41. sec (2) Tab S42. pri (2) Tab S42. sec (2) Tab S43. pri (2) Tab S43. sec (2) Tab S44. pri (2) Tab S44. sec (2) ↙
                                                  :
Freq (N) Tab S11. pri (N) Tab S11. sec (N) Tab S12. pri (N) Tab S12. sec (N) Tab S13. pri (N) Tab S13. sec (N) Tab S14. pri (N) Tab S14. sec (N) ↙
         Tab S21. pri (N) Tab S21. sec (N) Tab S22. pri (N) Tab S22. sec (N) Tab S23. pri (N) Tab S23. sec (N) Tab S24. pri (N) Tab S24. sec (N) ↙
         Tab S31. pri (N) Tab S31. sec (N) Tab S32. pri (N) Tab S32. sec (N) Tab S33. pri (N) Tab S33. sec (N) Tab S34. pri (N) Tab S34. sec (N) ↙
         Tab S41. pri (N) Tab S41. sec (N) Tab S42. pri (N) Tab S42. sec (N) Tab S43. pri (N) Tab S43. sec (N) Tab S44. pri (N) Tab S44. sec (N) ↙
```

数据

Freq(n)：测量点 n 的频率 [Hz]
Sxy. pri(n)：在测量点 n 处测得的参数 Sxy 的实部 (RI)、线性幅度 (MA) 或 dB (DB)
Sxy. sec(n)：在测量点 n 处测得的参数 Sxy 的虚部 (RI) 或相位（MA、DB）
N：测量点号数
Tab : Tab
↙ : 换行

e5071c349

Touchstone

Touchstone

- Z0 Z0 Z0
- S
- Touchstone
-

Touchstone

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”
2. “ **Save/Recall**”/
3. “ **SnP**”>“**SnP Format**”SnP
4.

Auto	(LogMag) (LinMag) (Real/Imag) -
LogMag/Angle /	“ - ”
LinMag/Angle /	“ - ”
Real/Imaginary /	“ - ”

5. “ **s1p**”“ **s2p**”“ **s3p**”“ **s4p**”
6. 5 “ **s3p**”“ **1-2-3**”“ **1-2-4**”“ **1-3-4**”“ **2-3-4**” 1 2 4 “ **1-2-4**”

7. “Save As”
8. “ **Save**”

- Touchstone Touchstone “HOLD”

-
-

E5071C (.bmp) (.png) PC

1.

2. “ **System**”“ **System**”

3. “ **Dump Screen Image**”“Save As”“Bitmap Files (*.bmp)” (*.bmp)“Portable Network Graphics (*.png)” (*.png)

4.

5.

6. “ **Save**” E5071C

-
-
-

E5071C USB E5071C

/

//

“**System**”“ **System**”

/

1. E5071C
2. E5071C
3. E5071C
4. “ **System**”
5. “ **Printer Setup**”“Printers”
6. ( ) “Printers”“File”“ **Set as Default Printer**”
7. “File”“ **Printing Preferences...**”... “Printing Preferences”“Page Size”“
8. “File”“ **Close** ”

1.
2. “ **System**”
3. “ **Invert Image**” [OFF] [ON]
4. “ **Print**”
5. “ **Abort Printing**”

“Printers Folder” “ **Cancel**” “Printers Folder”

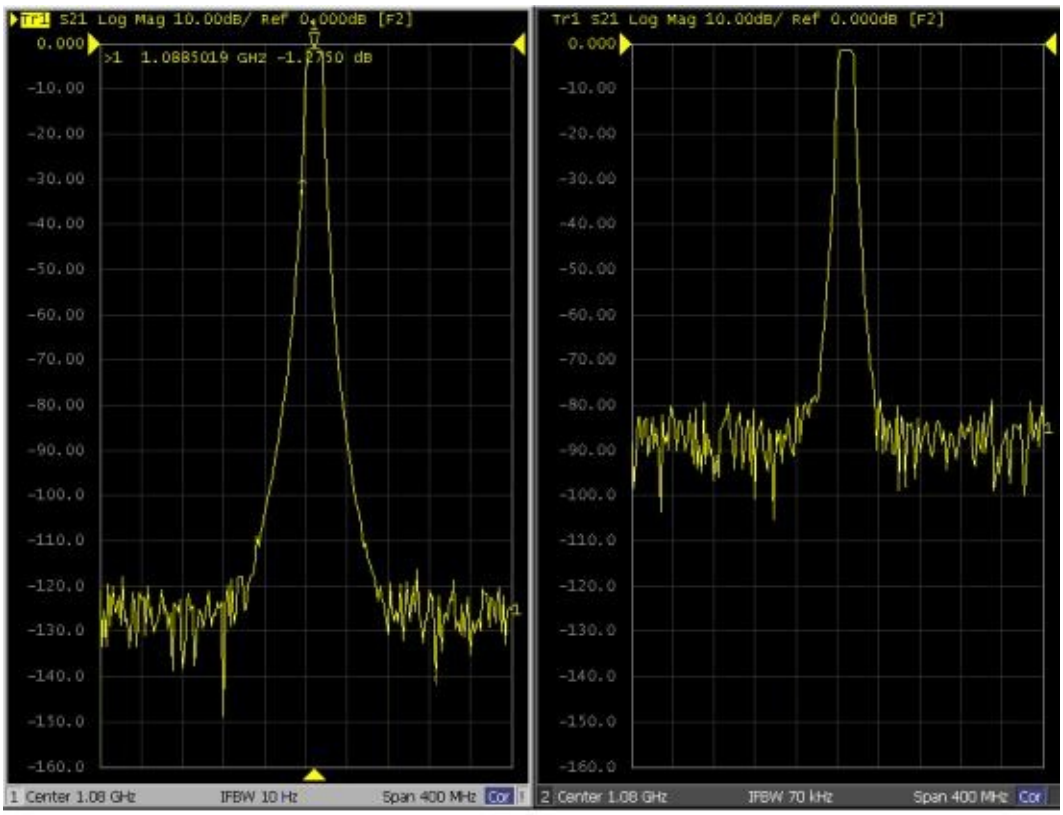
-
-
-
-
-

-
-

- IF
-

IF

IF IF 1/10 10 dB



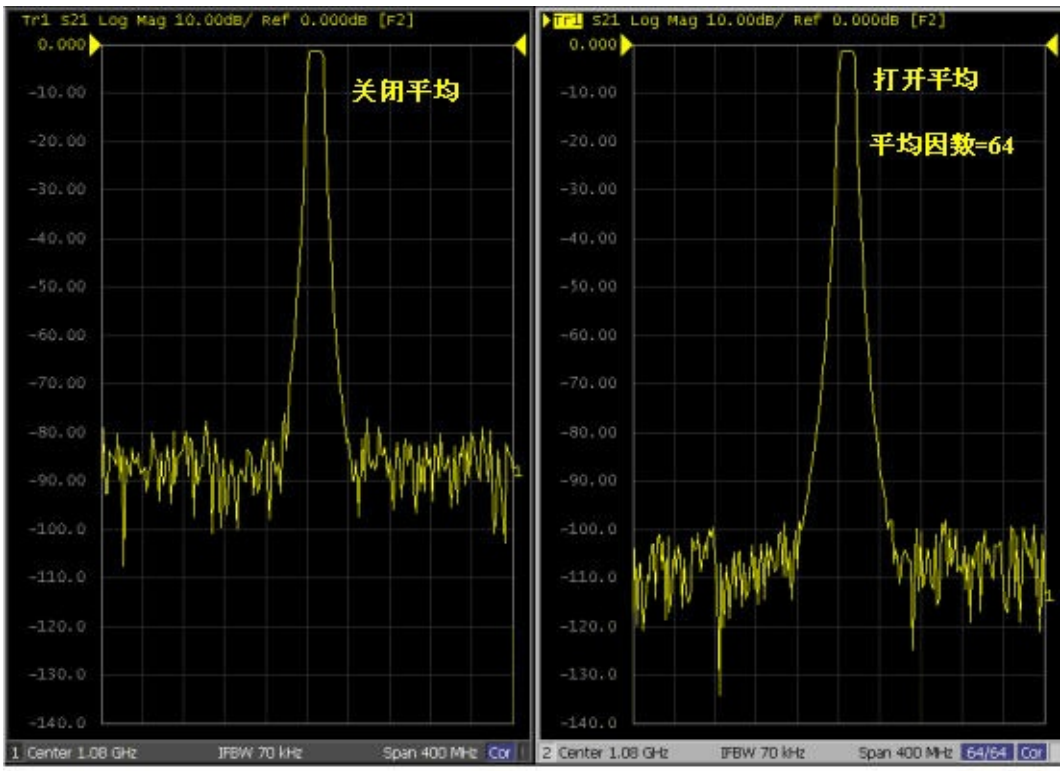
e5071c080

IF

1. “ **Channel Next**”/“ **Channel Prev**” IF
2. “ **Avg**”
3. “ **IF Bandwidth**”IF IF

$$A_n = \frac{S_n}{F} + \left(1 - \frac{1}{F}\right) \times A_{n-1}$$

- **A_n** = n
- **S_n** = n
- **F** = 1 999



e5071c081

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Avg**”
3. “ **Avg Factor**”

4. “ **Averaging**”

“ **Averaging Restart**” n 1

-
-
- IF

E5071C Network Analyzer
1 Active Ch/Trace 2 Response 3 Stimulus 4 Mkr/Analysis 5 Instr State
Close
F2 S12 Log Mag 20.00dB/ Ref -40.00dB [F2]
60.00
40.00
20.00
0.000
-20.00
-40.00
-60.00
-80.00
Average
Averaging Restart
Avg Factor 16
Averaging OFF
Avg Trigger OFF
Smo Aperture 1.5000 %
Smoothing OFF
IF Bandwidth 70 kHz
Return
关闭
平滑
打开
E5071C Network Analyzer
1 Active Ch/Trace 2 Response 3 Stimulus 4 Mkr/Analysis 5 Instr State
F2 S12 Log Mag 20.00dB/ Ref -40.00dB [F2 Smo]
60.00
40.00
20.00
0.000
-20.00
-40.00
-60.00
-80.00
Average
Averaging Restart
Avg Factor 16
Averaging OFF
Avg Trigger OFF
Smo Aperture 1.5000 %
Smoothing ON
IF Bandwidth 70 kHz
Return

e5071c075

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Avg**”
3. “ **Smo Aperture**” (%)
4. “ **Smoothing**”

-
-
-
-

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Scale**”
3. “ **Electrical Delay**”
4.
5. “ **Media**” 0

Coaxial	
Waveguide	

6. “ **Waveguide**”“ **Cutoff Frequecy**”

“ **Waveguide**”

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2.
3. “ **Maker Fctn**”
4. “ **Marker”->“Delay** ” 20%

- 360° +360 °

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Scale**”
3. “ **Phase Offset**” (°)

0.66 (?Ã r)

$$速度因数 = \frac{1}{\sqrt{\varepsilon_r}}$$

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Velocity Factor**”

DUT DUT DUT

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Port Extensions**”

Extension Port 1 1	1
Extension Port 2 2	2
Extension Port 3 3	3
Extension Port 4 4	4

1.

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Port Extensions**”>“ **Loss**”
4. “ **Select Port**”
5. “ **Loss1 [OFF]**” 1 []“ **Loss1 [ON]**” 1 []
   “ **Loss1**” 1“ **Freq1**” 1
6. “ **Loss2 [OFF]**” 2 []“ **Loss [ON]**” []
   “ **Loss2**” 2“ **Freq2**” 2
7. 4 6

- Loss 1

$$\mathrm{Loss}(f) = \mathrm{Loss1} \times \sqrt{\left(\frac{f}{\mathrm{Freq1}}\right)}$$

- Loss 1 Loss 2

$$\mathrm{Loss}(f) = \mathrm{Loss1} \times \left(\frac{f}{\mathrm{Freq1}}\right)^{n}$$

$$n = \frac{\log_{10}\left|\frac{\mathrm{Loss1}}{\mathrm{Loss2}}\right|}{\log_{10}\frac{\mathrm{Freq1}}{\mathrm{Freq2}}}$$

Loss1 Loss2

2.

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Port Extensions**”>“ **Loss**”

4. “ **Select Port**”
5. “ **Loss at DC**”
6. 4 5

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Port Extensions**”
4. “ **Extensions**”

/

1.

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Port Extensions**”>“ **Auto Port Extension**”>“ **Select Ports**”

2.

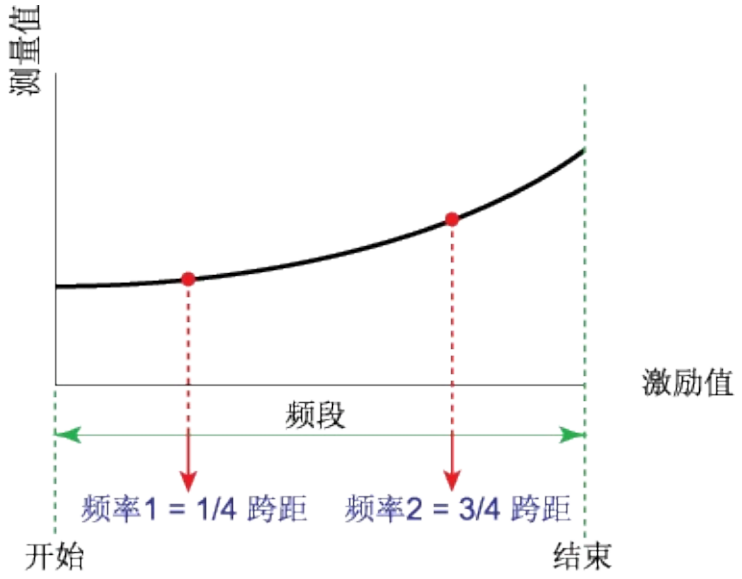
1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”

3. “ **Port Extensions**”>“ **Auto Port Extension**”>“ **Select Ports**”>“ **Method**”

Current Span	
Active Marker	Loss1 Loss2
User Span	

4. “ **User Span**”“ **User Span Start**”“ **User Span Stop**”

- “ **Current Span**”“ **User Span**” 1/4
  “ **Freq1**” 1 3/4 “ **Freq2**” 2
- /



e5071c466

3.

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Port Extensions**”>“ **Auto Port Extension**”>“ **Select Ports**”>“ **Include Loss**”

/

4.

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Port Extensions**”>“ **Auto Port Extension**”>“ **Adjust Mismatch**”

/

5. /

/

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Cal**”
3. “ **Port Extensions**”>“ **Auto Port Extension**”
4. “ **Measure OPEN**”
5. “ **Measure SHORT**”

6. /

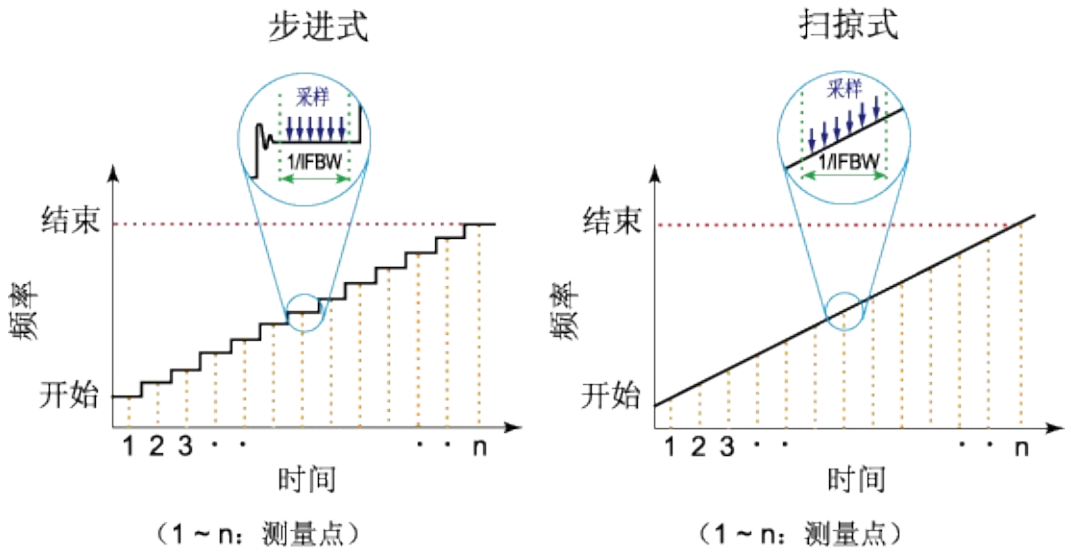
/ GPIB

-
- LCD
-

E5071C

		DUT

1. “ **Sweep Setup**”
2. “ **Sweep Mode**”
3. “ **Stepped**”“ **Swept**”



e5071c397

IF 30 MHz

- DUT DUT
-
-

DUT

DUT (F) DUT (△T) DUT DUT (

$$\Delta F = \frac{dF}{dt} \times \Delta T$$

DUT

DUT

LCD

LCD

LCD

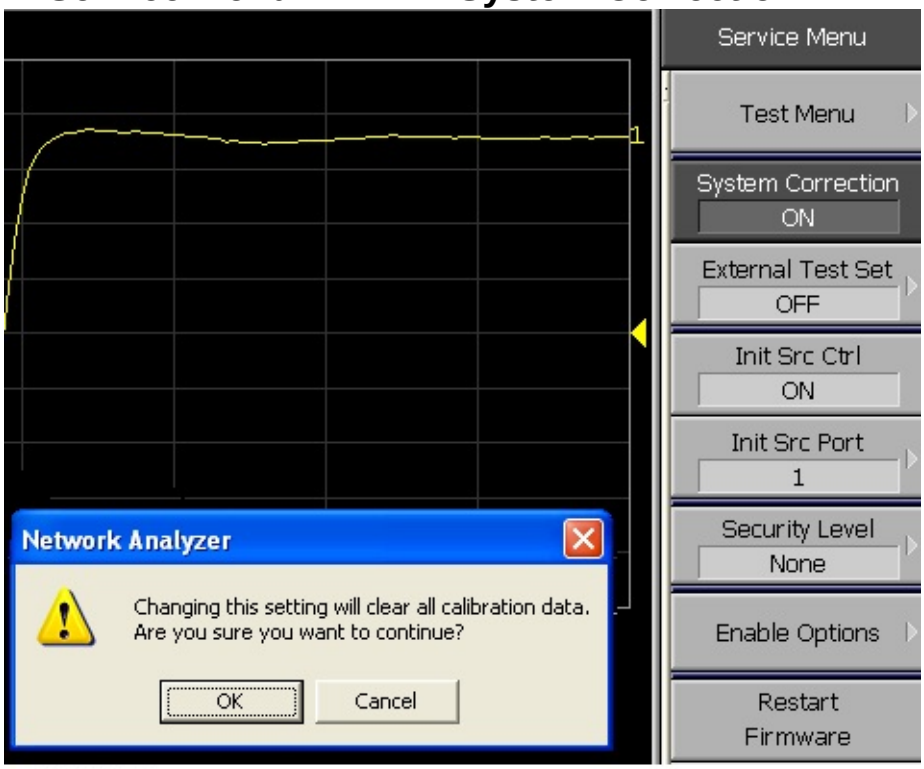
1. “ **Display**”
2. “ **Update**”/ LCD

LCD “ **Update Off**”

E5071C “ **Cal**”

/

1. “ **System**”
2. “ **Service Menu**”>“ **System Correction**”



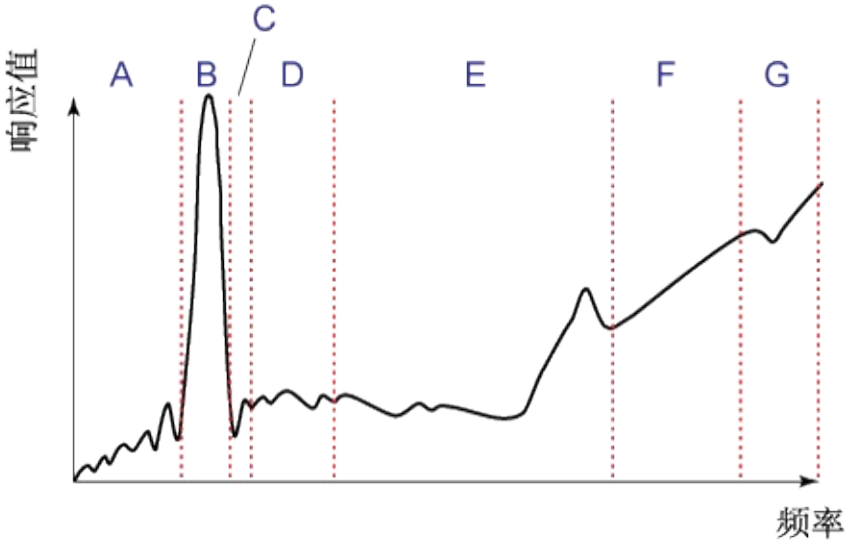
e5071c074

3. “ **OK**”/

- .
- .
- .
- /
- /

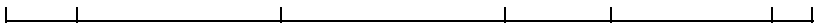
IF

A G



e5071c393

				IF	
A	440 MHz	915 MHz	50	50 kHz	
B	915 MHz	980 MHz	130	70 kHz	
C	980 MHz	1.035 GHz	60	50 kHz	
E	1.07 GHz	2 GHz	100	70 kHz	
G	2.6 GHz	3 GHz	40	70 kHz	

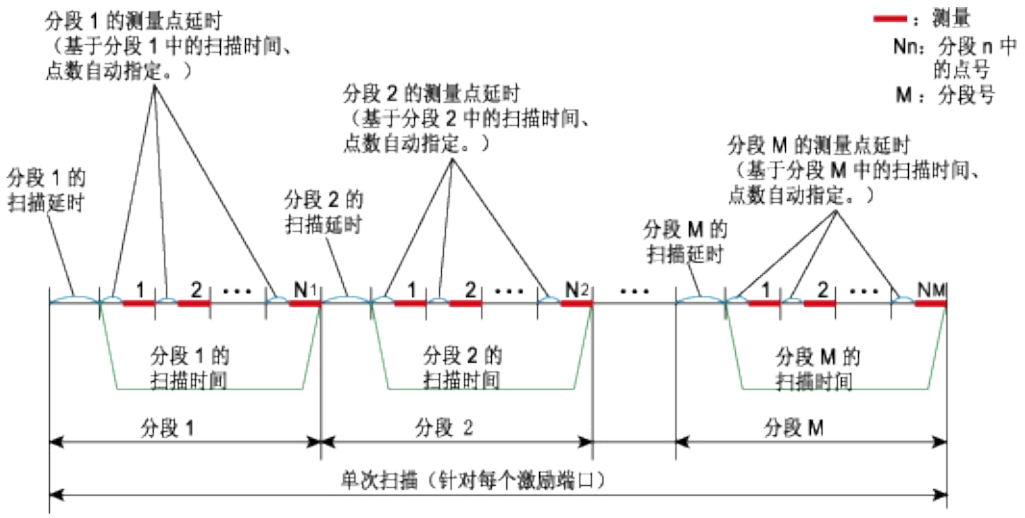


-
- 1 9 kHz 3 GHz 8.5 GHz
- 2 1601
- 1 1601
- 2 1601
- 1 201

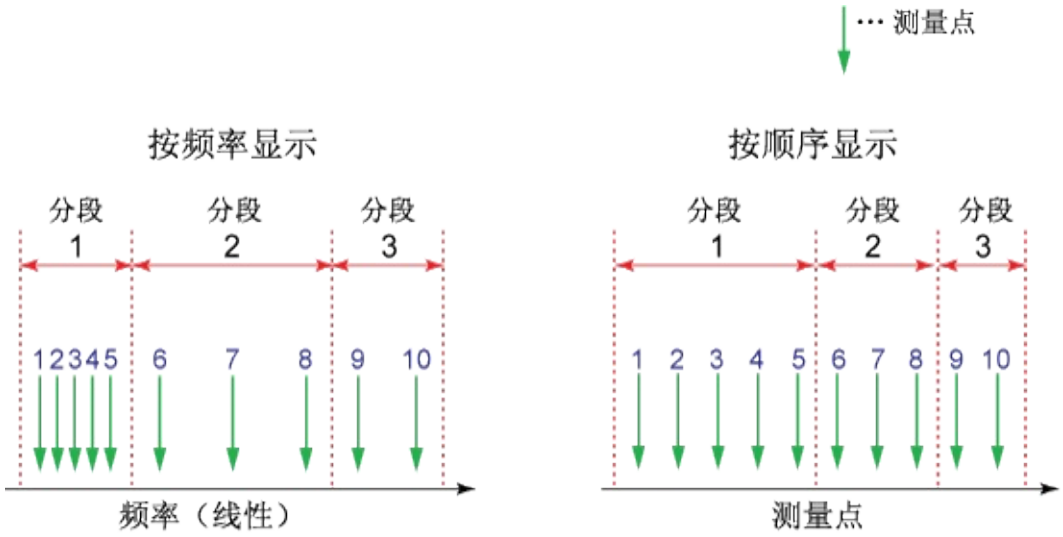
IF

/

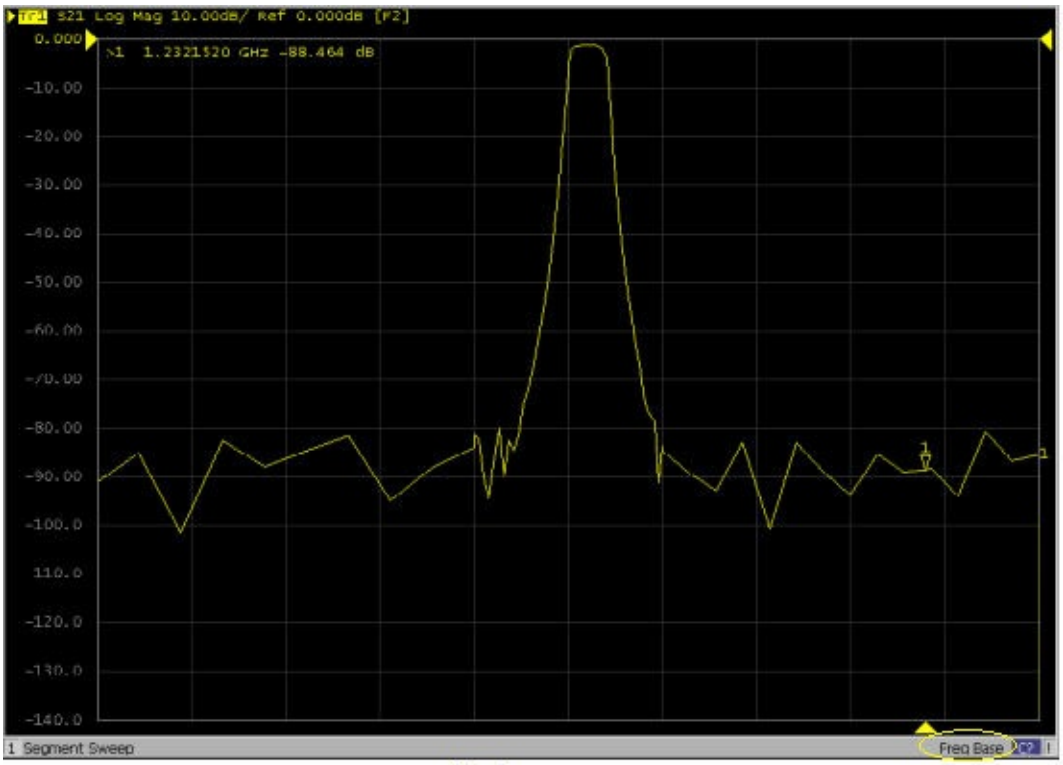
IF	/ IF “ **Avg**” >“ **IF Bandwidth**”IF
	/“ **Sweep Setup**” >“ **Power**”
	0
	/“ **Sweep Setup**” >“ **Sweep Mode**”


分段 1 的测量点延时
（基于分段 1 中的扫描时间、
点数自动指定。）
：测量
Nn：分段 n 中
的点号
M：分段号
分段 2 的测量点延时
（基于分段 2 中的扫描时间、
点数自动指定。）
分段 M 的测量点延时
（基于分段 M 中的扫描时间、
点数自动指定。）
分段 1 的
扫描延时
分段 2 的
扫描延时
分段 M 的
扫描延时
1
2
N1
N2
NM
分段 1 的
扫描时间
分段 2 的
扫描时间
分段 M 的
扫描时间
分段 1
分段 2
分段 M
单次扫描（针对每个激励端口）

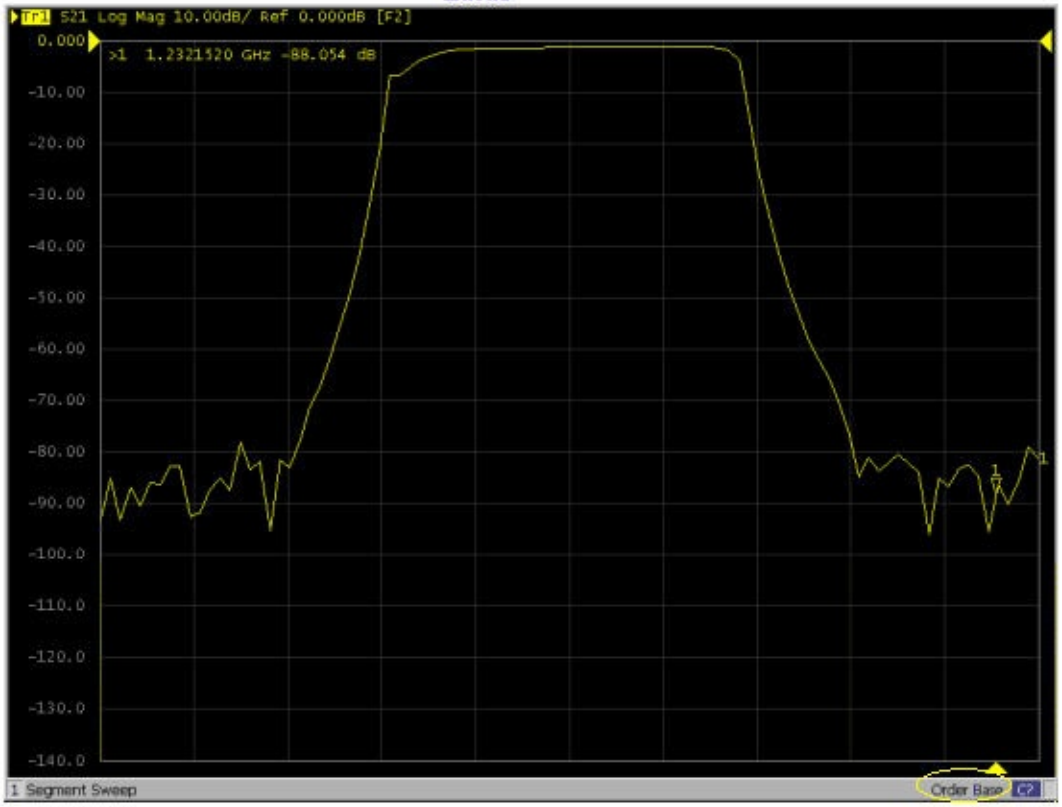
e5071c308


··· 测量点
按频率显示
分段 1
分段 2
分段 3
1 2 3 4 5 6 7 8 9 10
频率（线性）
按顺序显示
分段 1
分段 2
分段 3
1 2 3 4 5 6 7 8 9 10
测量点

e5071c394



显示 —— 按频率 / 按顺序



e5071c082

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”
3. “ **Edit Segment Table**”
4. IF

注意 “ ”“ **Focus**”

Freq Mode	//
List IFBW IFBW	IF /“ **IFBW**”
List Power	/“ **Power**”
List Delay	/“ **Delay**”
List Sweep Mode	/“ **Sweep Mode**”
List Time	/“ **Time**”

5.

Start	
Stop	

Center	
Span	
Points	
IFBW	IF
Power	/“ **Sweep Setup**” >“ **Power Range**”
Delay	
Sweep Mode	**STEPPED** **SWEPT**
Time	“ **AUTO**” 0

Copy	
Paste	
Insert	
Delete	

Undo	

Cut	
Copy	
Paste	
Delete	
Select All	

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”
3. “ **Sweep Type**”>“ **Segment**”

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”
3. “ **Segment Display**”
4.

Freq Base	X
Order Base	X

CSV

CSV

1. “ **Sweep Setup**”
2. “ **Edit Segment Table**”>“ **Export to CSV File**” CSV
   “Save As” CSV (*****
3. “File Name”“ **Save**”

CSV

E5071C

1. “ **Sweep Setup**”
2. “ **Edit Segment Table**”>“ **Import from CSV File**” CSV
   “Open” CSV (***.csv**
3. CSV “ **Open**”

/ CSV E5071C

- SAW

SAW

-
-

947.5 MHz SAW

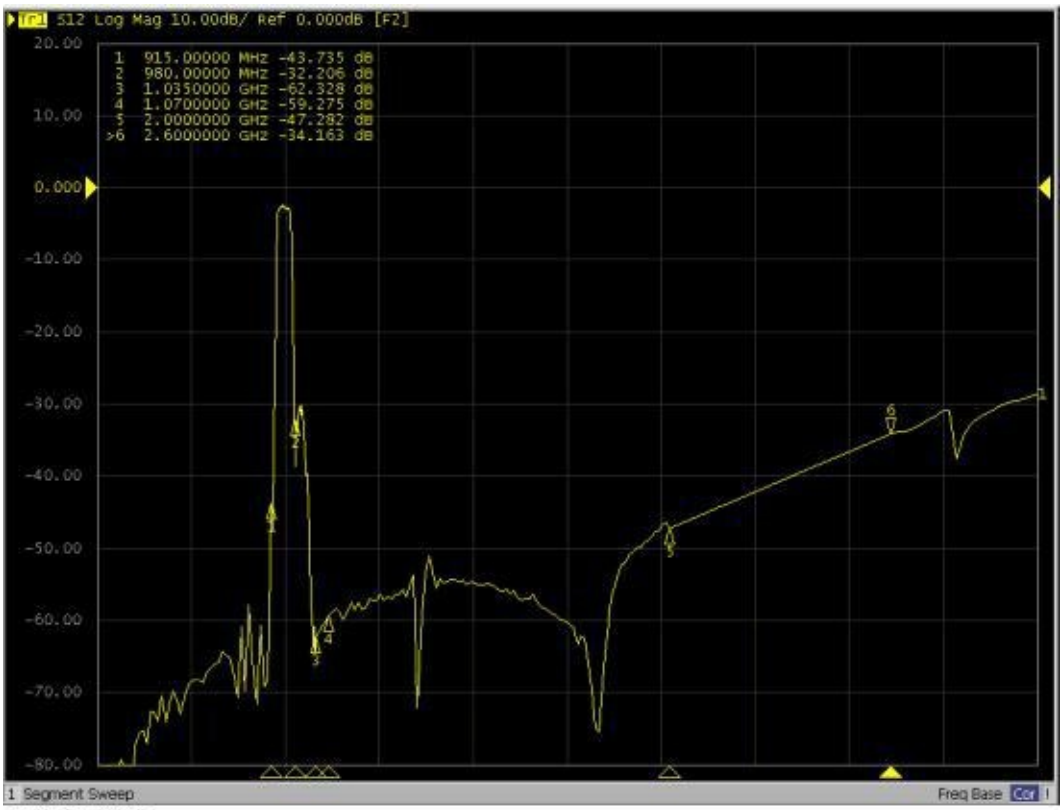
DUT

1.	DUT
2.	E5071C
3.	
4.	DUT 2 ECal
5. DUT	DUT
6.	
7.	

1.

SAW 440 MHz 3 GHz

SAW 440 MHz 3 GHz



e5071c101

			IF
440 MHz	915 MHz 1	47	70 kHz
915 MHz 1	980 MHz 2	130	100 kHz
980 MHz	1.035 GHz	55	70 kHz

2	3		
1.07 GHz 4	2 GHz 5	93	70 kHz
2.6 GHz 6	3 GHz	41	70 kHz

2.

1.

	“**Preset**”>“ **OK**”
	“**Sweep Setup**”>“ **Edit Segment Table**”

2. IF

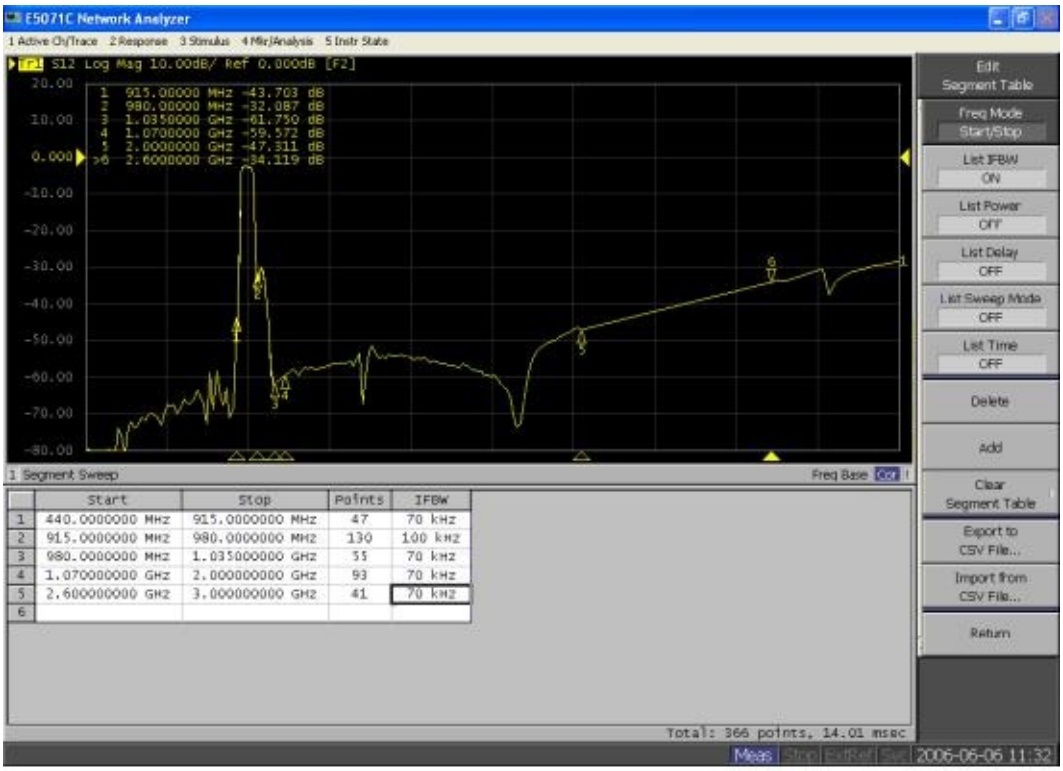
	Focus
IF	“**List IFBW**” IFBW “ **ON**”

3.

	Focus
1	
440 MHz	**4** > **4** > **0** > **M/μ**
915 MHz	**9** > **1** > **5** > **M/μ**
47	**4** > **7** > **x1**

IF 70 kHz	**7** > **0** > **k/m**
2	
915 MHz	**9** > **1** > **5** > **M/µ**
980 MHz	**9** > **8** > **0** > **M/µ**
130	**1** > **3** > **0** > **x1**
IF 100 kHz	**1** > **0** > **0** > **k/m**
3	
980 MHz	**9** > **8** > **0** > **M/µ**
1.035 GHz	**1** > **.** > **0** > **3** > **5** > **G/n**
55	**5** > **5** > **x1**
IF 70 kHz	**7** > **0** > **k/m**
4	
1.07 GHz	**1** > **.** > **0** > **7** > **G/n**
2 GHz	**2** > **G/n**
93	**9** > **3** > **x1**
IF 70 kHz	**7** > **0** > **k/m**
5	
2.6 GHz	**2** > **.** > **6** > **G/n**

3 GHz	**3** > **G/n**
41	**4** > **1** > **x1**
IF 70 kHz	**7** > **0** > **k/m**



e5071c102

3.

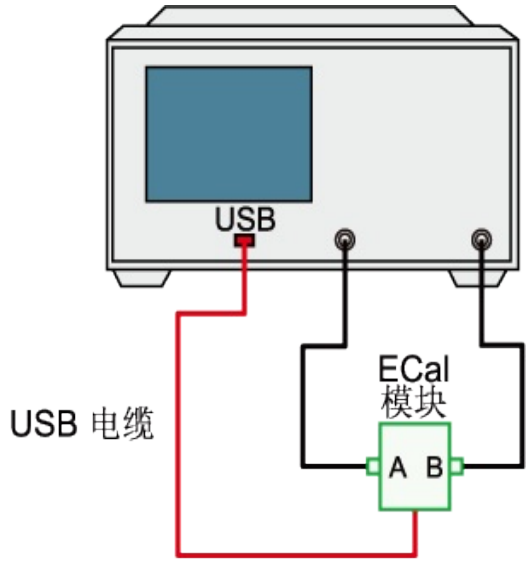
	“**Sweep Setup**”>“ **Sweep Type**”>“ **Segment**”

4.

2 ECal

1. 1 2 ECal

ECal



e5071c322

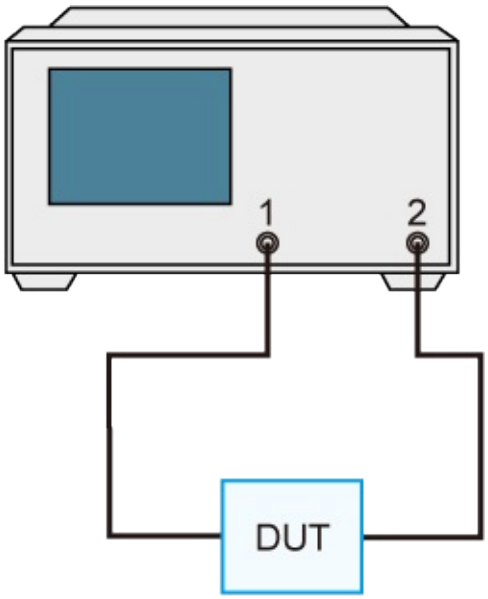
2. 2 ECal

1 2 2 ECal	“**Cal**”>“ **ECal**”>“ **2 Port ECal**”2 >“ **Port 1-2**” 1-2

5. DUT

1 2 DUT

DUT



e5071c320

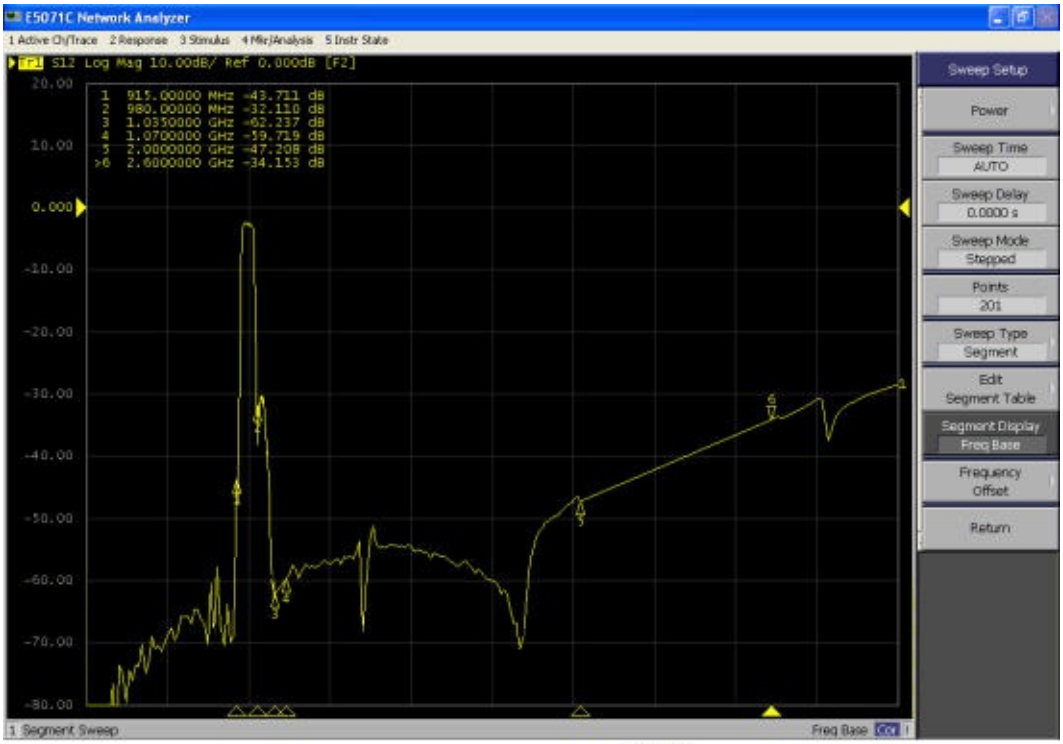
6.

	“**Trigger**”>“ **Single**” “ **Continuous**”

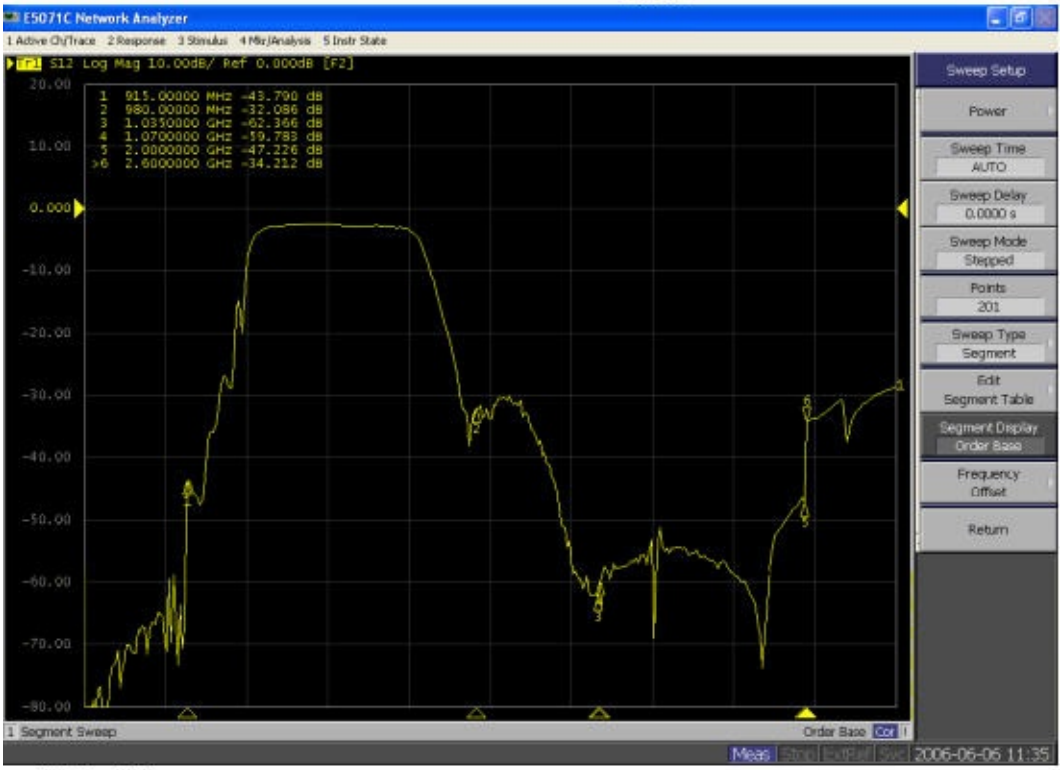
7.

	“Sweep Setup”>“ Segment Display”>“ Frequency Base”“ Order Base”

/



分段显示 —— 按频率 / 按顺序



e5071c103

-
-

1.88 GHz1.96 GHz

DUT

1.	DUT
2.	E5071C
3.	
4.	2 ECal 3
5. DUT	DUT
6.	
7.	
8.	
9.	
10.	
11.	

1.

1.73 GHz	1.83 GHz	50
1.83 GHz	2.03 GHz	400
2.03 GHz	2.13 GHz	50
3.65 GHz	4.03 GHz	38
5.5 GHz	6.02 GHz	52

2.

1.

	“**Preset**”>“ **OK**”
	“**Sweep Setup**”>“ **Edit Segment Table**”

2.

IF

- IF 70 kHz“ **Avg**”>“ **IF Bandwidth**”IF
- 0 dBm“ **Sweep Setup**”>“ **Power**”
- 0 sec“ **Sweep Setup**”>“ **Sweep Delay**”
- “Stepped mode”“ **Sweep Setup**”>“ **Sweep Mode**”
- “Automatic”“ **Sweep Setup**”>“ **Sweep Time**”

	start	stop	points
1	1.730000000 GHz	1.830000000 GHz	50
2	1.830000000 GHz	2.030000000 GHz	400
3	2.030000000 GHz	2.130000000 GHz	50
4	3.650000000 GHz	4.030000000 GHz	38
5	5.500000000 GHz	6.020000000 GHz	52
6			

e5071c083

3.

	“Focus”>“ Return”>“ Sweep Type” >“Segment” “ Sweep Setup”>“ Sweep Type”>“ Segment”

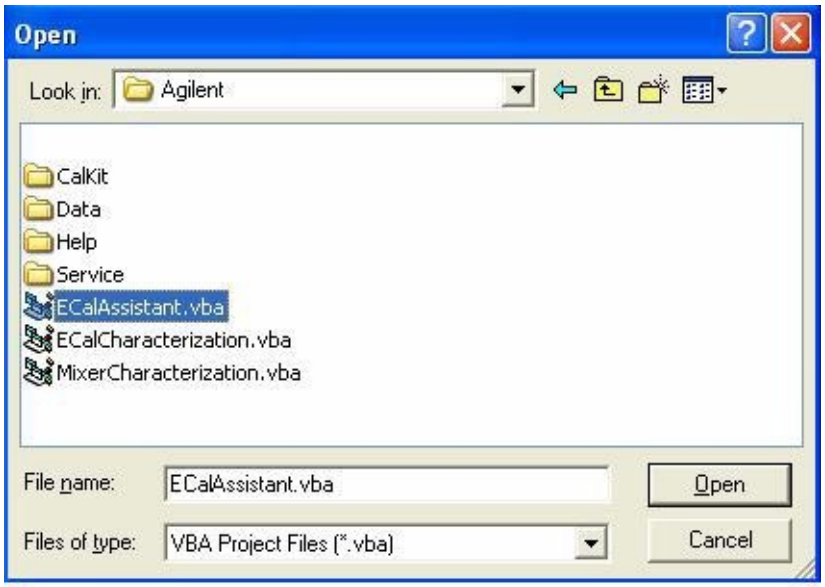
4.

E5071C 2 ECal 3/4

1. USB 2 ECal USB E5071C USB
2. 3/4 ECal

“VBA Project Open” VBA	“**Macro Setup**”>“ **Load Project**”
ECalAssistant.VBA	**D:\Agilent\ECalAssistant.VBA** “ **Open**”
	“Macro Run”

1. “EcalAssistant (start)”EcalAssistant



e5071c084

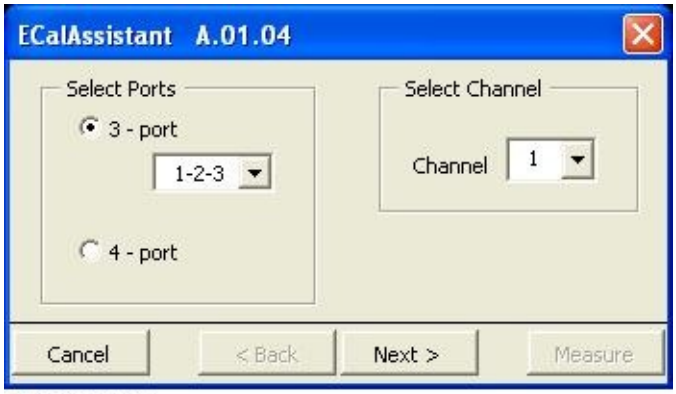
2. “ **Next**”“EcalAssistant (port/channel selection)”EcalAssistant/

e5071c085

3. ECal

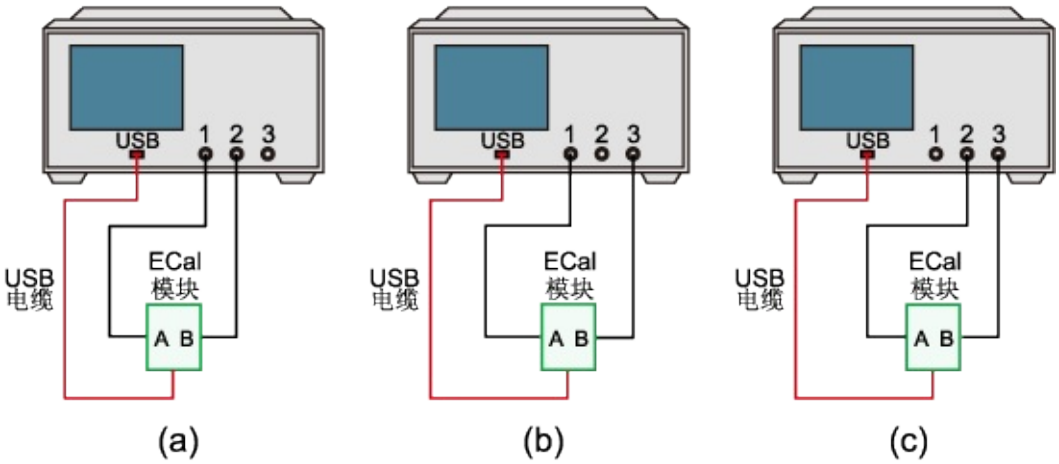
ECal 3	**3 Port3**
ECal 12 3	**123**
ECal 1	**Channel 1**

4. “ **Next**”“EcalAssistant (connection)”EcalAssistant “3-port”3 /“Channel”1



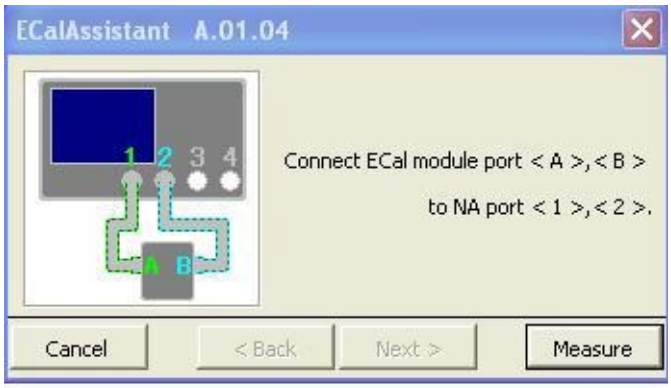
e5071c086

ECal



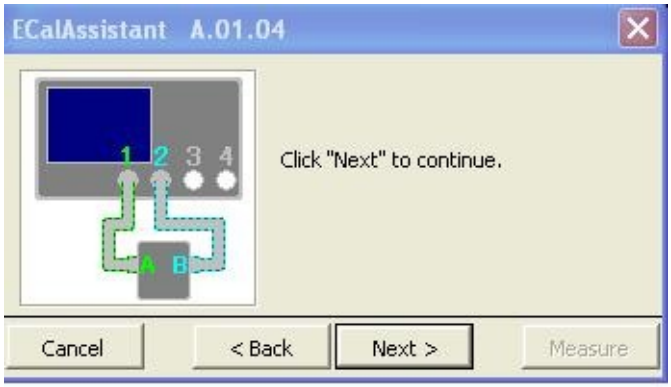
e5071c361

5. “ **Next**” 1-2



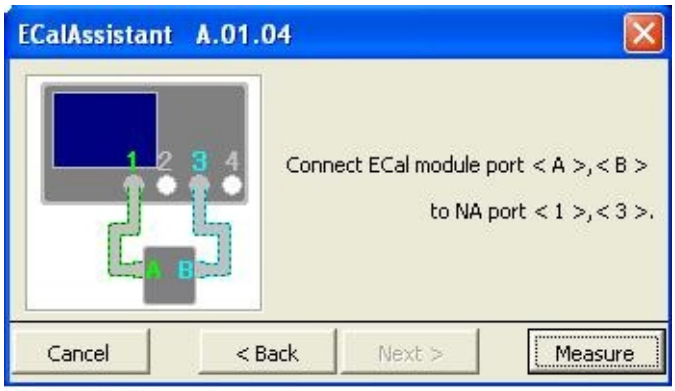
e5071c087

6. “ **Measure**” 1-2 1-2



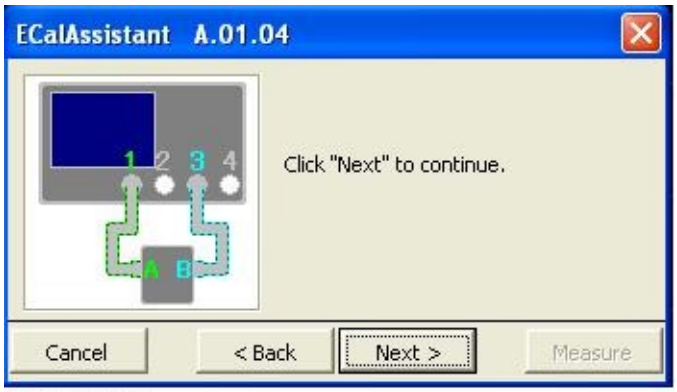
e5071c088

7. “ **Next**” 1-3



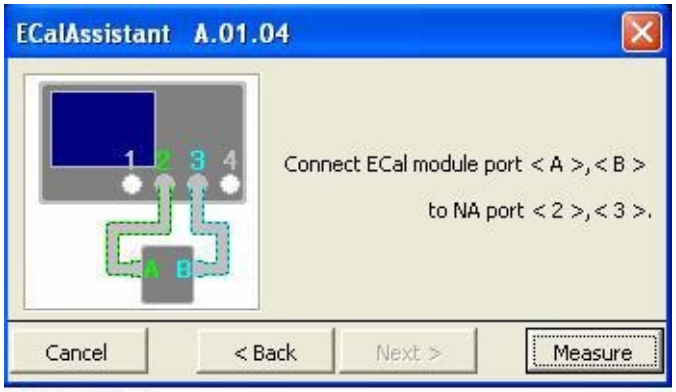
e5071c089

8. “ **Measure**” 1-3 1-3



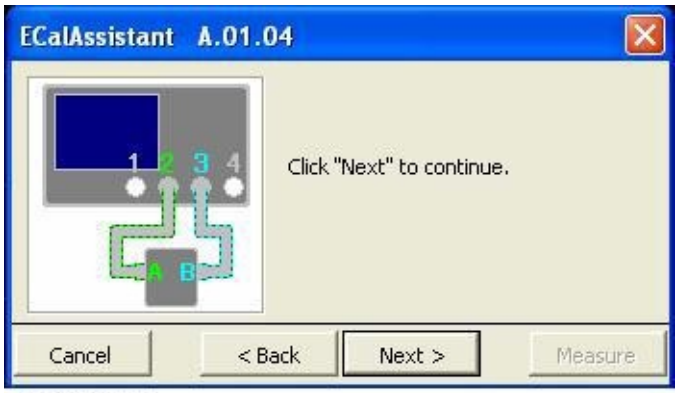
e5071c090

9. “ **Next**” 2-3



e5071c100

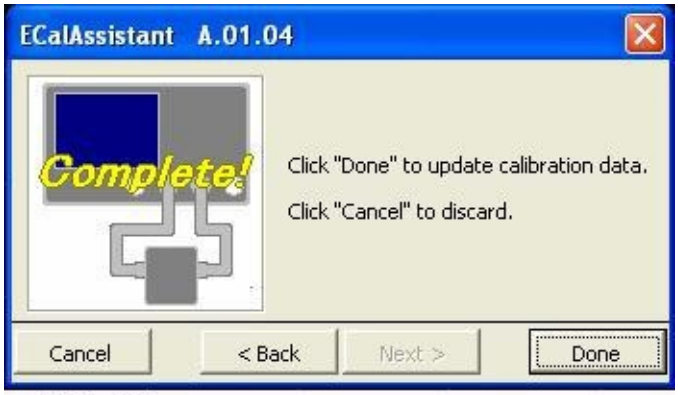
10. “ **Measure**” 2-3 2-3



e5071c098

11. “EcalAssistant (complete)”EcalAssistant“ **Done**”

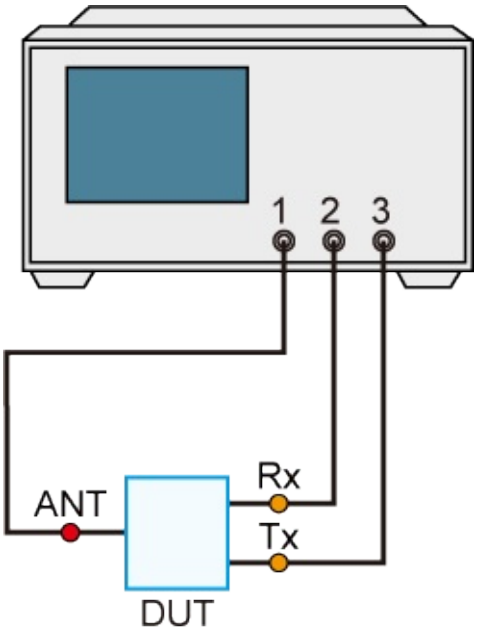
“ECalAssistant (complete)”ECalAssistant



e5071c099

5. DUT

DUT 12 3



e5071c359

6.

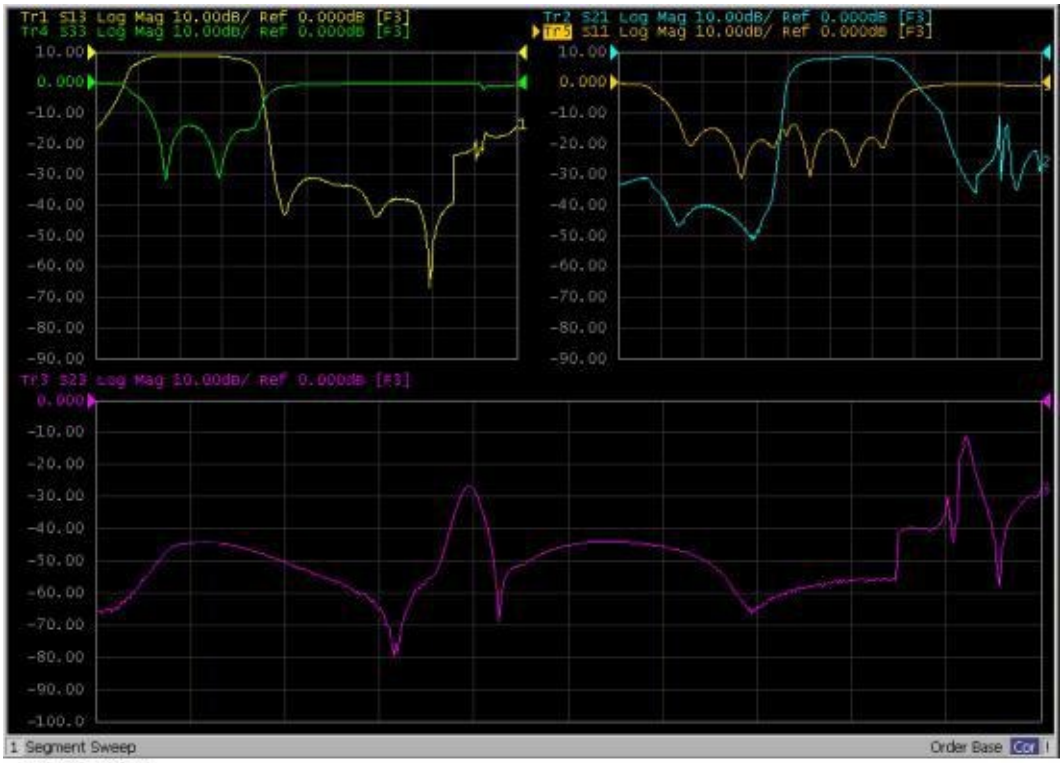
5	“**Display**”>“ **Number of Traces**”> **5**
	“**Allocate Traces**”>
1S13	“**Meas**”-“ **S13**”
2S21	“**Trace Next**”>“ **Meas**”>“ **S21**”
3S23	“**Trace Next**”>“ **Meas**”>“ **S23**”
4S33	“**Trace Next**”>“ **Meas**”>“ **S33**”
5S11	“**Trace Next**” >“**Meas**”>“ **S11**”

7.

	“**Trigger**”>“ **Single**”“ **Continuous**”

8.

	“Sweep Setup”>“ Segment Display”>“ Order Base”
1 10	“Trace Next”>“ Scale”>“ Reference Position” > 1 > 0 > x1
2 10	“Trace Next”>“ Reference Position”> 1 > 0 > x1
3 10	“Trace Next”>“ Reference Position”> 1 > 0 > x1
4 9	“Trace Next”>“ Reference Position”> 9 > x1
5 9	“Trace Next”>“ Reference Position”> 9 > x1


Tr1 S13 Log Mag 10.00dB/ Ref 0.000dB [F3]
Tr4 S33 Log Mag 10.00dB/ Ref 0.000dB [F3]
Tr2 S21 Log Mag 10.00dB/ Ref 0.000dB [F3]
Tr5 S11 Log Mag 10.00dB/ Ref 0.000dB [F3]
Tr3 S23 Log Mag 10.00dB/ Ref 0.000dB [F3]
1 Segment Sweep
Order Base

e5071c091

9.

1. 3-dB

	“**Marker Fctn**”>“ **Couple**”“ **OFF**”
1	**Trace Next**
1	**Marker**
/	“**Marker Search**”>“ **Tracking**”“ **ON**”
1	**Max**
	“**Bandwidth**”“ **ON**”

“ **loss**” -1.243 dB3-dB (**BW**) 85.53 MHz

2. 3-dB

2	**Trace Next**
1	**Marker**
/	“**Marker Search**”>“ **Tracking**”“ **ON**”
1	**Max**

	“**Bandwidth**”“ **ON**”

“ **loss**” -1.627 dB3-dB (**BW**) 71.04 MHz

3.

3	**Trace Next**
1	**Marker**
/	“**Marker Search**”>“ **Tracking**” “ **ON**”
1 1.92 GHz	“**Peak**”-“ **Search Left**”“ **Search Right**”

1 -6.612 dB

4.

4	**Trace Next**
1	**Marker**
/	“**Marker Search**”>“ **Tracking**”“ **ON**”
1	“**Peak**”-“ **Search Left**”“ **Search Right**”

1 12.65 dB

5.

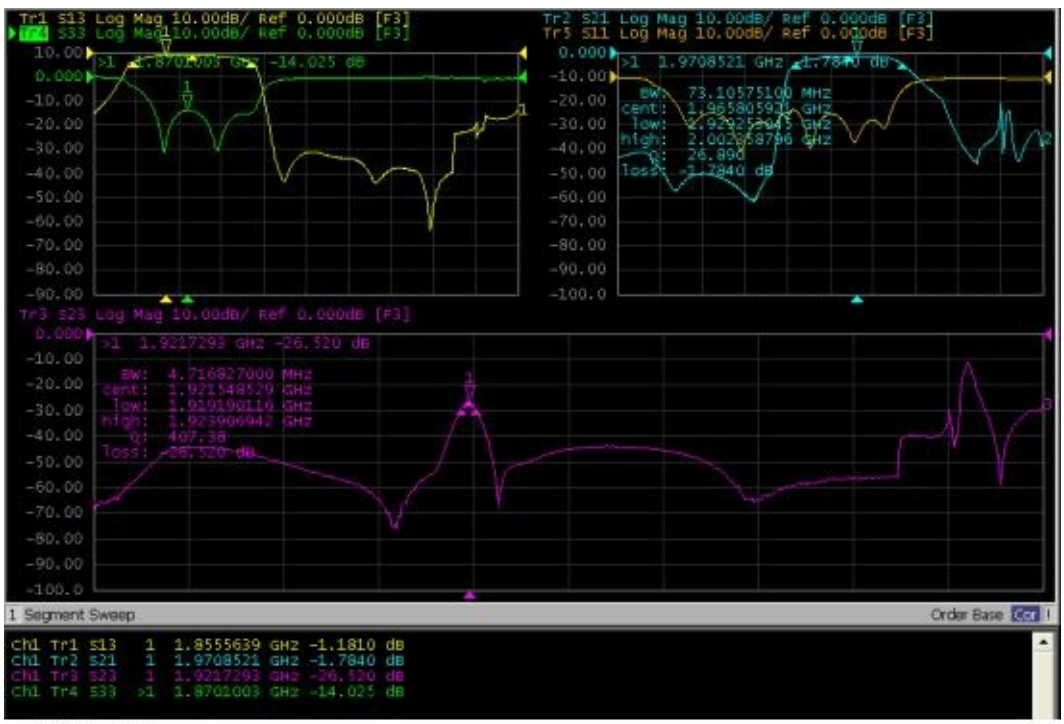
5	**Trace Next**

1	**Marker**
/	“**Marker Search**”>“ **Tracking**”“ **ON**”
1	“**Peak**”>“ **Search Left**”“ **Search Right**”

1 13.80 dB

6.

	“**Marker Fctn**”>“ **Marker Table**”“ **ON**”



e5071c092

10.

1. 1 (S13)

1	**Trace Next**
	“**Analysis**”>“ **Limit Test**”>“ **Edit Limit Line**”

2. 1

1

	Type	Begin Stimulus	End Stimulus	Begin Response	End Response
1	MAX	1.730000000 GHz	1.940000000 GHz	0 dB	0 dB
2	MIN	1.850000000 GHz	1.910000000 GHz	-4 dB	-4 dB
3	MAX	1.930000000 GHz	1.990000000 GHz	-35 dB	-35 dB
4	MAX	1.990000000 GHz	2.130000000 GHz	-30 dB	-30 dB
5	MAX	2.130000000 GHz	6.020000000 GHz	-20 dB	-20 dB
6					

e5071c093

3. 2 (S21)

2	**Trace Next**

4. 2

2

	Type	Begin Stimulus	End Stimulus	Begin Response	End Response
1	MAX	1.730000000 GHz	1.850000000 GHz	-40 dB	-40 dB
2	MAX	1.850000000 GHz	1.910000000 GHz	-45 dB	-45 dB
3	MAX	1.910000000 GHz	6.020000000 GHz	0 dB	0 dB
4	MIN	1.930000000 GHz	1.990000000 GHz	-5 dB	-5 dB
5					

e5071c094

11.

1. 1

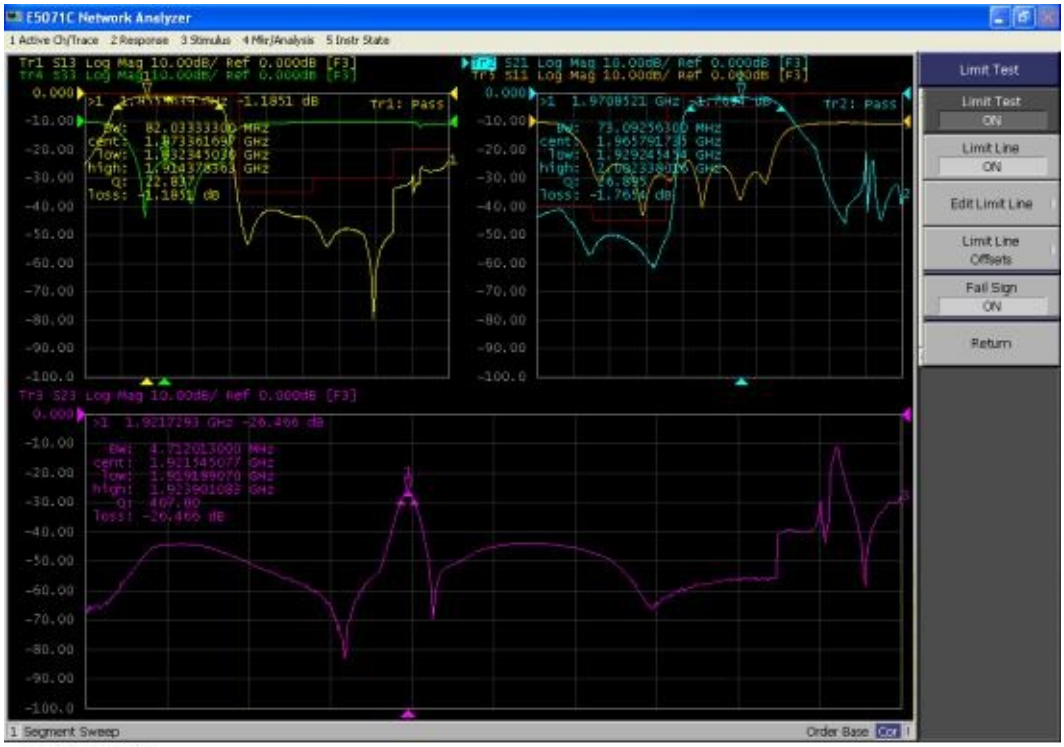
1	**Trace Prev**
	“**Analysis**”>“ **Limit Test**”-“ **Limit Line**” “ **ON**”
	“**Limit Test**”“ **ON**”

2. 2

2	**Trace Next**
	“**Limit Line**”“ **ON**”
	“**Limit Test**”“ **ON**”

3.

	“**Trigger**”>“ **Single**”“ **Continuous**”

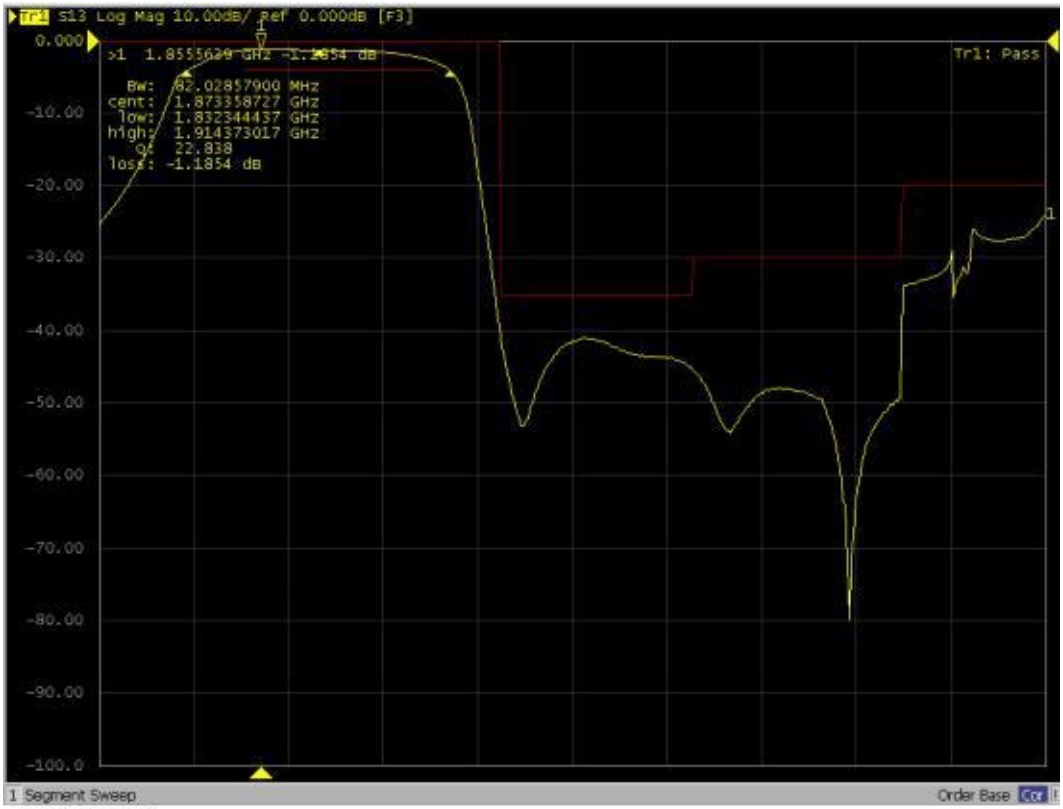


e5071c095

4. 1

1	**Trace Prev**
1	**Trace Max**

1

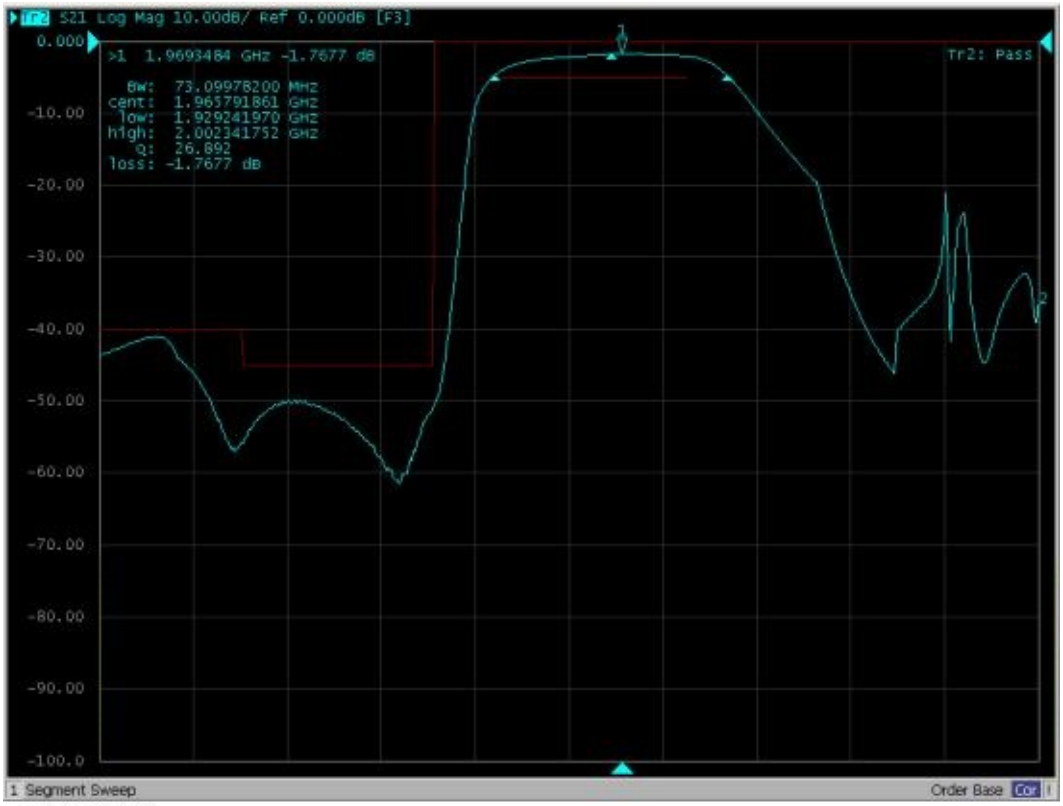


e5071c096

5. 2

2	“**Trace Prev**” 2

2


Tr2 S21 Log Mag 10.00dB/ Ref 0.000dB [F3]
>1 1.9693484 GHz -1.7677 dB
BW: 73.09978200 MHz
cent: 1.965791861 GHz
low: 1.929241970 GHz
high: 2.002341752 GHz
Q: 26.892
loss: -1.7677 dB
Tr2: Pass
0.000
-10.00
-20.00
-30.00
-40.00
-50.00
-60.00
-70.00
-80.00
-90.00
-100.0
1 Segment Sweep
Order Base

e5071c097

-
-

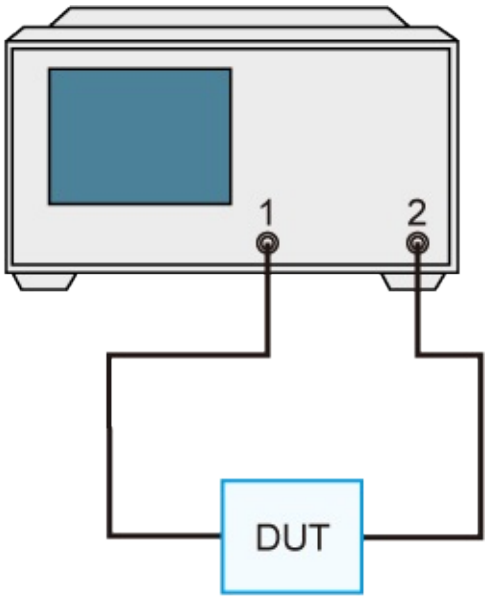
1.09-GHz

DUT

1. DUT	DUT
2.	
3.	
4. DUT	DUT
5.	
6.	-

1. DUT

DUT



e5071c320

2.

	“**Preset**”>“ **OK**”
1.09 GHz	“**Center**”> **1** > **.**> **0** > **9** > **G/n**
20 MHz	“**Span**”> **2** > **0** > **M/u**
S21	“**Meas**”>“ **S21**”
	“**Format**”>“ **Expand Phase**”
	“**Scale**”>“ **Auto Scale**”

3.

	DUT“ **Cal**”>“ **Calibrate**”>“ **Response (Thru)**”>“ **Thru**”>“ **Done**”

4. DUT

DUT DUT

	“**Scale**”>“ **Auto Scale**”

e5071c113

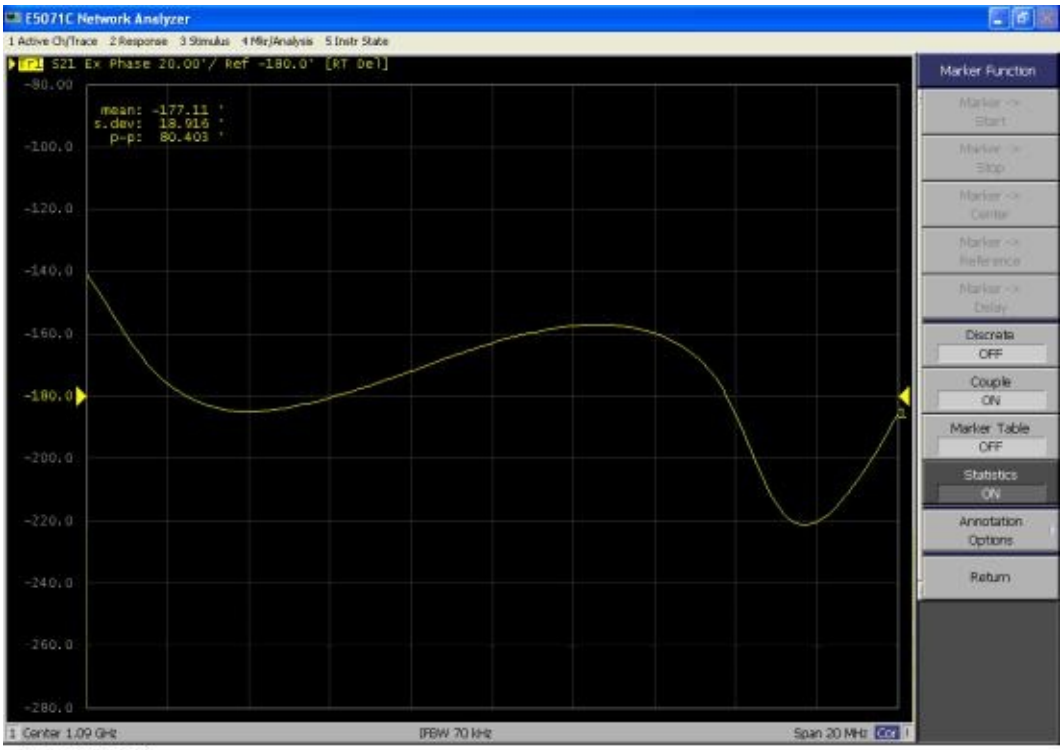
5.

	“Scale”>“ Electrical Delay”>“ Up Arrow”/“ Down Arrow”

6.

-

	“Scale”>“ Auto Scale”
	“Marker Function”>“ Statistics”“ ON”

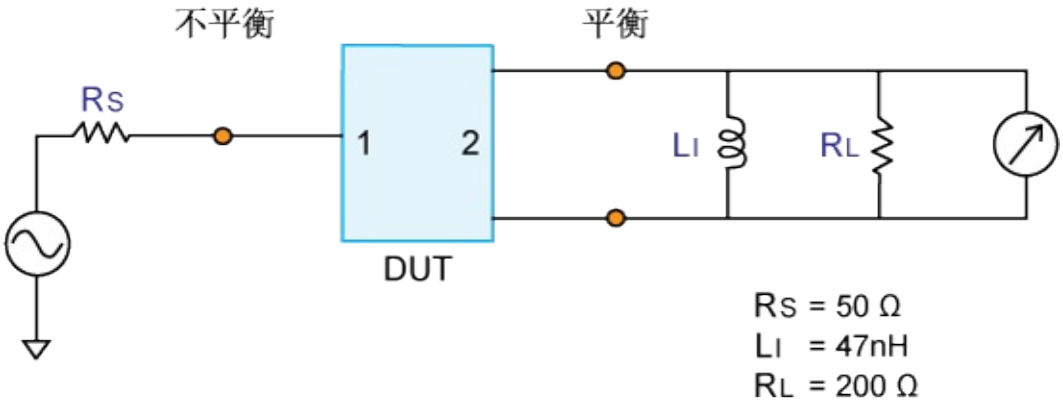


e5071c114

-

-
-

942.5 MHz SAW DUT



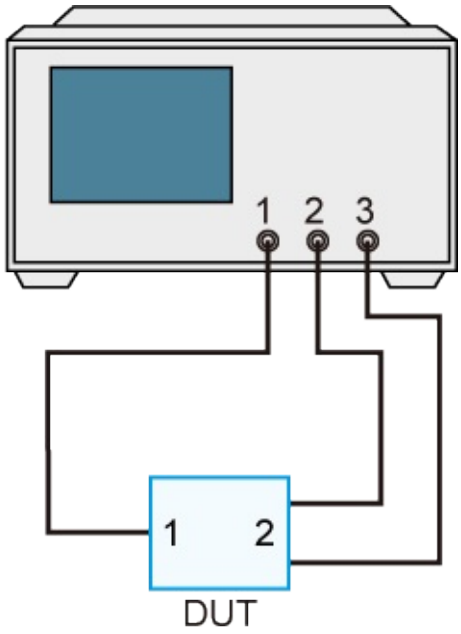
e5071c378

DUT

1. DUT	DUT
2.	
3.	3
4.	
5.	S
6.	
7.	
8.	

1. DUT

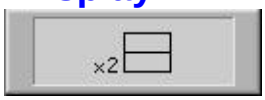
DUT E5071C

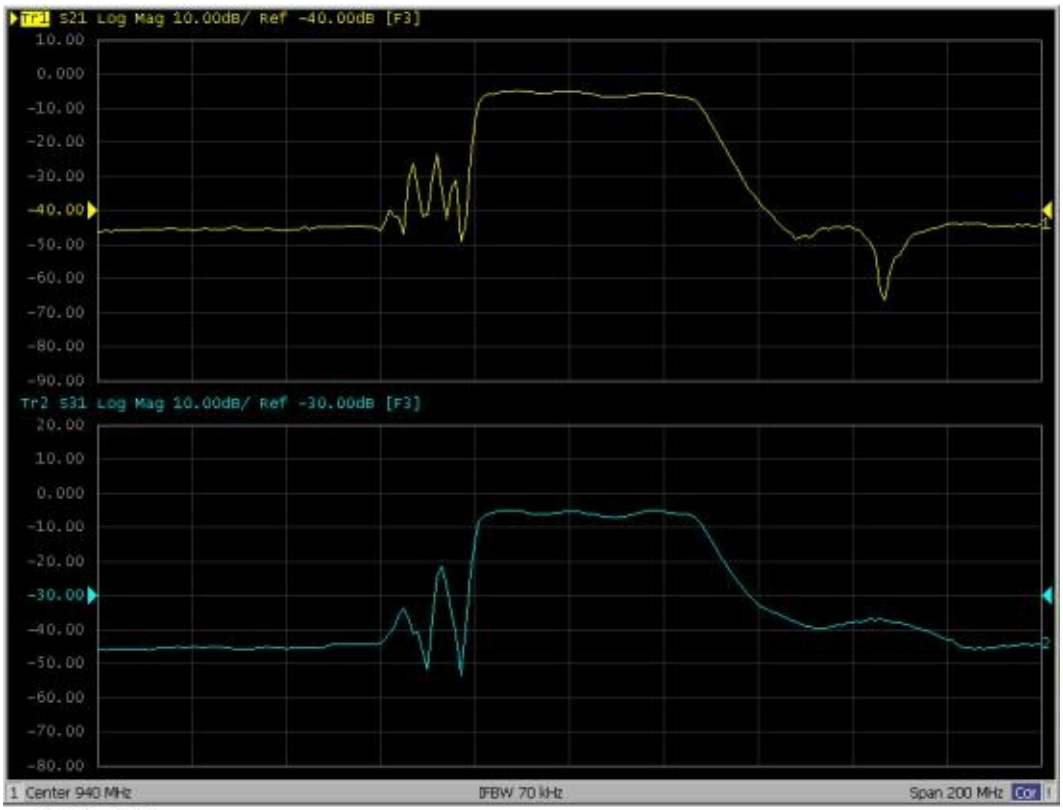


e5071c358

2.

-

	“**Preset**”>“ **OK**”
940 MHz	“**Center**”> **9** > **4** > **0** > **M/u**
200 MHz	“**Span**”> **2** > **0** > **0** > **M/u**
2	“**Display**”>“ **Num of Traces**”> **2**
1 S21	“**Meas**”>“ **S21**”
2 S31	“**Trace Next**”>“ **Meas**”>“ **S31**”
	“**Display**”>“ **Allocate Traces**”> 
	“**Scale**”>“ **Auto Scale All**”


Tr1 S21 Log Mag 10.00dB/ Ref -40.00dB [F3]
10.00
0.000
-10.00
-20.00
-30.00
-40.00
-50.00
-60.00
-70.00
-80.00
-90.00
Tr2 S31 Log Mag 10.00dB/ Ref -30.00dB [F3]
20.00
10.00
0.000
-10.00
-20.00
-30.00
-40.00
-50.00
-60.00
-70.00
-80.00
1 Center 940 MHz
IFBW 70 kHz
Span 200 MHz
Cor

e5071c104

3.

1.

85033D	“**Cal**”>“ **Cal Kit**”>“ **85033D**”
	“**Calibrate**”>“ **3-Port Cal**”3
123	“**Select Ports**”>“ **1-2-3**”

2.

	Reflection
1	“ **Port 1 OPEN**” 1 “ **Port 1 SHORT**” 1 “ **Port 1 LOAD**” 1
2	“ **Port 2 OPEN**” 2 “ **Port 2 SHORT**” 2 “ **Port 2 LOAD**” 2

3	“ **Port 3 OPEN**” 3 “ **Port 3 SHORT**” 3 “ **Port 3 LOAD**” 3

3.

	“**Return**” >“**Reflection**”
1 2	“ **Port 1-2 Thru**” 1-2
1 3	“ **Port 1-3 Thru**” 1-3
2 3	“ **Port 2-3 Thru**” 2-3

4.

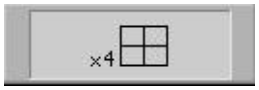
	“**Return**”>“ **Done**” “ **Correction**” “ **ON**”
	“**Return**”>“ **Return**” >“ **Property**” “ **ON**”

4.

DUT 1 DUT 2	“**Analysis**”>“ **Fixture Simulator” >“Topology”> “Device” > “SE-Bal- ”**
DUT 1 1	“**Port 1 (se)**” 1 >“**1**”
DUT 2 2 3	“**Port 2 (bal)**” 2>“ **2-3**”

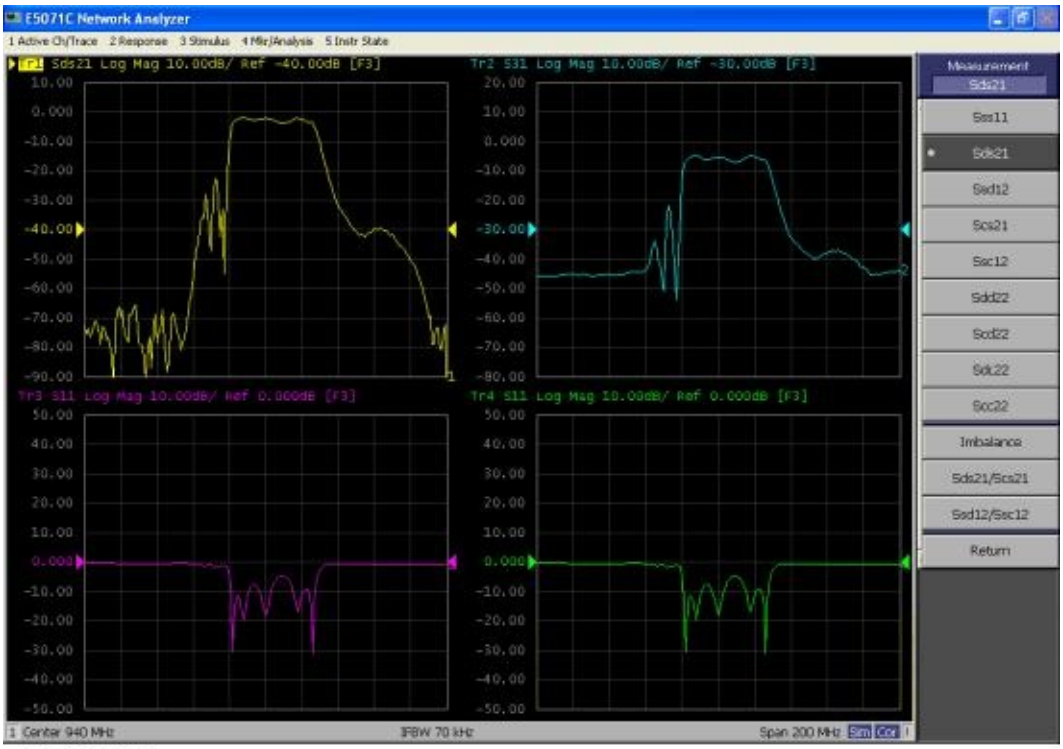
5.

1.

4	“**Display**”>“ **Number of Traces**”> **4**
4	“**Allocate Traces**”> 

2. 1 S

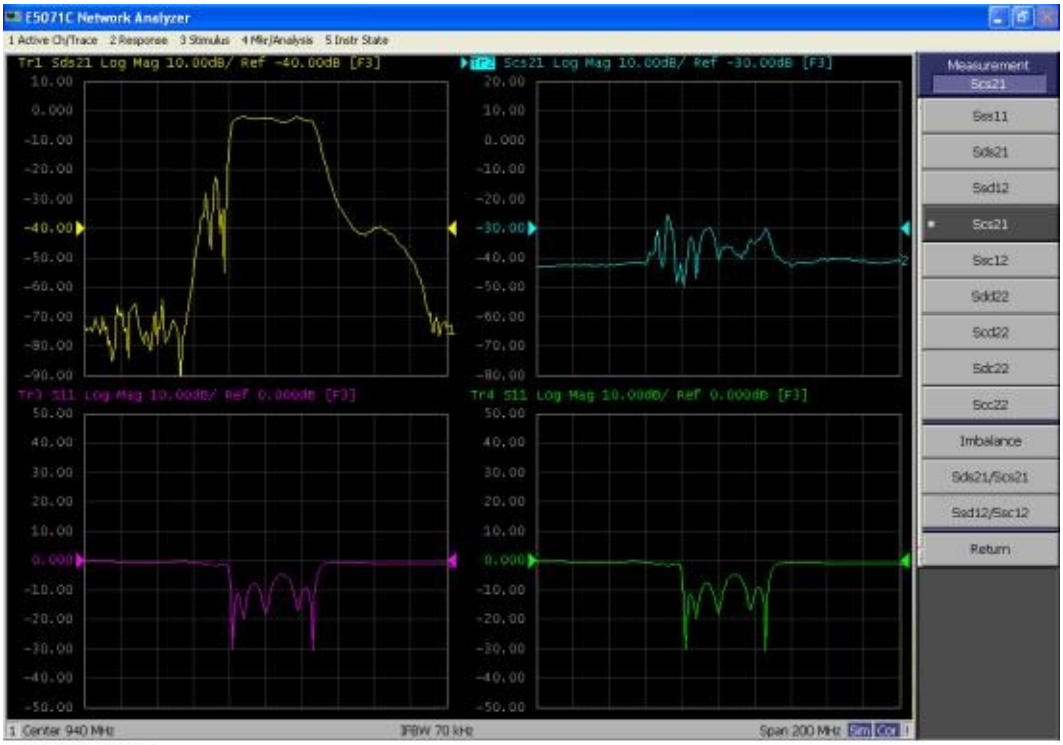
	“**Analysis**”>“ **Fixture Simulator**” >“**Fixture Simulator**” “ **ON**”
1 -	“**BalUn**” - “ **ON**”
Sds21	“**Meas**”>“ **Sds21**”



e5071c105

3. 2 S

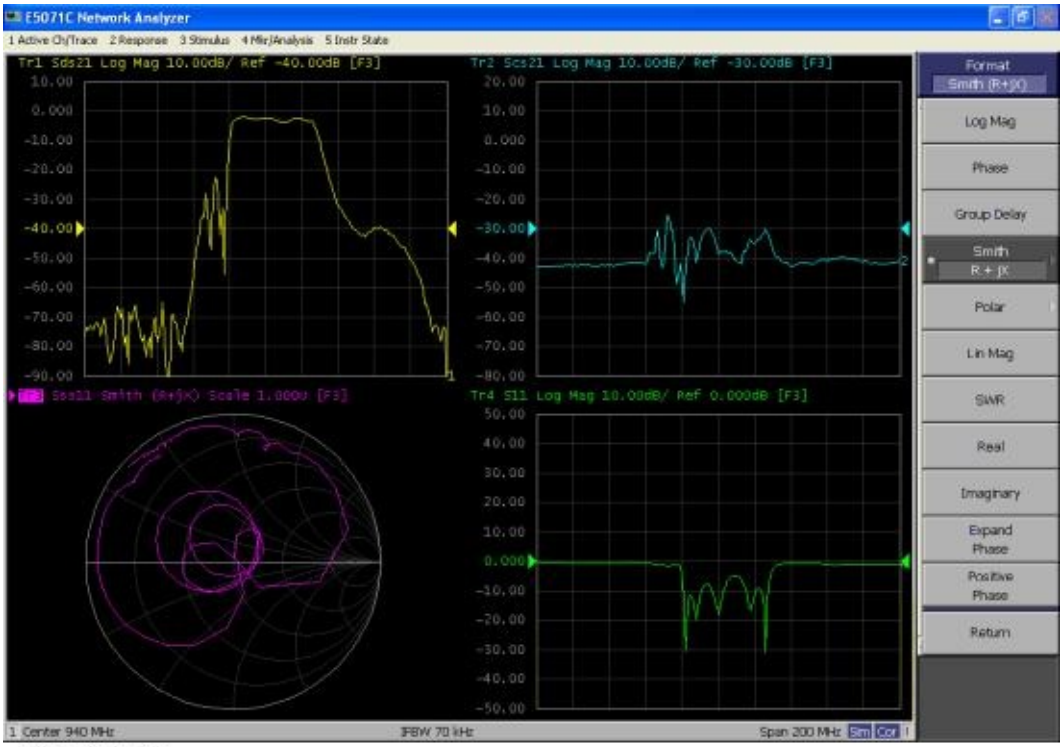
2 -	"**Trace next**" >"**Analysis**">" **Fixture Simulator**">" **BalUn**" - " **ON**"
Scs21	"**Meas**">" **Scs21**"



e5071c106

4. 3 S

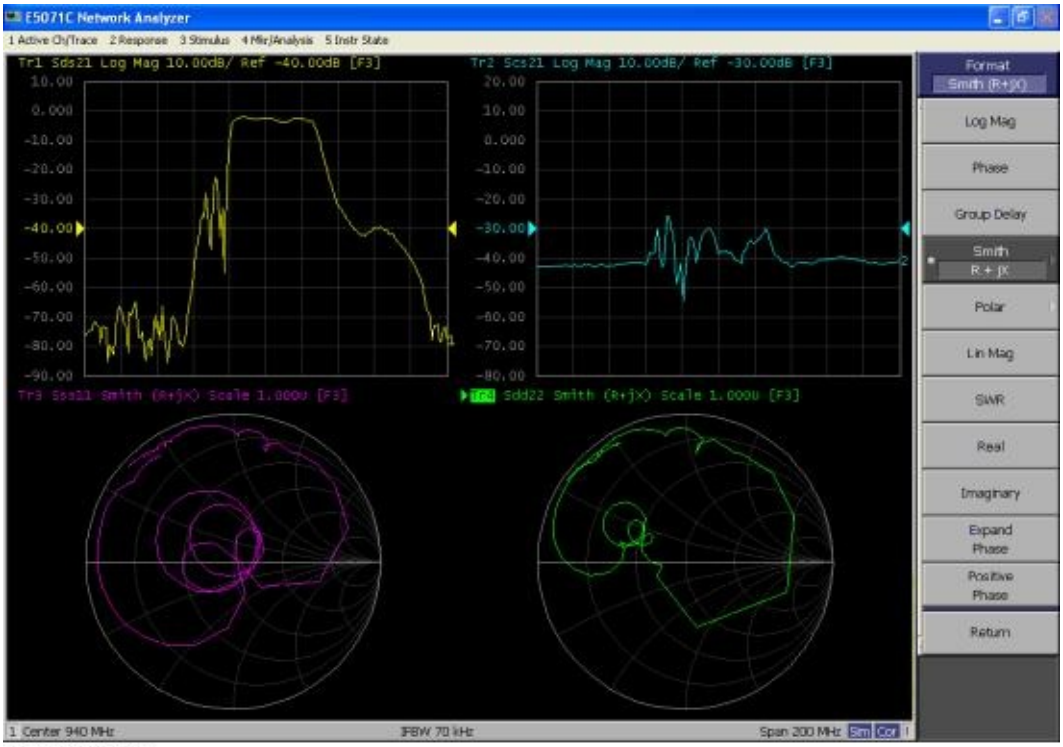
3 -	“**Trace next**”>“ **Analysis**”>“ **Fixture Simulator**”>“ **BalUn**” - “ **ON**”
Sss11	“**Meas**”>“ **Sss11**”
R+jX	“**Format**”>“ **Smith**”>“ **R + jX**”



e5071c107

5. 4 S

4 -	"**Trace next**" >"**Analysis**">" **Fixture Simulator**">" **BalUn**" - " **ON**"
Sdd22	"**Meas**">" **Sdd22**"
R+jX	"**Format**">" **Smith**" >"**R + jX**"

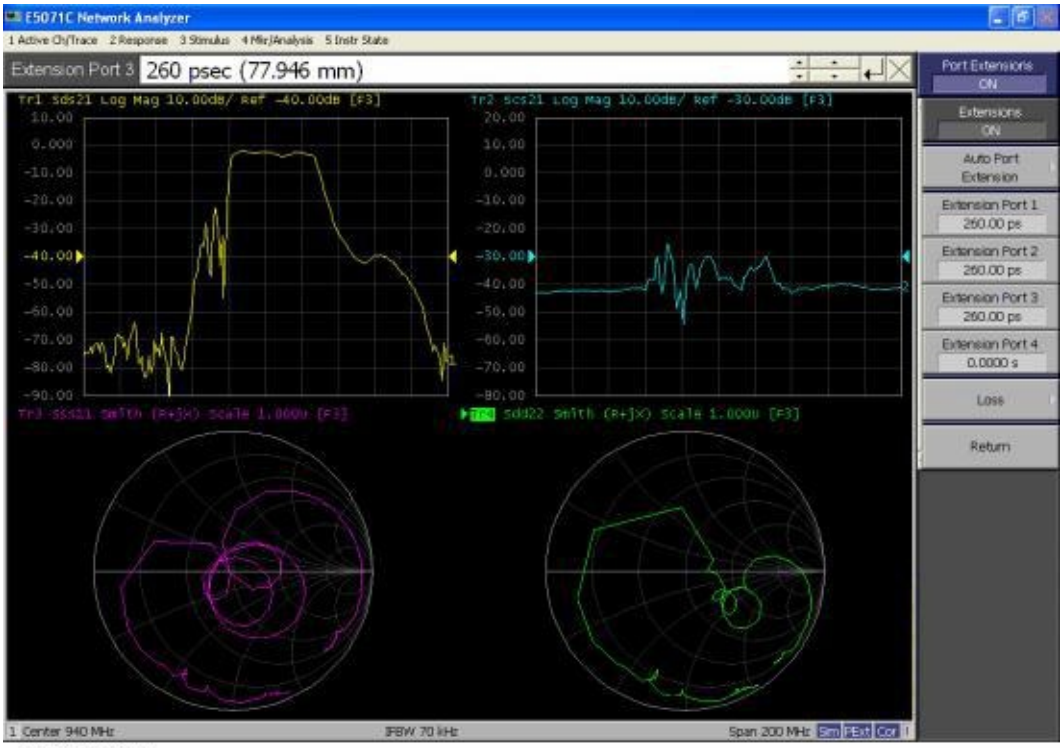

E5071C Network Analyzer
1 Active Ch/Trace 2 Response 3 Stimulus 4 Mkr/Analysis 5 Instr State
Tr1 Sds21 Log Mag 10.00dB/ Ref -40.00dB [F3]
Tr2 Scs21 Log Mag 10.00dB/ Ref -30.00dB [F3]
Tr3 Sss11 Smith (R+jX) Scale 1.000U [F3]
Tr4 Sdd22 Smith (R+jX) Scale 1.000U [F3]
Format
Smith (R+jX)
Log Mag
Phase
Group Delay
Smith
R + jX
Polar
Lin Mag
SWR
Real
Imaginary
Expand Phase
Positive Phase
Return
1 Center 940 MHz
IFBW 70 kHz
Span 200 MHz

e5071c108

6.

DUT Touchstone

1 260 ps	“**Cal**”>“ **Port Extensions**”>“ **Extension Port 1**” 1> **.**> **2** > **6** > **G/n**
2 260 ps	“**Extension Port 2**” 2> **.**> **2** > **6** > **G/n**
3 260 ps	“**Extension Port 3**” 3> **.**> **2** > **6** > **G/n**
	“**Extensions**” “ **ON**”



e5071c109

7.

Z0 Z0/2 2Z0

1. DUT 1 50 Ω

1 50 Ω	"**Analysis**">" **Fixture Simulator**" >"**Port Z conversion**" Z >" **Port 1 Z0 Real**" 1 Z0 > **5** > **0** > **x1**

2. DUT 2 200 Ω 100 Ω

2 100 Ω	"**Port 2 Z0 Real**" 2 Z0 > **1** > **0** > **0** > **x1**
3 100 Ω	"**Port 3 Z0 Real**" 3 Z0 > **1** > **0** > **0** > **x1**

3.

4.

	"**Port Z Conversion**" Z " **ON**"

DUT 2 50 Ω

8.

DUT 2 47 nH -

Shunt L - Shunt C L - C	“**Return**”“ **Analysis**” >“**Fixture Simulator**”>“ **Diff. Matching**” >“**Select Circuit**”>“ **Shunt L-Shunt C**” L - C e5071c110
47 nH	“**L**”> **4** > **7** > **G/n**
C=0 G=0R=0	“ **C**”“ **G**”“ **R**”“ **0**”
	“**Diff. Matching**”“ **ON**”

-
-

DUT

1.	
2.	
3. DUT	DUT
4.	
5.	

1.

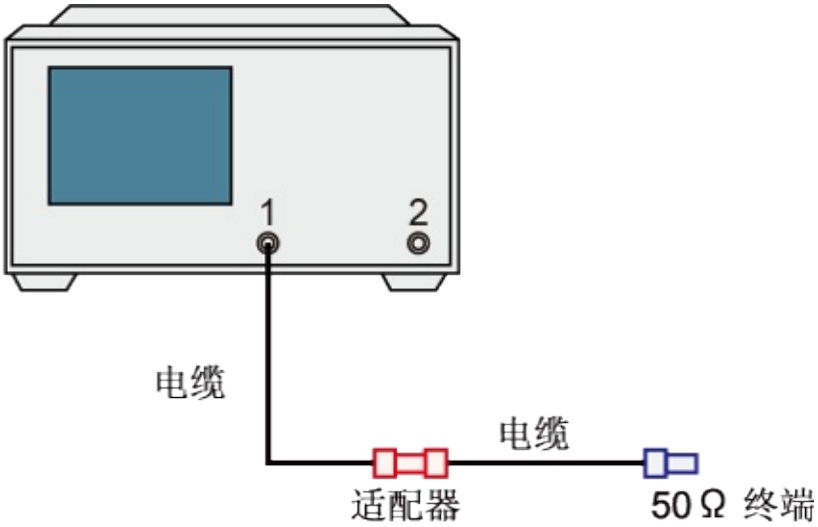
	“**Preset**”>“ **OK**”
3 GHz	“**Start**”> **3** > **G/n**
201	“**Sweep Setup**” >“**Points**”> **2** > **0** > **1** > **x1**
	“**Analysis**” >“**Transform**”>“ **Set Freq Low Pass**”
S11	“**Meas**”>“ **S11**”

2.

1 1 1

3. DUT

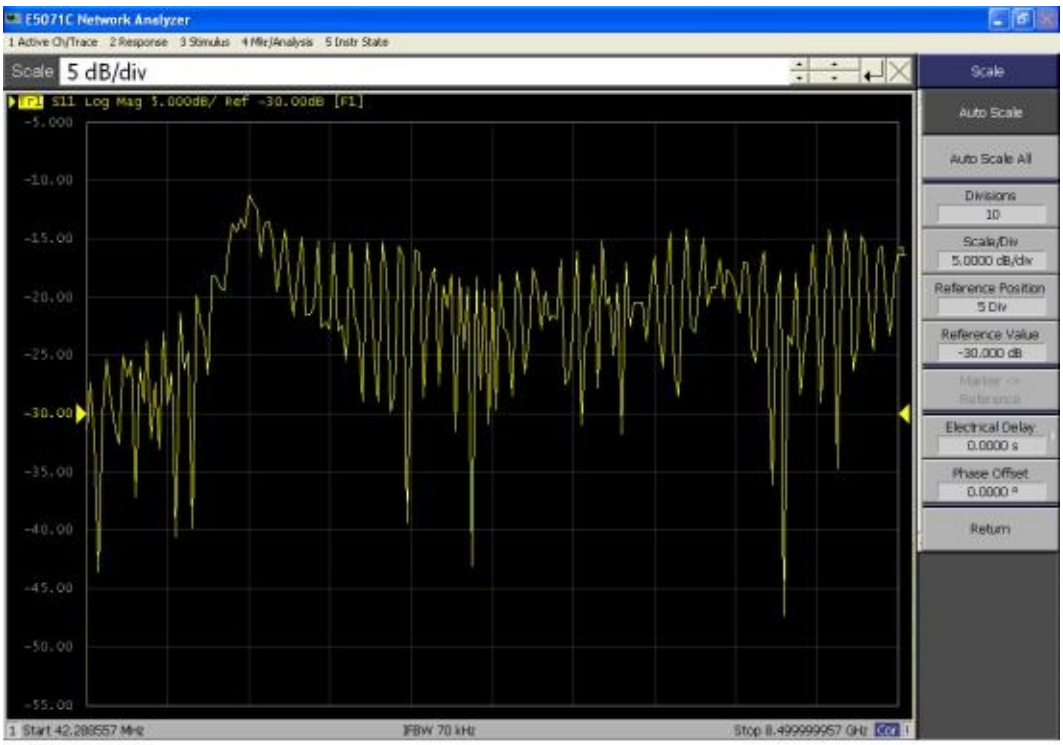
DUT



e5071c327

4.

	“**Scale**”>“ **Auto Scale**”



e5071c111

5.

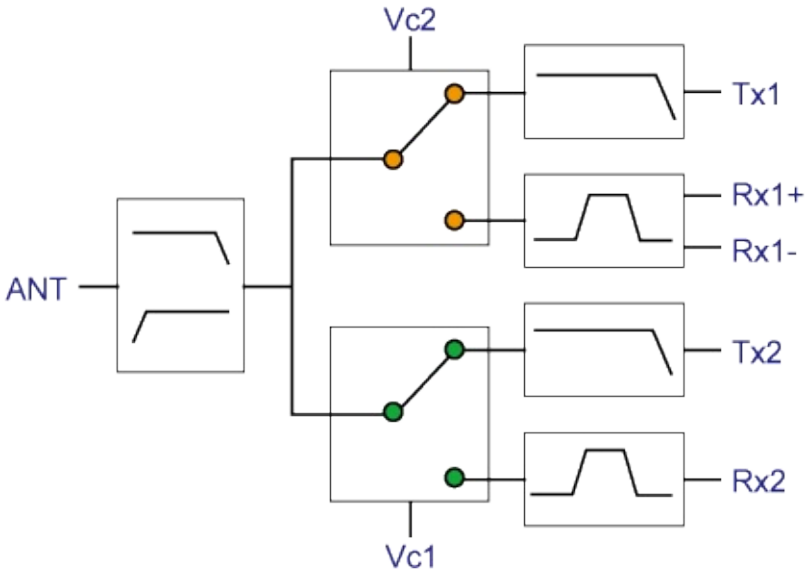
	“Format”>“ Real”
	“Analysis”>“ Transform”>“ Type” >“Lowpass Imp”
	“Window”>“ Maximum”
0 s 20 ns	“Start”> 0 > x1
	“Stop”> 2 > 0 > G/n
	“Transform”“ ON”
	“Scale”>“ Auto Scale”

E5071C Network Analyzer
1 Active Ch/Trace 2 Response 3 Stimulus 4 Mkr/Analysis 5 Instr State
Stop 20 nsec (5.9958 m)
Tr1 S11 Real 10.00mU/ Ref 0.000U [F1]
50.00m
40.00m
30.00m
20.00m
10.00m
0.000
-10.00m
-20.00m
-30.00m
-40.00m
-50.00m
1 Start 0 s
IFBW 70 kHz
Stop 20 ns
Transform
Transform
ON
Start
0.0000 s
Stop
20.000 ns
Center
10.000 ns
Span
20.000 ns
Type
Lowpass Imp
Window
Return

e5071c112

-
-

E5071C E5091A 6



e5071c472

DUT

1.	
2.	
3.	
4.	E5091A
5.	
6.	
7.	
8.	4 ECal
9. DUT	DUT
10.	

1.

1	400 MHz	1.4 GHz	201	1 - A 2 - T1 3 - R1+ 4 - R1-	1 2	S12	2	12
2	880 MHz	1 GHz	101		1 2	Sds21	3	1 34
3	1.34 GHz	2.34 GHz	201	1 - A 2 - T2 3 - R2+ 4 - R1-	1 2	S12	2	12
4	1.665 GHz	2.015 GHz	101		1 2	S31	2	13

2.

2 2

	“**Preset**”>“ **OK**”
	“**Display**”>“ **Allocate Channels**”>

3.

1. E5091A ID1 9

E5091A	“System”>“ **Multiport Test Set Setup**”
ID1 9	**“Test Set 1” 1>“Select Test Set”>“E5091_9”**

2. E5091A

E5091A	**Property**

3. 1 1 4

A 1	“**Port1**” 1>“ **A**”
T1 2	“**Port2**” 2>“ **T1**”
R1+ 3	“**Port3**” 3>“ **R1+**”
R1- 4	“**Port4**” 4>“ **R1-**”

4. 23 4 “ **Channel Next**” 1
5. E5091A

E5091A	“**Control**” “ **ON**”

4.

1. 1

1	**Channel Next**
1	“**Control Lines**”
2	“**Control Lines**”>“ **Line 1**” 1“ **High**”

2. 1 23 4

5.

1. 1

1	**Channel Next**
400 MHz	“**Start**”> **4** > **0** > **0** > **M/u**
1.4 GHz	“**Stop**”> **1** > **.**> **4** > **G/n**
201	“**Sweep Setup**>“ **Point**”> **2** > **0** > **1** > **x1**

2. 1 23 4

6.

2

2	“**Channel Next**” 2
DUT 1 DUT 2	“**Analysis**”>“ **Fixture Simulator**”>“ **Topology**” >“ **Device**”>“ **SE-Bal -** ”
DUT 1 1	“**Port 1 (se)**” 1>“ **1**”
DUT 2 3 4	“**Port 2 (bal)**” 2>“ **3-4**”
1 -	“**BalUn**” - “ **ON**”
	“**Fixture Simulator**” “ **ON**”

7.

1. 1 1

1	“**Channel Next**” 1
1 S12	“**Meas**”>“ **S12**”

E5071C

2. 1 23 4

8.

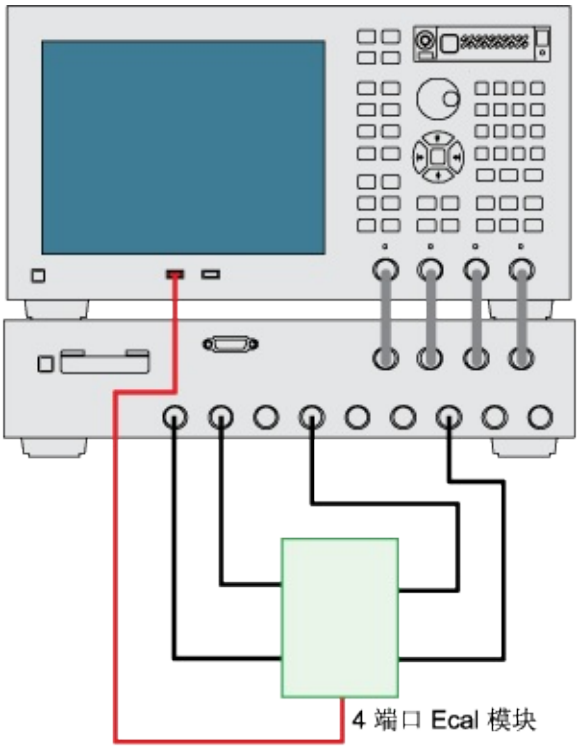
1. “Ecal”

“ECal”	“**Cal**”>“ **ECal**”

2. 1

	Channel Next

3. E5091A 1 4 4 ECal



e5071c413

4.

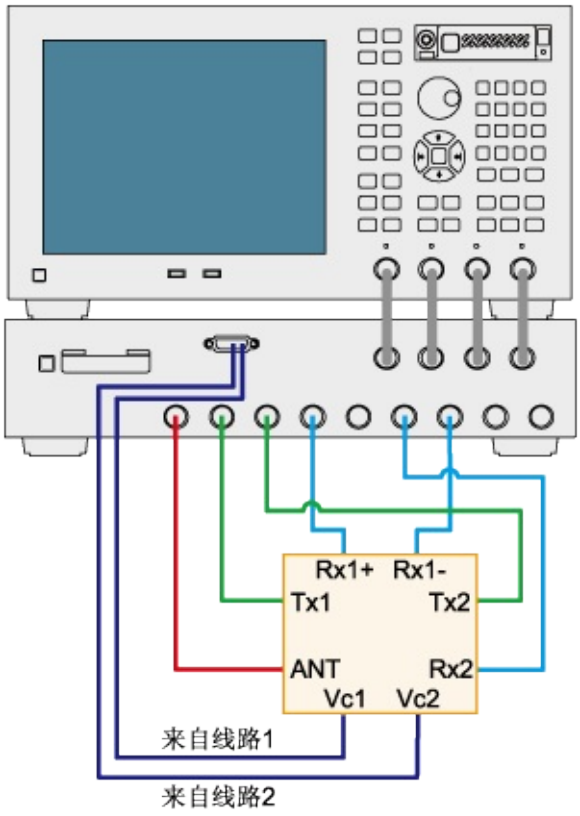
2	**2-Port ECal2**
	1-2

5. 2 4 23 4

1 2 3 4 ECal

9. DUT

DUT



e5071c412

10.

1. “”

“”	**Trigger**

2. “”

“”	**“Trigger Source”>“ Manual”**

3. 1 “”

1	**Channel Next**
“”	**Continuous**

4. 3 23 4 “”

5.

	Trigger

6.

	Channel Next
	“Scale”>“ Auto Scale”

-
-

E4418B E4412A

1.	E5071C
2.	E5071C GPIB
3.	
4.	
5.	
6.	
7.	

1.

E4418B GPIB GPIB 14 USB/GPIB E5071C

2.

GPIB

GPIB 14	“**System**”>“ **Misc Setup**”>“ **GPIB Setup**”GPIB >“ **Power Meter Address**”> **1** > **4** > **x1**

3.

	“**Preset**”>“ **OK**”
	“**Sweep Setup**”>“ **Sweep Type**”>“ **Power Sweep**”
1 GHz	“**Sweep Setup**”>“ **Power**”>“ **CW Freq**”CW > **1** > **G/n**
-40 dBm	“**Start**”> **+/-** > **4** > **0** > **x1**
-10 dBm	“**Stop**”> **+/-** > **1** > **0** > **x1**
61	“**Sweep Setup**”>“ **Points**”> **6** > **1** > **x1**

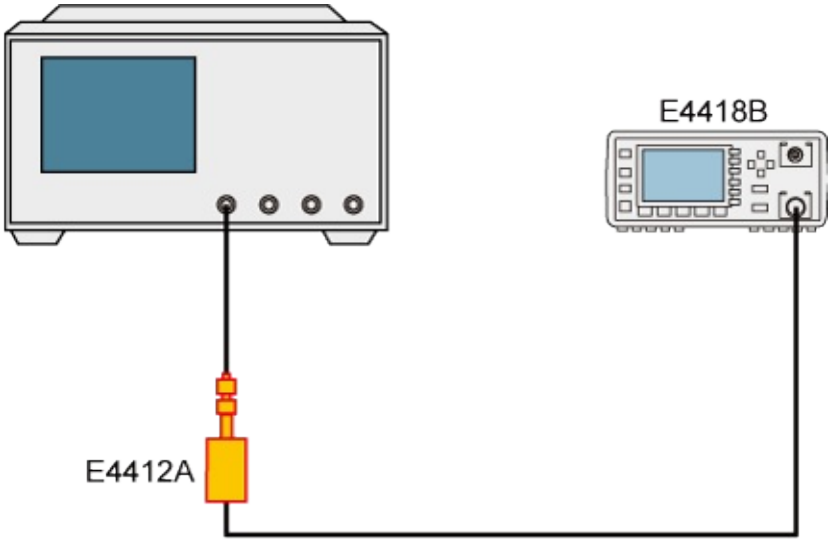
4.

E4418B

5.

1	“Cal”>“ Power Calibration”>“ Select Port” > **1**
A	“Cal”>“ Power Calibration”>“ Use Sensor [A]” [A]
4	“Cal”>“ Power Calibration”>“ Num of Readings”> 4 > x1
5 dB	“Cal”>“ Power Calibration”>“ Tolerance” > 5 > x1

6.



e5071c335

7.

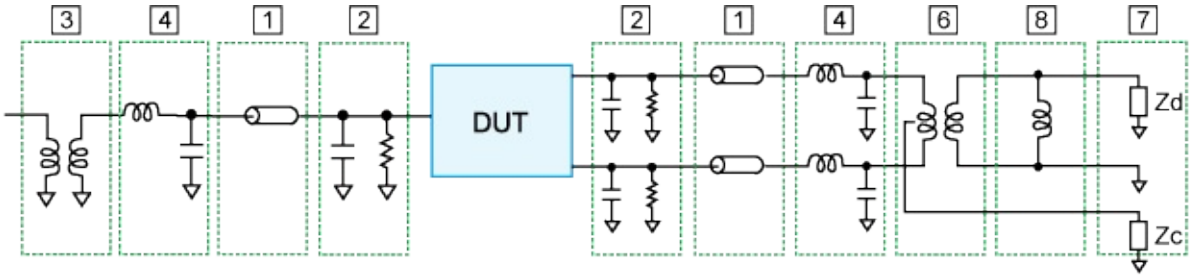
	“**Cal**”>“ **Power Calibration**”>“ **Take Cal Sweep**”

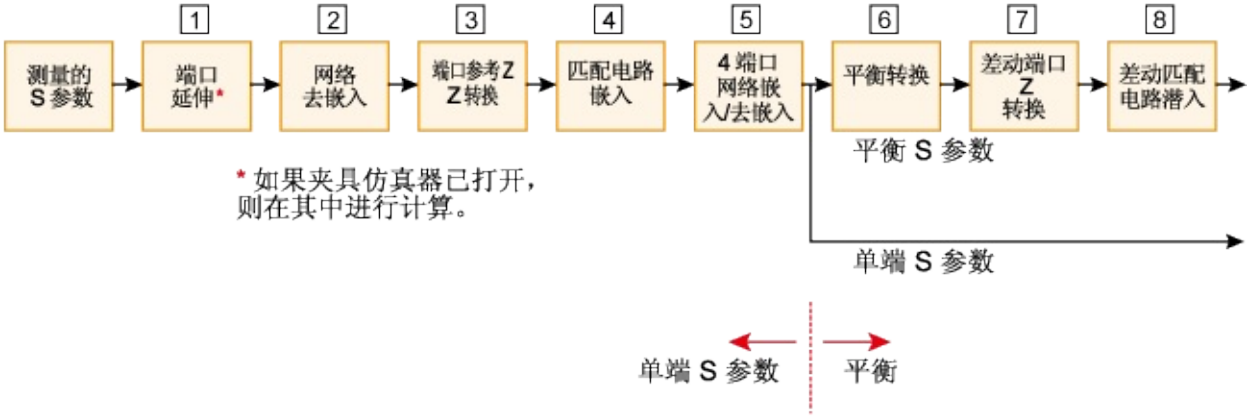
- .
- .
- .
- .
- / 4
- -
- .
- .
- .

-
-

E5071C

- .
- .
- .
- 4 /
- -
- /
- .





e5071c354

-

Touchstone 50 Ω DUT DUT

50 Ω S

DUT Touchstone

4 /

4 Touchstone

/ 4

- 3 4

DUT DUT E5071C E5071C DUT

- -

3 4

-

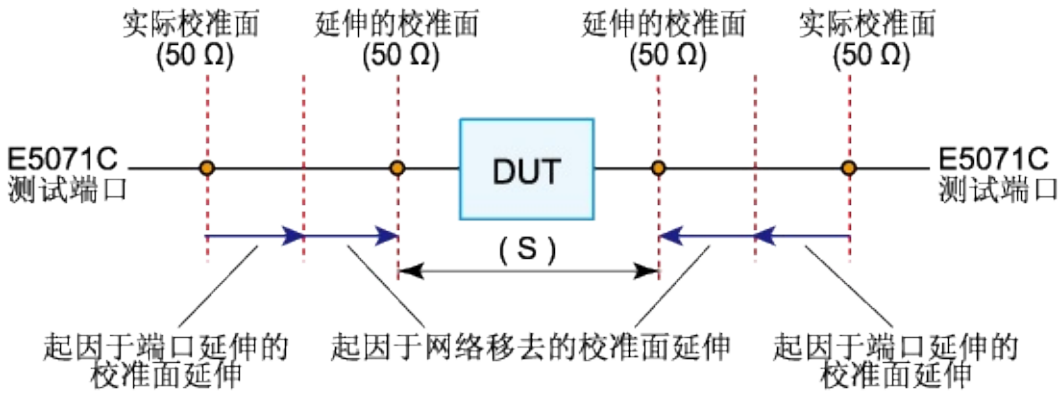
/

- Z0 - Z0 2 Z0 1/2 -

-

-
-

Touchstone DUT

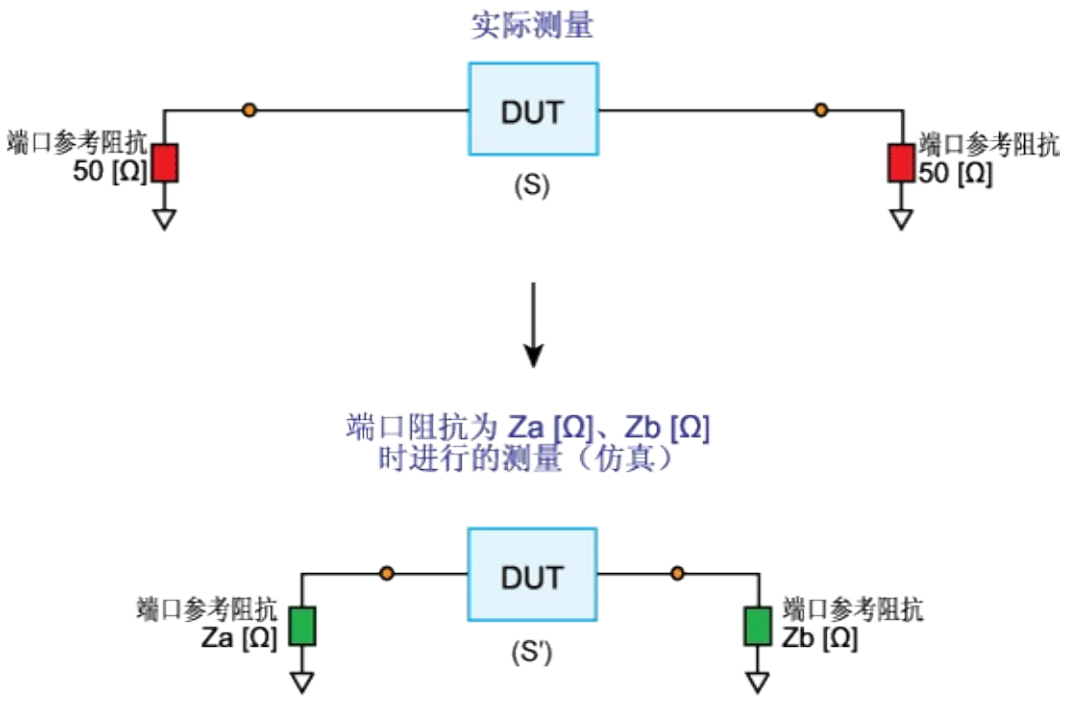


e5071c454

1. Touchstone .s2p
2. “ **Analysis**”“ **Fixture Simulator**”>“ **De-Embedding**”
3. “ **Select Port**”
4. “ **1**”“ **2**”“ **3**”“ **4**”
5. “ **User File**”
6. Touchstone “ **Select Type**”“ **User**” “ **Select Type**”>“ **None**”
7. Touchstone
8. “ **De-Embedding**”“ **ON**”
9. “ **Return**”
10. “ **Fixture Simulator**”“ **OFF**”“ **ON**”

-
-

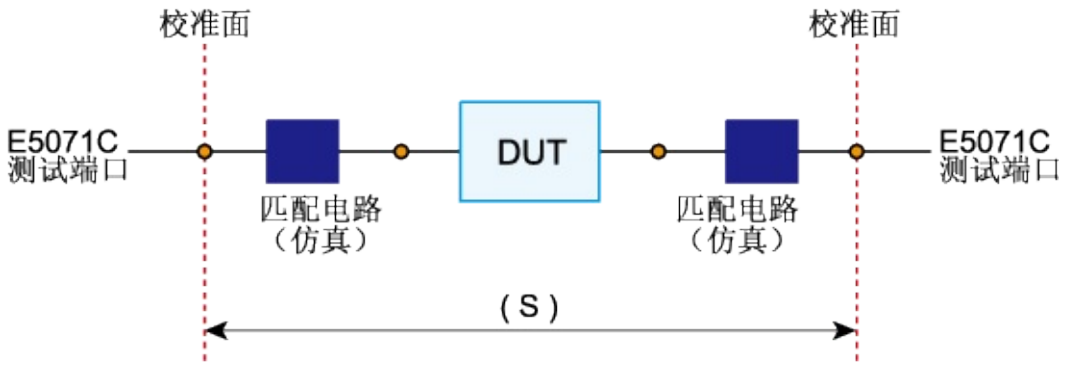
50 W



e5071c377

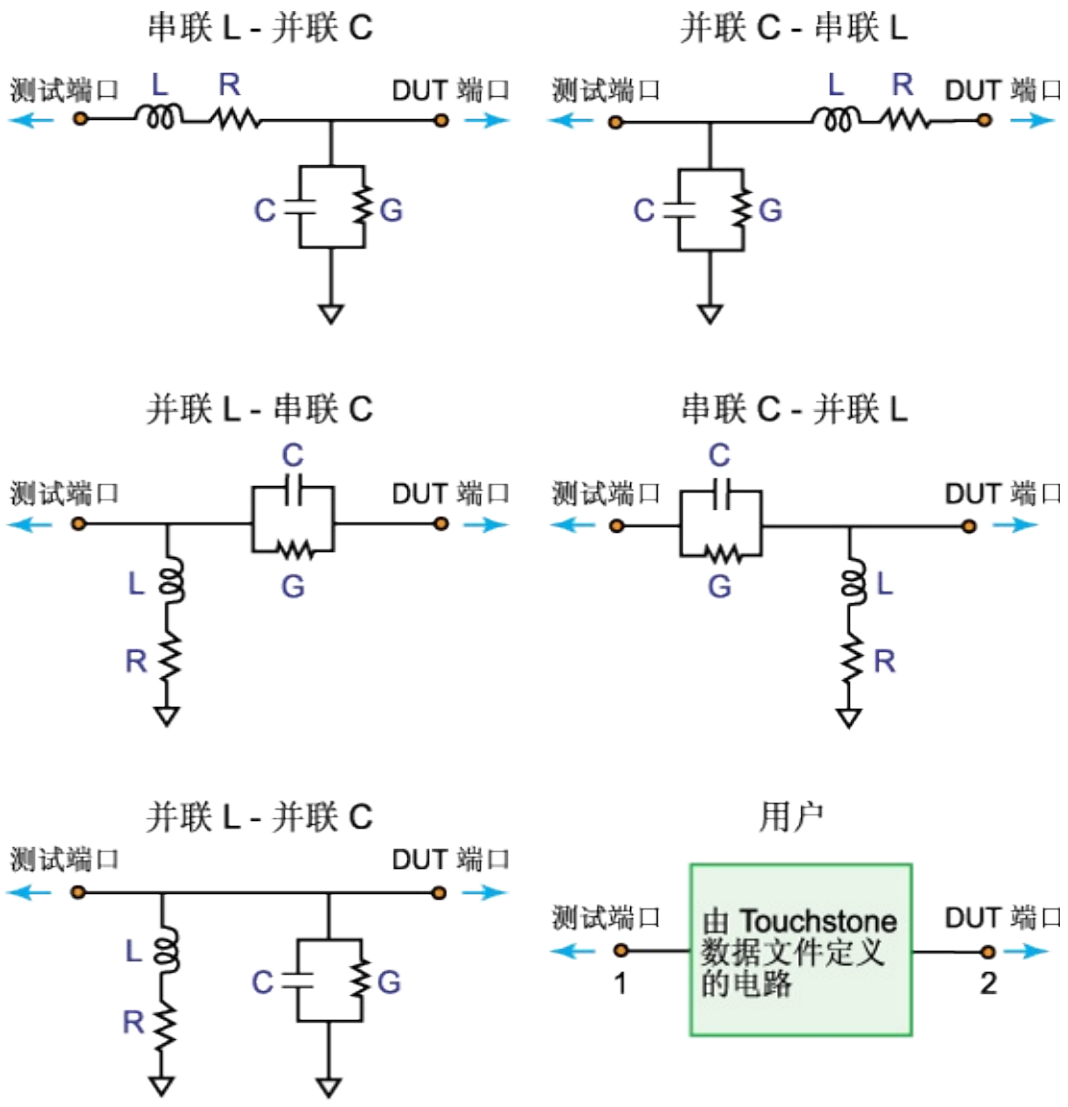
1. “ **Analysis**”“ **Fixture Simulator**”>“ **Port Z Conversion**” Z
2.
   - “ **Port 1 Z0 Real**” 1 Z0 “ **Port 2 Z0 Real**” 2 Z0 “ **Port 3 Z0 Real**” 3 Z0 “ **Port 4 Z0 Real**” 4 Z0
   - “ **Port 1 Z0 Real**” 1 Z0 “ **Port 1 Z0 Imag**” 1 Z0 “ **Port 2 Z0 Real**” 2 Z0 “ **Port2 Z0 Imag**” 2 Z0 “ **Port 3 Z0 Real**” 3 Z0 “ **Port 3 Z0 Imag**” 3 Z0 “ **Port 4 Z0 Real**” 4 Z0 “ **Port 4 Z0 Imag**” 4 Z0
3. “ **Port Z Conversion**” Z “ **ON**”
4. “ **Return**”
5. “ **Fixture Simulator**”“ **OFF**”“ **ON**”

-
-



e5071c453

-
- 2 Touchstone



e5071c380

2 2 2 Touchstone

2 Touchstone 3202 3202

1 2 DUT

1. “ **Analysis**”
2. “ **Fixture Simulator**”>“ **Port Matching**”>“ **Select Port**”
3. “ **1**”“ **2**”“ **3**”“ **4**”
4. “ **Select Circuit**”
5.

None	
Series L - Shunt C L - C	
Shunt C - Series L C - L	
Shunt L - Series C L - C	
Series C - Shunt L C - L	
Shunt L - Shunt C L - C	
User	1. “ **User File**” 2. 2 Touchstone .s2p

	“ **Select Circuit**” “ **User**”

6.

7.

C	[F]
G	[S]
L	[H]
R	[Ω]

注意 C “C”“G” 0 “Open” 0 “Short”
L “L”“R” 0 “Short” 0 “Open”

8.

9. “ **Port Matching**”“ **ON**”

10. “ **Return**”

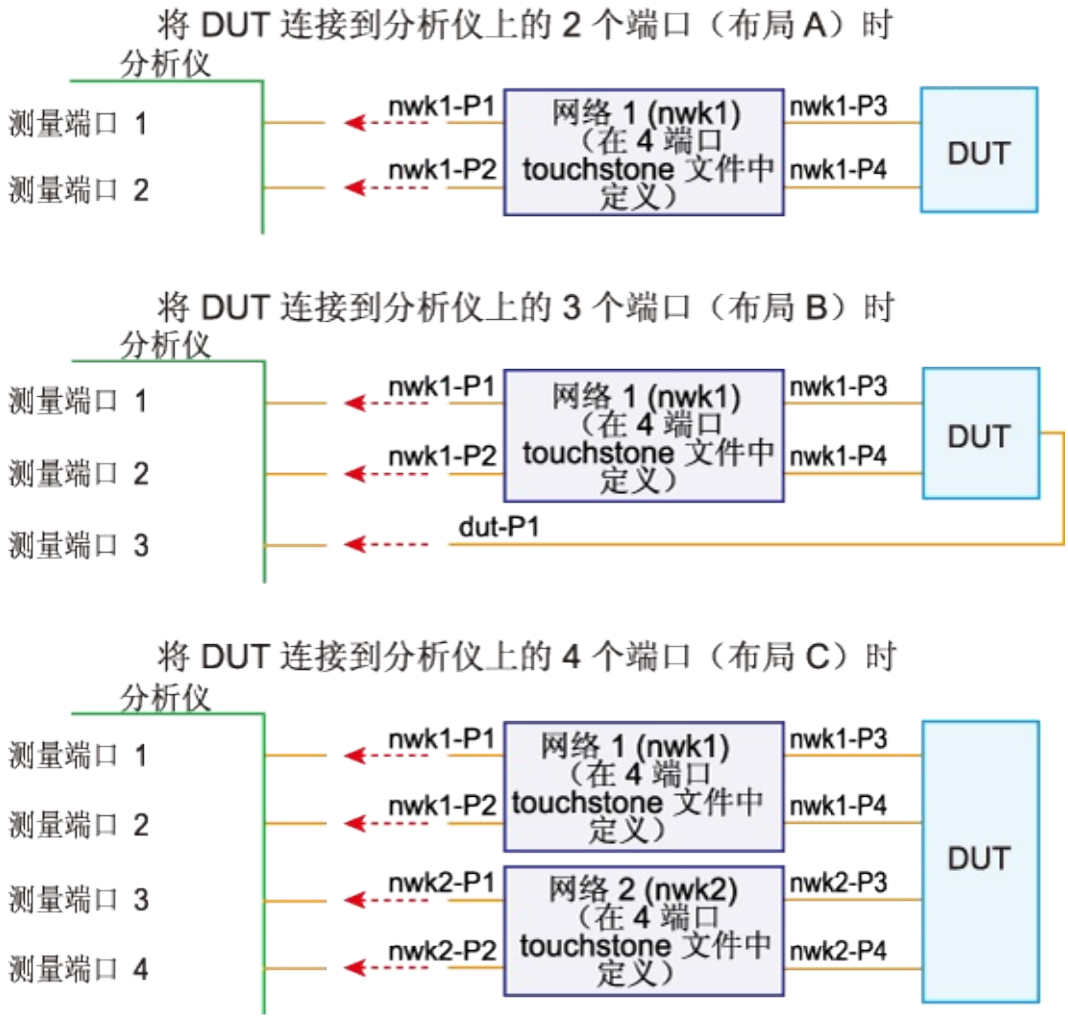
11. “ **Fixture Simulator**”“ **OFF**”“ **ON**”

/ 4

-
-

4 / 4 Touchstone 4 /

/4 Touchstone



e5071c455

1. / 4 Touchstone .s4p
2. “ **Analysis**”
3. “ **Fixture Simulator**”>“ **De-Embedding S4P**” S4P >“**Topology**”>“ **Select Topology**”
4. AB C
5. “ **Ports**” 4 1 2
   - **A** nwk1-P1 nwk1-P2 nwk1-P1 nwk1-P2 1 2 “Ports”“1-2”
   - **B** nwk1-P1nwk1-P2 dut-P1 nwk1-P1nwk1-P2 dut-P1 12 3 “Ports”“1-2-3”
   - **C** nwk1-P1nwk1-P2nwk2-P1 nwk2-P2 nwk1-P1nwk1-P2nwk2-P1 nwk2-P2 123 4 “Ports”“1-2-3-4”

注意

4 Touchstone 4 1 2 3 4 DUT

4 / 4 DUT C A/ “None”

4 / 2 / 3 DUT B A 2 2 / 4 / 2 A

6. “ **User File (nwk1)**” (nwk1) 1 4 Touchstone
7. “ **Type (nwk1)**” (nwk1) 1
8. C 2 (**nwk2**)
9. “ **De-Embedding S4P**” S4P“ **ON**”4 /
10. “ **Return**”
11. “ **Fixture Simulator**”“ **OFF**”“ **ON**”

4 / 4 Touchstone 4 Touchstone

4 Touchstone E5071C VBA
4 Touchstone

Touchstone 4 / Touchstone 4
C 1 2 3 4

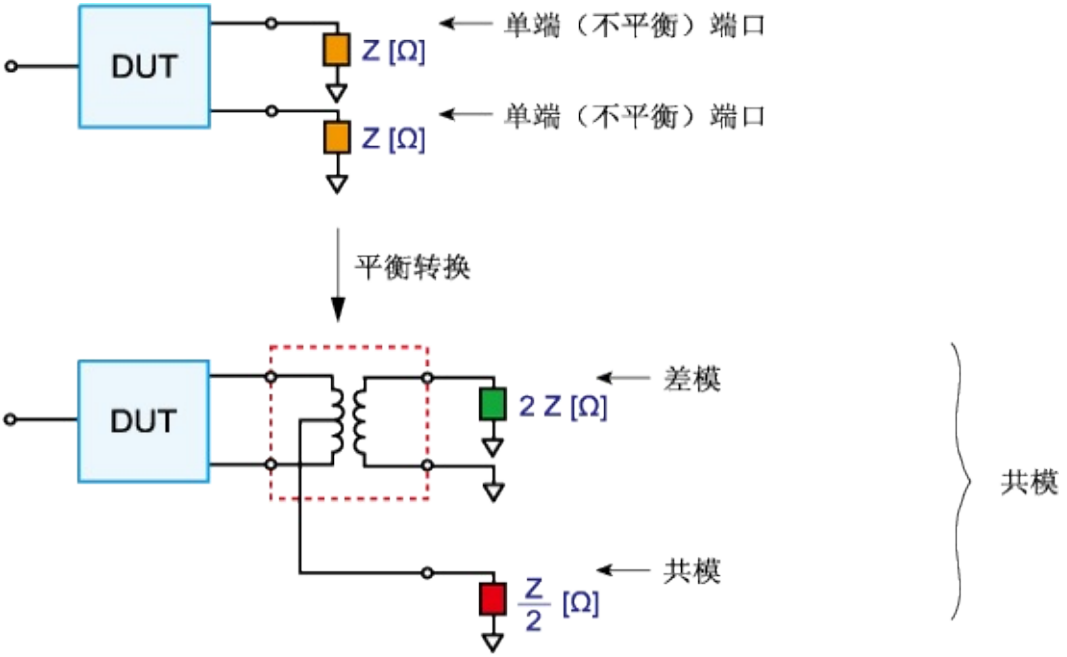
4 Touchstone 3202 3202

-

-
-
- -
-
-

-

-

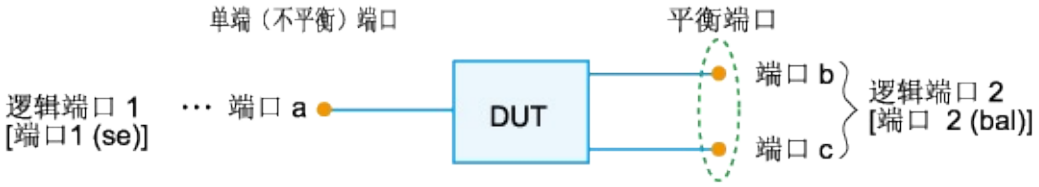


e5071c406

E5071C E5071C

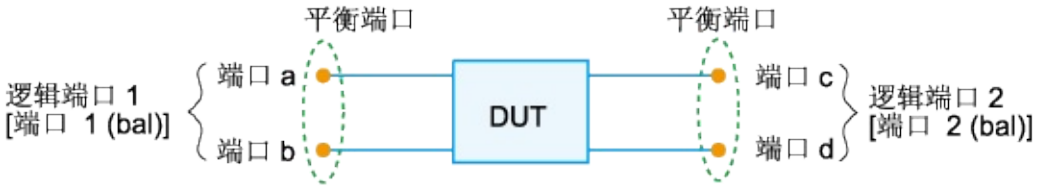
E5071C

单端 - 平衡设备 (SE-Bal)



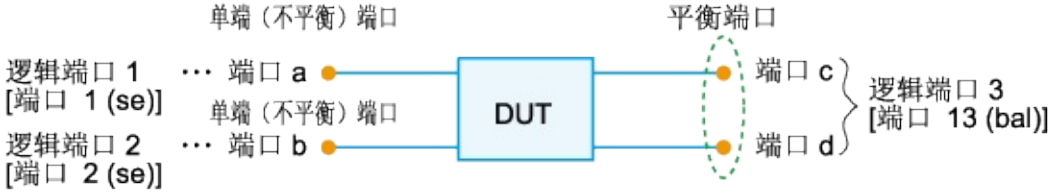
分析仪所需的测试端口数：3

平衡－平衡设备 (Bal-Bal)



分析仪所需的测试端口数：4

单端－单端－平衡设备 (SE-SE-Bal)



分析仪所需的测试端口数：4

用 [ ] 括起来的是功能键标记。

e5071c375

E5071C DUT

E5071C a d

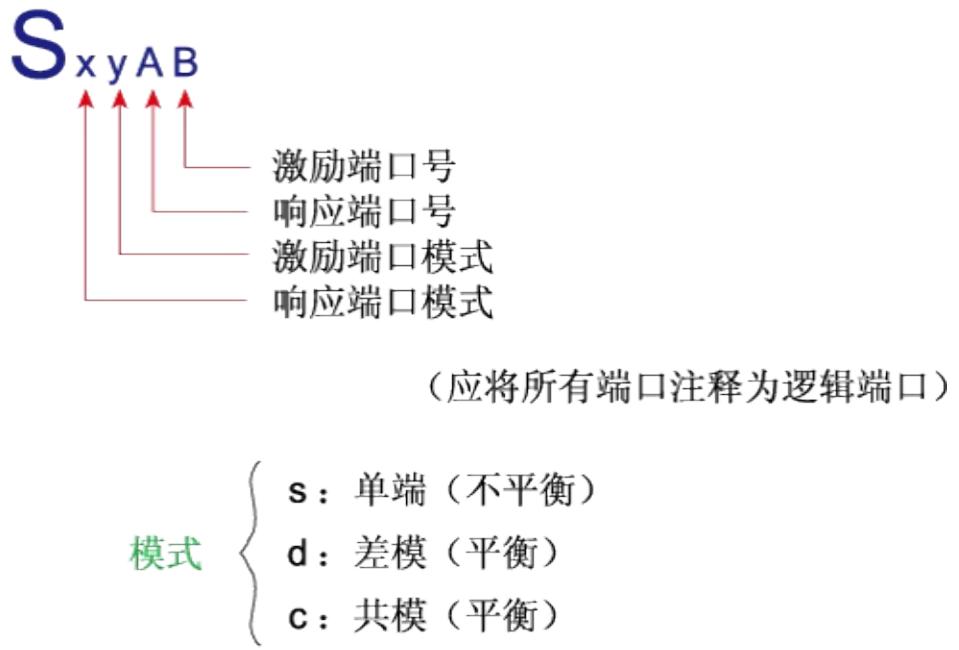
-

- S
- .
- CMRR

S

- S S S

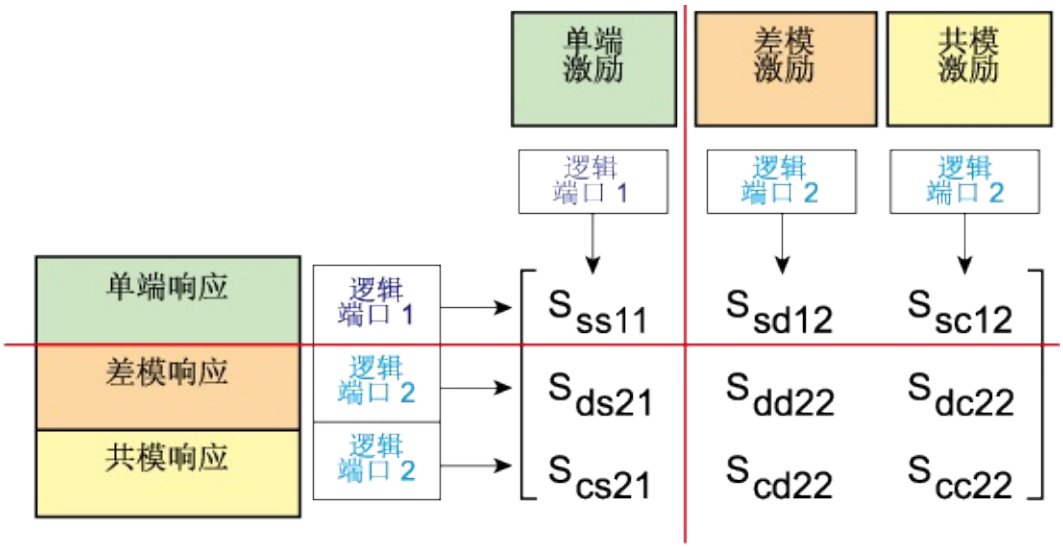
S



e5071c381

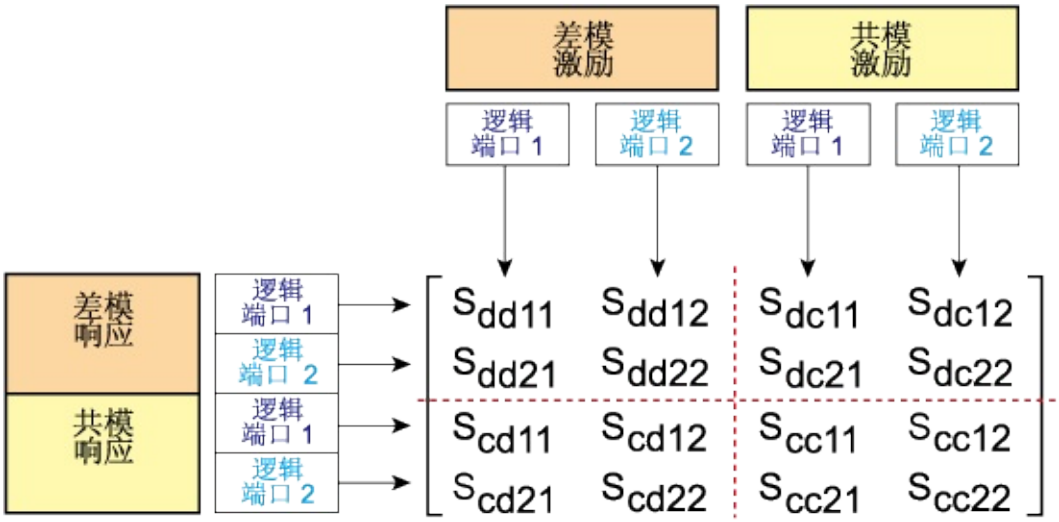
S

- S



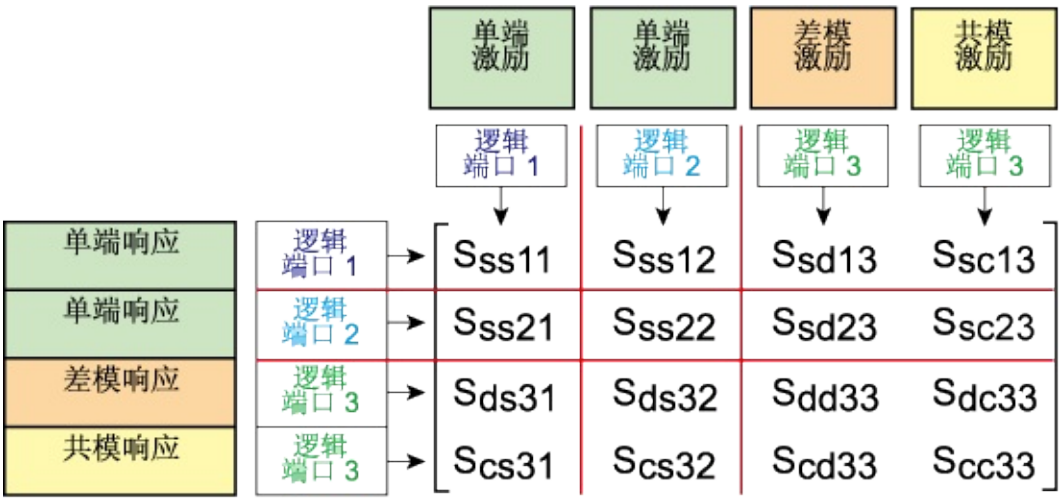
e5071c408

- S



e5071c407

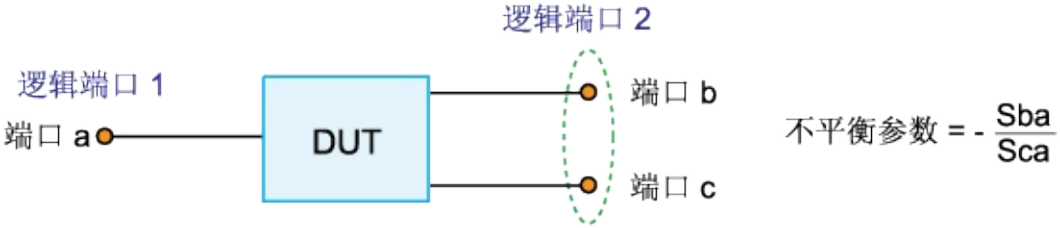
- - S



e5071c409

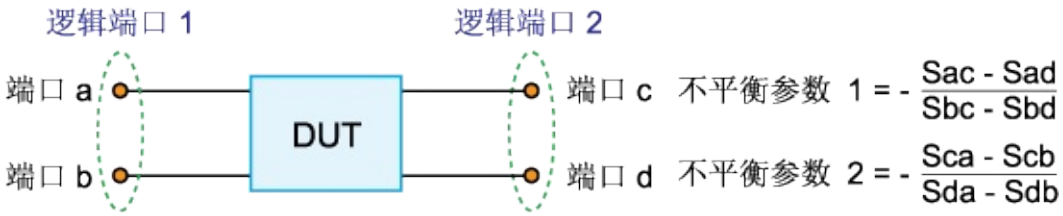
-

-



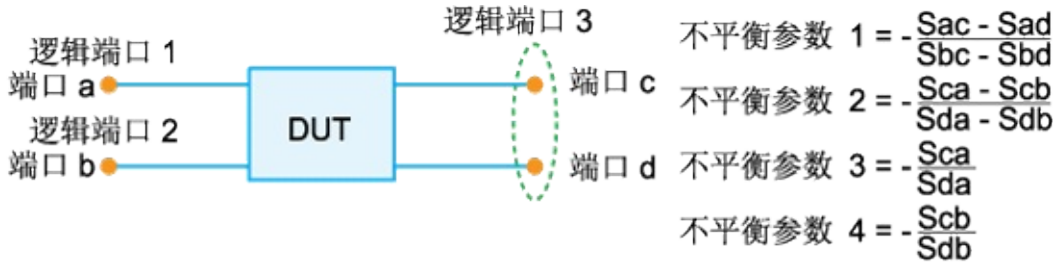
e5071c410

- 1 2



e5071c411

- - 1 2 3 4



e5071c456

CMRR

- CMRR CMRR

	CMRR
-	S_{ds21}/S_{cs21} S_{sd12}/S_{sc12}
-	S_{dd21}/S_{cc21}
- -	S_{ds31}/S_{cs31} S_{sd32}/S_{sc32}

-

1. “ **Analysis**”
2. “ **Fixture Simulator**”>“ **Topology**”>“ **Device**”
3. /

SE-Bal-	DUT 1 2
Bal-Bal-	DUT 1 2
SE-SE-Bal--	DUT 1 2 3

4. DUT

- “ **SE-Bal**”-

Port 1 (se) **1**	“ **1**”“ **2**”“ **3**”“ **4**” 1 a
Port 2 (bal) **2**	“ **1-2**”“ **1-3**”“ **1-4**”“ **2-1**”“ **2-3**”“ **2-4**”“ **3-1**”“ **3-2**”“ **3-4**”“ **4-1**”“ **4-2**”“ **4-3**” 2 b c

- “ **Bal-Bal**”-

Port 1 (bal) 1	**“ 1-2”“ 1-3”“ 1-4”“ 2-1”“ 2-3”“ 2-4”“ 3-1”“ 3-2”“ 3-4”“ 4-1”“ 4-2”“ 4-3”** 1 a b
Port 2 (bal) 2	**“ 1-2”“ 1-3”“ 1-4”“ 2-1”“ 2-3”“ 2-4”“ 3-1”“ 3-2”“ 3-4”“ 4-1”“ 4-2”“ 4-3”** 2 c d

- “ **SE-SE-Bal**”--

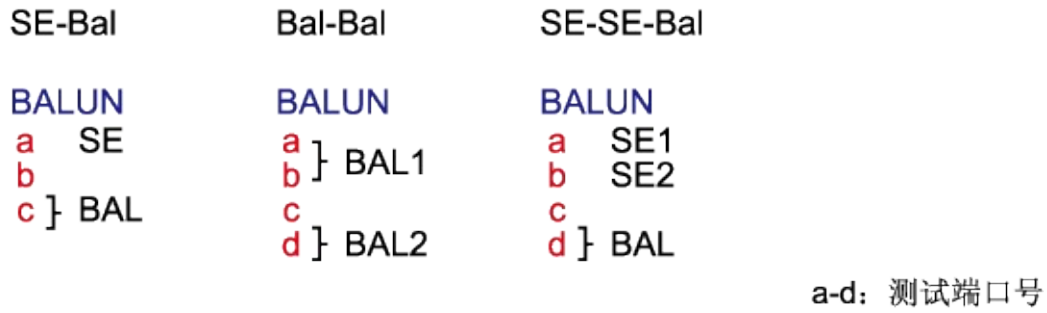
Port 1 (se) 1	**“ 1”“ 2”“ 3”“ 4”** 1 a
Port 2 (se) 2	**“ 1”“ 2”“ 3”“ 4”** 2 b
Port 3 (bal) 3	**“ 1-2”“ 1-3”“ 1-4”“ 2-1”“ 2-3”“ 2-4”“ 3-1”“ 3-2”“ 3-4”“ 4-1”“ 4-2”“ 4-3”** 3 c d

5. “ **Return**”
6. “ **BalUn**” - -
7. “ **Fixture Simulator**”

- S CMRR - -

1. " **Meas**"" **Analysis**">" **Fixture Simulator**" >"**Measurement**"
2.

-



e5071c445

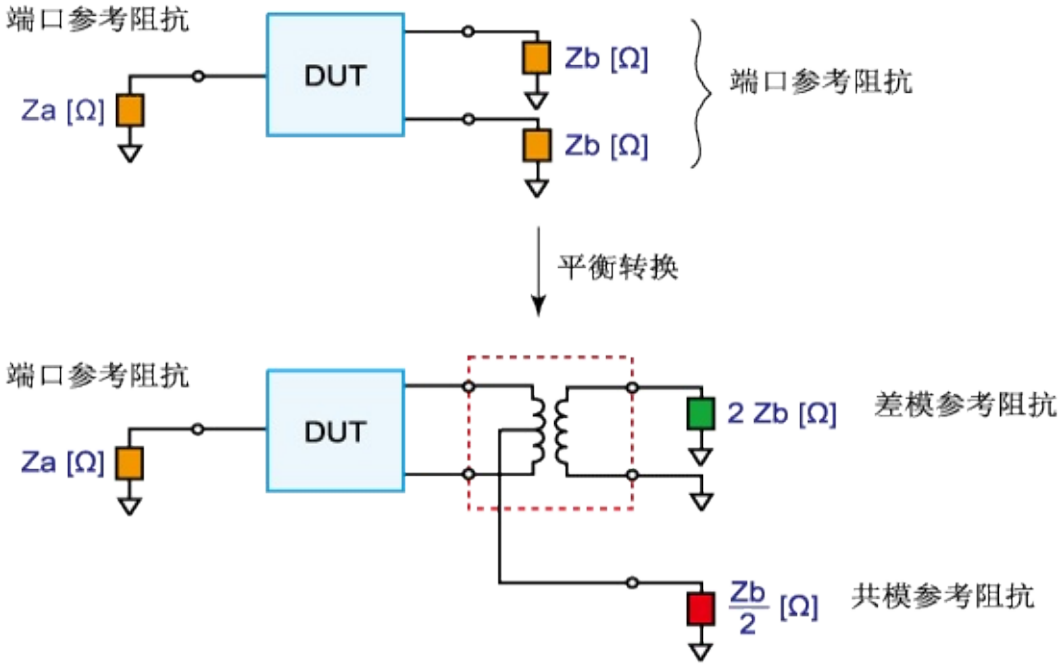
/

/

1. “ **Analysis**”
2. “ **Fixture Simulator**”>“ **Topology**”
3. “ **Property**”/

-
-
-

-


端口参考阻抗
DUT
Za [Ω]
Zb [Ω]
Zb [Ω]
端口参考阻抗
平衡转换
端口参考阻抗
DUT
Za [Ω]
2 Zb [Ω]
差模参考阻抗
Zb/2 [Ω]
共模参考阻抗

e5071c403

-

/

1. “ **Analysis**”“ **Fixture Simulator**”>“ **Diff Z Conversion**” Z
2. “ **Diff Z Conversion**” Z “ **ON**”

1. “ **Analysis**”“ **Fixture Simulator**”>“ **Diff Z Conversion**” Z
2.
3. “ **Port 1 (bal) Real**” 1“ **Port 2 (bal) Real**” 2 “ **Port 3 (bal) Real**” 3“”
4. “ **Port 1 (bal) Real**” 1“ **Port 1 (bal) Imag**” 1 “ **Port 2 (bal) Real**” 2“ **Port2 (bal) Imag**” 2“ **Port 3 (bal) Real**” 3“ **Port 3 (ba Imag**”“”

12 3 12 3

/

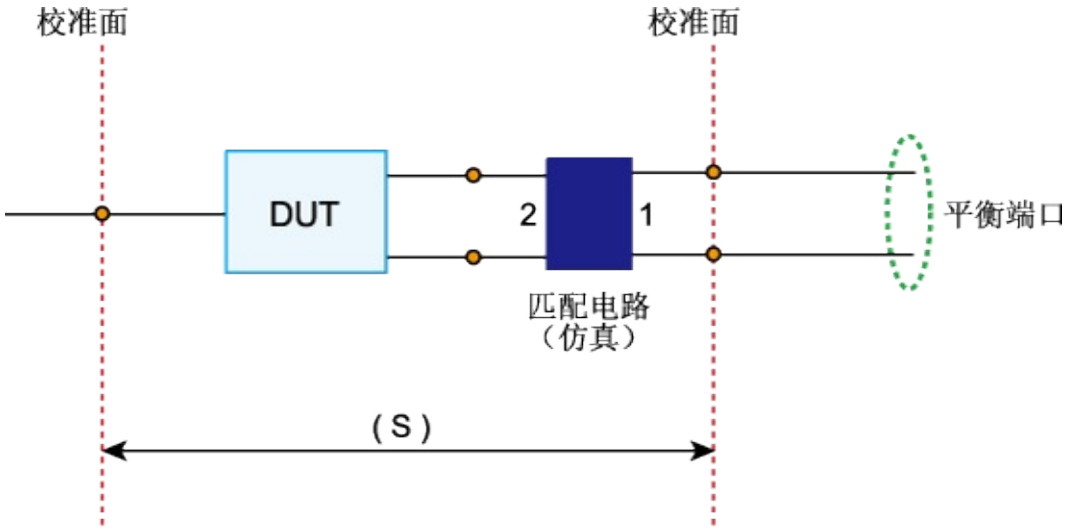
1. “ **Analysis**”“ **Fixture Simulator**”>“ **Cmn Z Conversion**” Z
2. “ **Cmn Z Conversion**” Z “ **ON**”
3.

1. “ **Analysis**”“ **Fixture Simulator**”>“ **Cmn Z Conversion**” Z
2.
3. “ **Port 1 (bal) Real**” 1“ **Port 2 (bal) Real**” 2 “ **Port 3 (bal) Real**” 3“”
4. “ **Port 1 (bal) Real**” 1“ **Port 1 (bal) Imag**” 1 “ **Port 2 (bal) Real**” 2“ **Port2 (bal) Imag**” 2“ **Port 3 (bal) Real**” 3“ **Port 3 (ba Imag**”“”

12 3 12 3

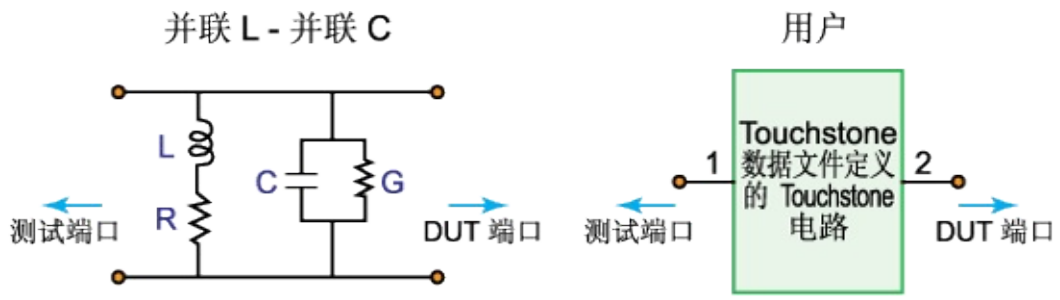
-
-

-



e5071c376

-
- 2 Touchstone



e5071c379

1 2 DUT

1. “ **Analysis**”“ **Fixture Simulator**”>“ **Diff Matching**”
2. “ **Select Port**”
3. “ **1**”“ **2**”“ **3**” DUT
4.
   a. “ **User File**”
   b. 2 Touchstone .s2p
   c. “ **Select Circuit**”“ **User**” 5 6

5. “ **Select Circuit**”
6.

None	
Shunt L - Shunt C L - C	
User	4

7.

C	[F]
G	[S]
L	[H]
R	[Ω]

注意 L “L”“R” 0 “Short” 0 “Open”

8. DUT
9. “ **Diff. Matching**”“ **ON**”
10. “ **Return**”
11. “ **Fixture Simulator**”“ **OFF**”“ **ON**”

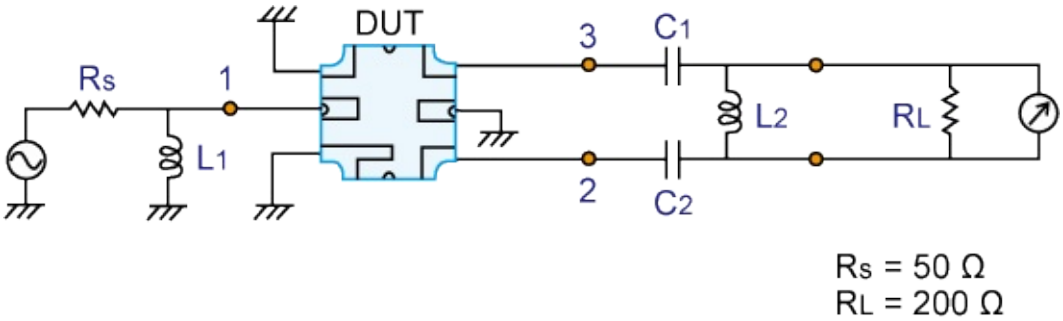
-
- DUT
-
-
- E5071C DUT
- DUT

DUT SAW

DUT

SAW DUT 1 R s L 1 DUT 2 C 1C2 L 2 RL

SAW



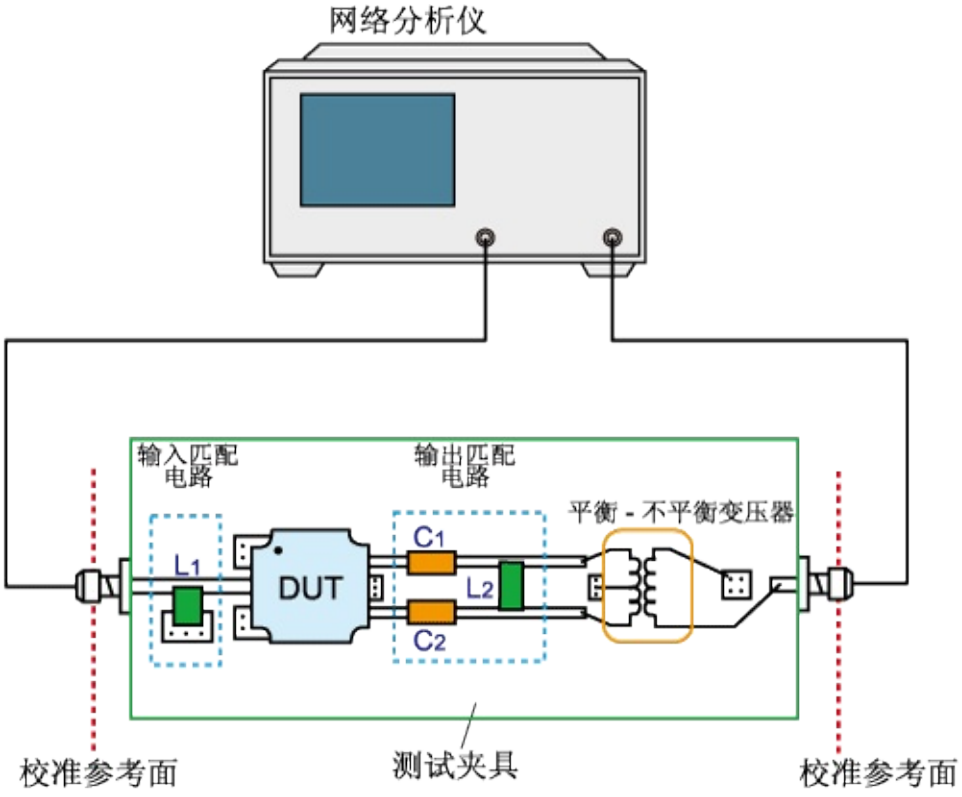
e5071c420

DUT

50 Ω DUT 1

DUT 2 balun - DUT

DUT



e5071c404

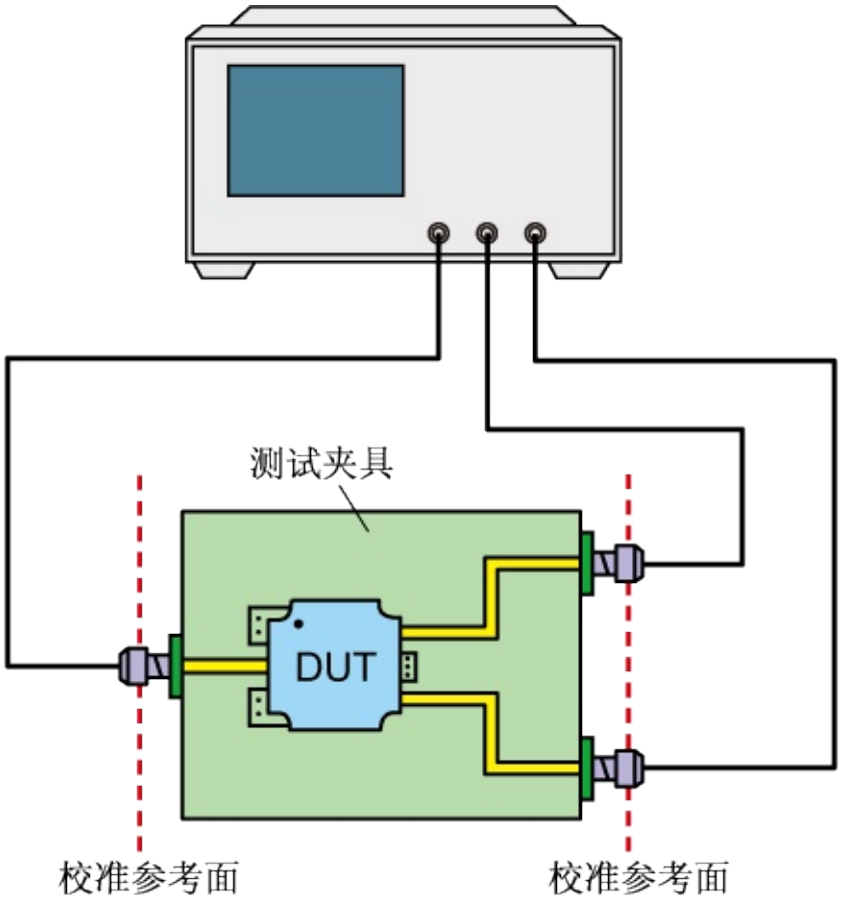
- DUT DUT DUT
- DUT DUT
- balun balun

E5071C DUT

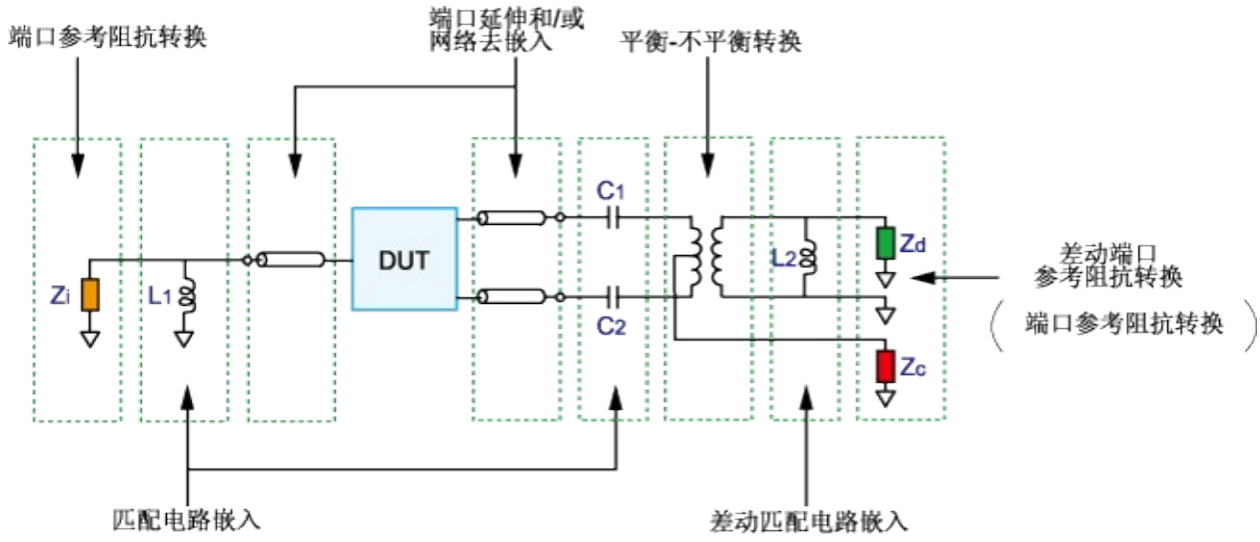
E5071C DUT

E5071C DUT DUT E5071C DUT E5071C
E5071C 50 Ω

DUT



e5071c362



e5071c405

/ DUT DUT / DUT -

S S DUT E5071C 50 Ω -

S DUT S -

- S S

S DUT S “” L2

Z [Ω] Z [Ω] 2 Z [Ω] 1/2 Zd 50 Ω * 2 =
100 Ω Zc 50 Ω / 2 = 25 Ω SAW 200 Ω Zd
200 Ω

DUT

- DUT
- DUT
- - S

008

-
-
-

(MWA)　790

- MWA
- MWA
-
-
-
-
- E5071C VBA

010

-
-
-
-

E5071C 008

E5071C 008

E5071C 008

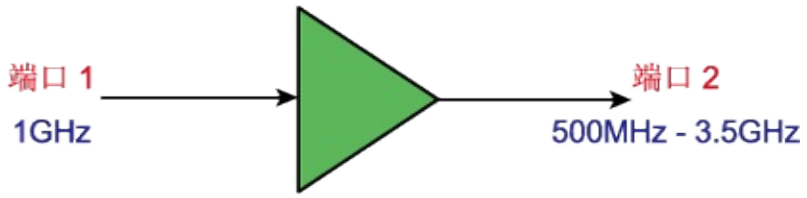
-
-

1.	
2.	
3.	•
4.	• DUT •

1.

1.

设置基频： 500MHz - 3.5GHz
设置端口 1 的频率：M_1=0、D_1=1、O_1=1GHz
设置端口 2 的频率：M_2=1、D_2=1、O_2=0Hz



e5071c463

2.

3.

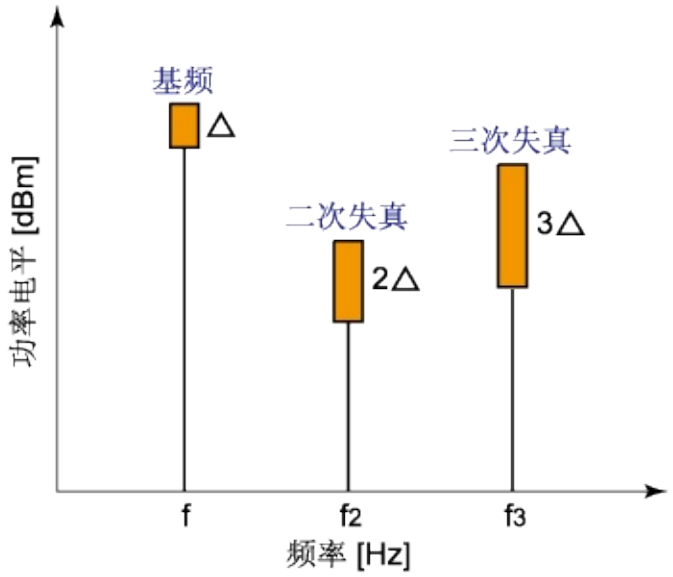
1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Meas**”“Measurement”
3. “ **Absolute**”

A (n)	1
B (n)	2
C (n)	3
D (n)	4
R1 (n)	1
R2 (n)	2
R3 (n)	3
	4

R4 (n)	

n

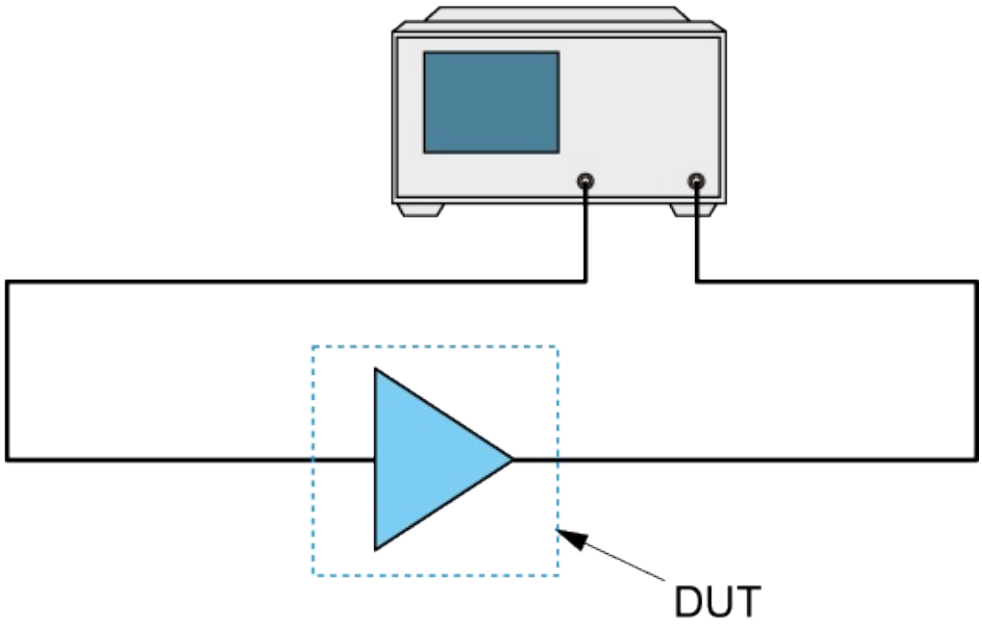
4.



e5071c464

DUT

DUT



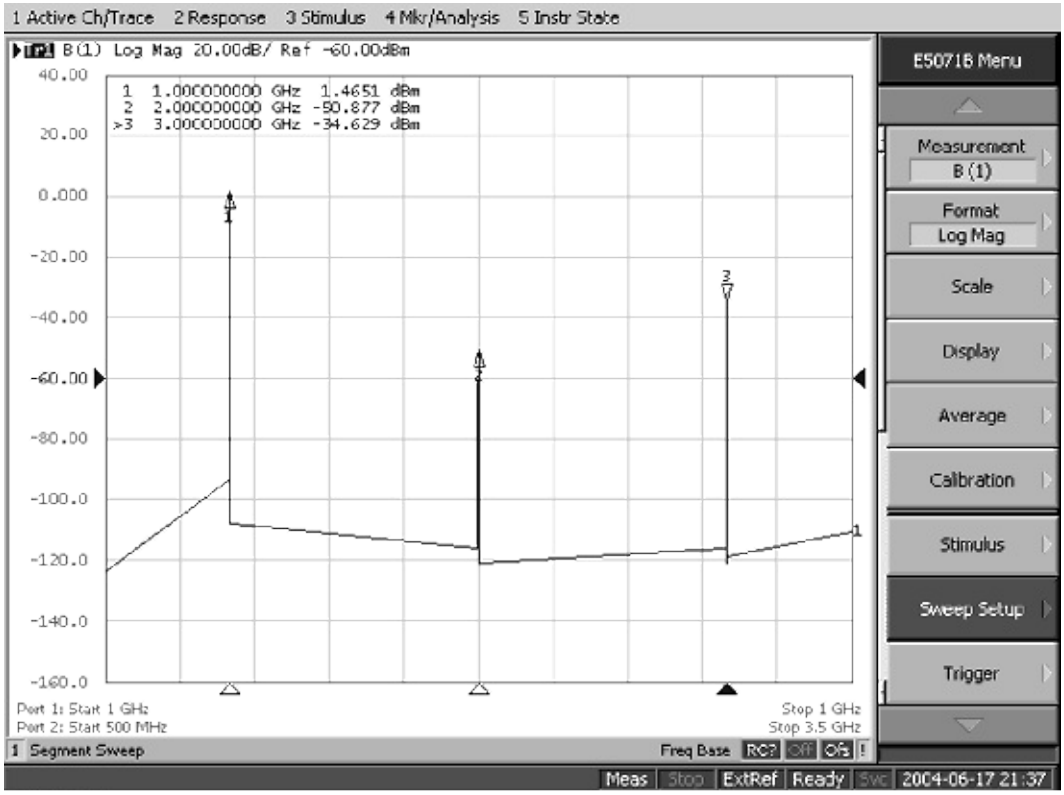
e5071c344

E5071C

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Meas**”“Measurement”
3. “ **Absolute**” (**B (1)**) 1 2
4. _____
5. “ **Sweep Setup**”“ **Edit Segment Table**”“Edit Segment Table”
6. “ **Frequency Mode**”“ **Center/Span**”/ “ **Return**”“Sweep Setup”

500 MHz	0 Hz	1
1 GHz	1 MHz	5
2 GHz	1 MHz	5
3 GHz	1 MHz	5
3.5 GHz	0 Hz	1

7. “ **Segment Display**”“ **Freq Base**”
8. “ **Sweep Type**”“ **Segment**”
9. “ **Scale**”“ **Auto Scale All**”



[Agilent application note 1463-6 Accurate Mixer]

Measurements Using the Frequency-Offset Mode, 5989-1420ENAgilent_1463-6_5989-1420EN Internet

-
-

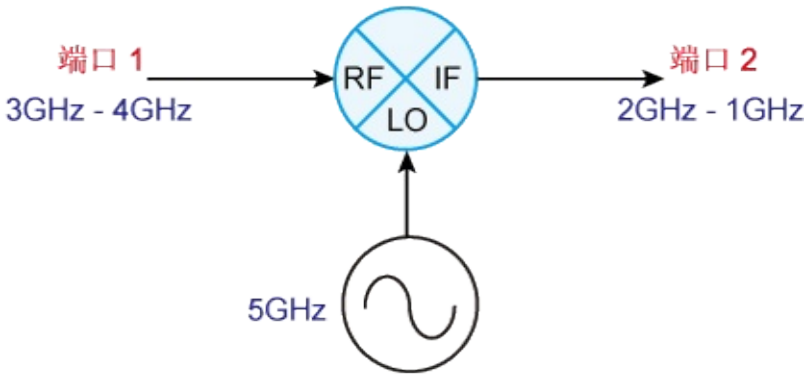
1.	• •
2.	LO • • GPIB • • • •
3.	•
4.	
5.	• • •

1.

nn (Mn) (Dn) (On)

$$< (n) > = <> \times Mn / Dn + On$$

设置基频：3GHz - 4GHz
设置端口 1 的频率：M_1=1、 D_1=1、O_1=0Hz
设置端口 1 的频率：M_2=-1、D_2=1、O_2=5GHz



e5071c461

“ **Start**”“ **Stop**”

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”“ **Frequency Offset**”
   “Frequency Offset”
3. “ **Port n**” n n “Port n” n

4. n

Multiplier	
Divisor	
Offset	
Start	n
Stop	n

“ **Multiplier**”/“ **Divisor**”/“ **Offset**”
“ **Start**”/“ **Stop**”“ **Multiplier**”
/“ **Divisor**”/“ **Offset**”
“ **Start**”/“ **Stop**” M O D

1. “ **Channel Next**”/“ **Channel Prev**”
2. “**Sweep Setup**” “ **Frequency Offset**”“Frequency Offset”
3. “ **Frequency Offset**”“ **ON**”
4.

e5071c178

“**ON**”

- “ **Sweep Mode**”“ **Swept**”“ **Stepped**”
- “ **Format**”“ **Phase**”“ **Group Delay**”
  “ **Smith**”“ **Polar, Real**”“ **Imaginary**”
  “ **Expand Phase**”“ **Positive Phase**”
  “ **Analysis**”“ **Conversion**”
- “**Fixture Simulator**”“ **Time Domain**”“ **Transform**”
  “ **Gating**”“ **OFF**”
- 300 kHz 3 GHz 8.5 GHz
- 2 S “ **ON**”

2.

E5071C USB/GPIB

E5071C USB/GPIB USB GPIB E5071C
GPIB

- USB/GPIB GPIB
- BNC E5071C E5071C

GPIB

GPIB

1. “ **System**”
2. “ **Misc Setup**”>“ **GPIB Setup**”GPIB >“ **Signal Generator Address**”>“ **Address**”

GPIB

E5071C

1	
2	8643A8644B8664A8665A/B
3	8648A/B/C/DESG PSG

- 1 RF
-

	""	
RF	"R3"	
	"FR %f% HZ"	(Hz) %f%
	"AP %p% DM"	(dBm) %p%

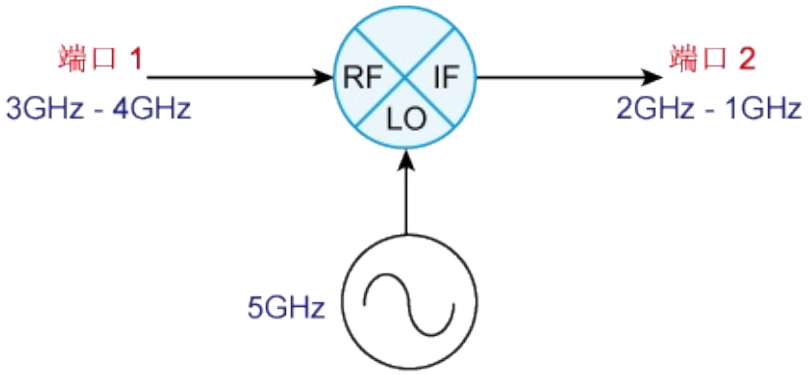
1. “ **System**”
2. “ **Misc Setup**”>“ **GPIB Setup**”GPIB >“ **Signal Generator Address**”
3.

“ **Switching Time**”ms

(MLO) (DLO) (OLO)

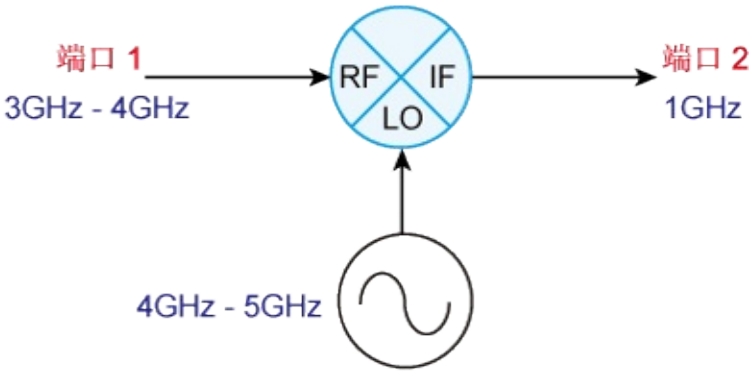
$$<LO> = <> \times MLO / DLO + OLO$$

设置基频：3GHz - 4GHz
设置端口 1 的频率：M_1=1、D_1=1、O_1=0Hz
设置端口 2 的频率：M_2=-1、D_2=1、O_2=5GHz
设置 LO 的频率：M_{LO}=0、D_{LO}=1、O_{LO}=5GHz



e5071c459

设置基频：3GHz - 4GHz
设置端口 1 的频率：M_1=1、D_1=1、O_1=0Hz
设置端口 2 的频率：M_2=0、D_2=1、O_2=1GHz
设置 LO 的频率：M_{LO}=1、D_{LO}=1、O_{LO}=1GHz



e5071c460

“ **Start**”“ **Stop**”

1. “ **Sweep Setup**”“ **Frequency Offset**”
   “Frequency Offset”

2. “ **External Source**”>“ **Power**”

3. :

“ **Multiplier**”/“ **Divisor**”/“ **Offset**”
“ **Start**”/“ **Stop**”“ **Multiplier**”
/“ **Divisor**”/“ **Offset**”
“ **Start**”/“ **Stop**” M O D

1. “ **Sweep Setup**”“ **Frequency Offset**”
“Frequency Offset”

2. “ **External Source**”>“ **Power**”

3.

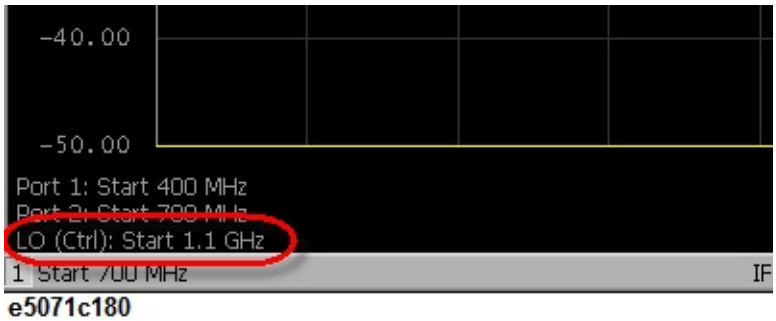
LO “External Source”“ **Slope**”
dB/GHz

1. “ **Sweep Setup**”“ **Frequency Offset**”
“Frequency Offset”

2. “ **External Source**”“External Source”

3. “ **LO Frequency**”LO “ **ON**” (LO)

4. “ **Control**”“ **ON**”“ **ON**”“(Ctrl)”



e5071c180

GPIB GPIB

3.

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Sweep Setup**”“ **Frequency Offset**” “Frequency Offset”
3. “ **X-Axis**”X

- “ **Start**”“ **Stop**”
-

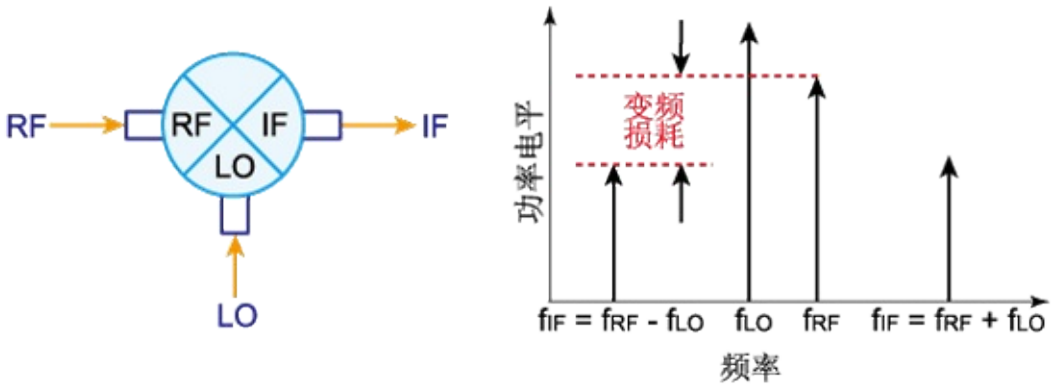
4.

E5071C

5.

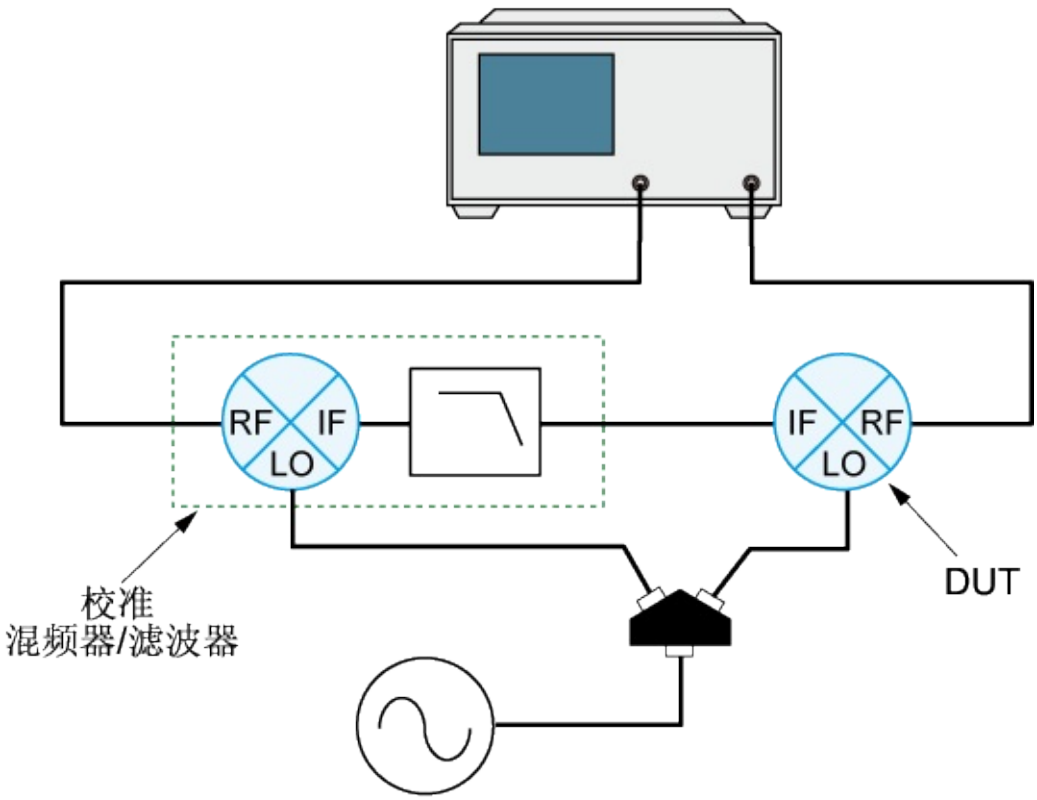
LO

$$\text{变频损耗 [dB]} = 10 \log \frac{\text{Mag}\ (f_{IF})\ \text{mW}}{\text{Mag}\ (f_{RF})\ \text{mW}}$$



e5071c462

IF



e5071c343

6.3 Vp-p

(S12)

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Display**”“ **Num of Traces**” **3**
3. “ **Allocate Traces**”“Graph Layout”( )
4. “ **Meas**”“ **S12**”
5. “ **Format**”“ **LogMag**”“Trace 1” 1
6. “ **Trace Next**”“Trace 2” 2
7. “ **Meas**”“ **S12**”

8. “ **Format**”“ **Phase**”“Trace 2” 2
9. “ **Trace Next**”“Trace 3” 3
10. “ **Meas**”“ **S12**”
11. “ **Format**”“ **Group Delay**”“Trace 3” 3
12. “ **Scale**”“ **Auto Scale All**”

Agilent application note 1463-6 Accurate Mixer Measurements Using the Frequency-Offset Mode, 5989-1420ENAgilent_1463-6_5989-1420EN Internet

MWA

(MWA) PC E5071C E5091A MWA

1. 9 16 E5091A 13 E5091A-016
2. DUT 30 DUT DUT
3. GPIB DUT
4. Excel
5.

MWA

MWA

-
-
-
-

MWA

MWA

PC Excel VBA Excel E5071C E5091A

- E5091A
- DUT E5091A
-
- GPIB
-
-
-

E5071C VBA

-
-
- GPIB
-
-

DUT GPIB E5071C

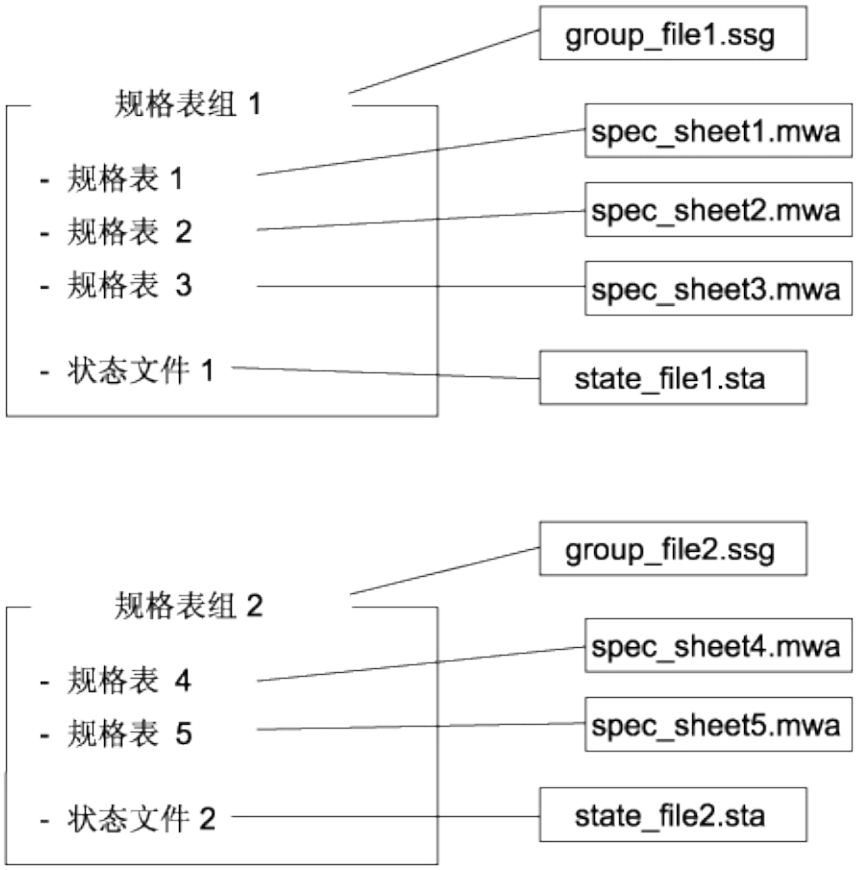
-
-
-
-
-

MWA

“Group”

-
- E5071C

ssg E5071C



e5070c005

- 10

MWA 10 10

-

MWA

- E5071C

E5071C E5071C

E5071C E5071C

“ E5071C ”“”E5071C

E5071C E5071C

- E5071C

-

MWA

“ **Save State**” E5071C “”

1.

- E5071C
-
- “ **Save State**”

2.

- E5071C
-

//

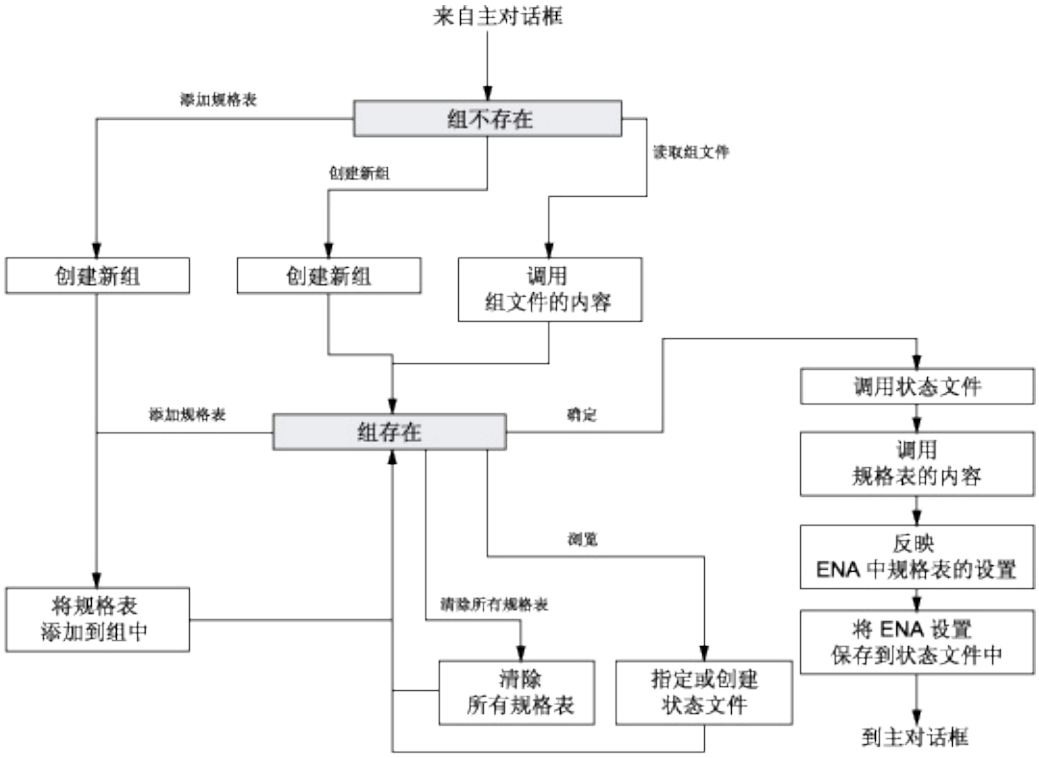
1.
2.
3. E5071C
4.
5. E5071C
6. E5071C

- E5071C /
-

E5071C / E5071C

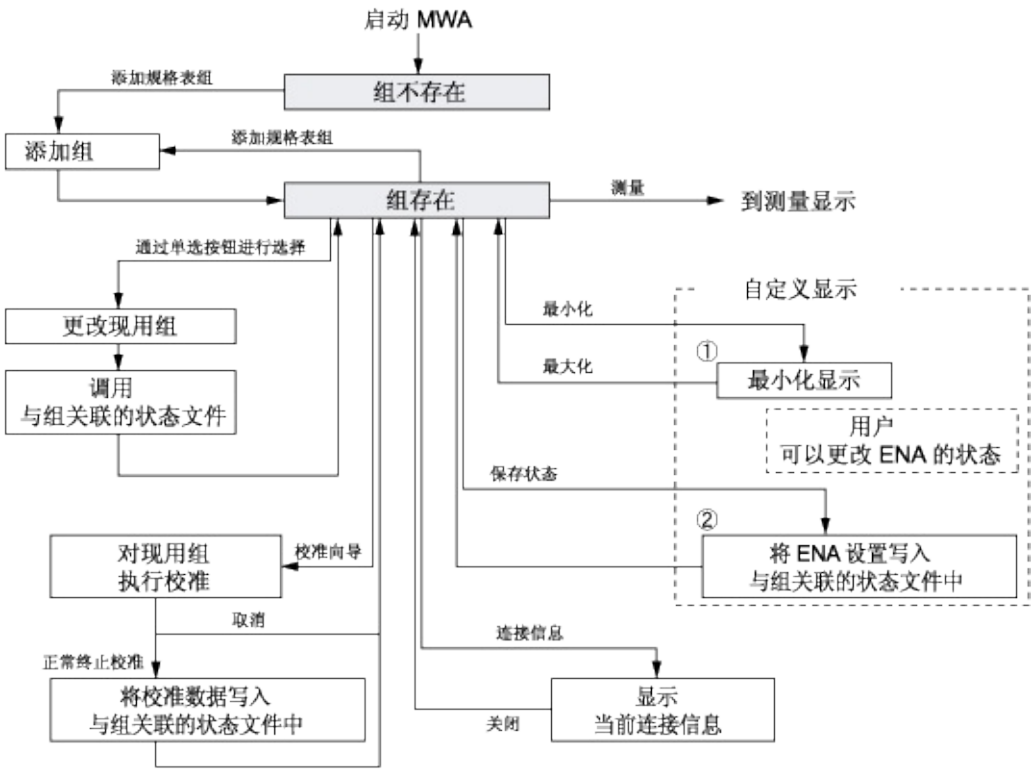
6 1 “”

3 4 ENA



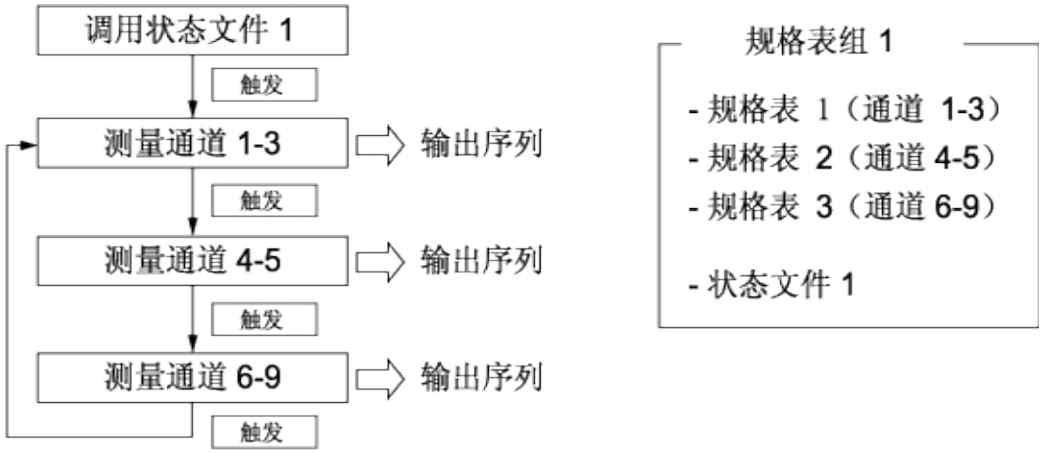
e5070c006

“ **OK**”“” E5071C



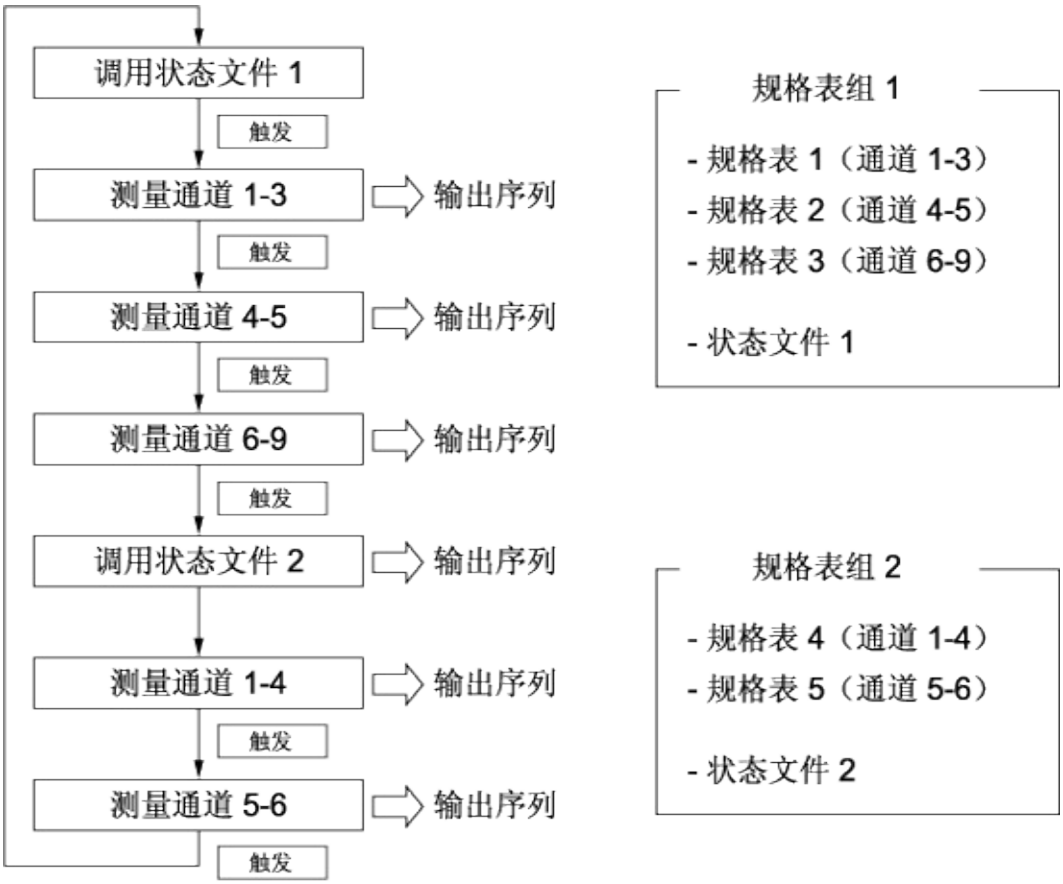
e5070c007

1.
2. “ **Minimize**”“ **Save State**”
3.
4.
5. DUT


调用状态文件 1
触发
测量通道 1-3
输出序列
触发
测量通道 4-5
输出序列
触发
测量通道 6-9
输出序列
触发
规格表组 1
- 规格表 1（通道 1-3）
- 规格表 2（通道 4-5）
- 规格表 3（通道 6-9）
- 状态文件 1

e5070c008

1



e5070c009

- 1 6-9 MWA “” 2
- 2 5-6 MWA “” 1

/

4 5 2 2 1

MWA

PC

- Windows2000Windows XP
- Microsoft Excel 2000 Microsoft Office 2000 Excel

E5071C

- MWA

ENA MWA

E5091A

- 9 (E5091A-009) 16 (E5091A-016)
- 13 16 (E5091A-016)

MWA

-
-
-
-

MWA

MWA PC E5071C

MWA PC

1. http://www.agilent.com/find/mwa/ MWA
2. MWA
3. **mwa_xxxx.xlsmwa_xxxx.xls x mwa_0106.xls** PC

E5071C 790 MWA **D:\VBA**

1. http://www.agilent.com/find/mwa/ MWA
2. MWA
3. USB LAN **mwa_xxxx.vba** E5071C **D:\VBA**

mwa_xxxx.xls **mwa_xxxx.vba**

MWA

MWA

1. MWA
2. MWA “ **AGREE**”“ **DO NOT AGREE**” E5071C
3. “Do you have a key code?”“ **Yes**”“ **NO**”
4. E5005A “ **OK**”“ **Cancel**”“ **OK**”
5. MWA

- 100
- 500

MWA MWA MWA E5005A MWA

e5071c252

-
- MWA
-
- DUT
-
- GPIB
-
-
- /

MWA

DUT

MWA

1. Excel“ **File**”>“ **Open**”
2. “ **mwa_xxxx.xls**”“ **Open**”
3.

“Multiport Test Set”

1. “Multiport Test Set” (1)
2. 1 (2)

0	None	E5071C
1	E5091A 9-Port9 E5091A	E5091A 9 (E5091A-009)
2	E5091A 13-Port13 E5091A	13 E5091A 16 (E5091A-016)
3	E5091A 16-Port16 E5091A	E5091A 16 (E5091A-016)
4	Z5623AK64	PNA E5071C
5	Z5623AK66	PNA E5071C

3. 1 “Label” (3)
4. 2 (4)

0	None	
1	E5091A 9-Port9 E5091A	E5091A 9 (E5091A-009)
	E5091A 13-Port13	13 E5091A 16

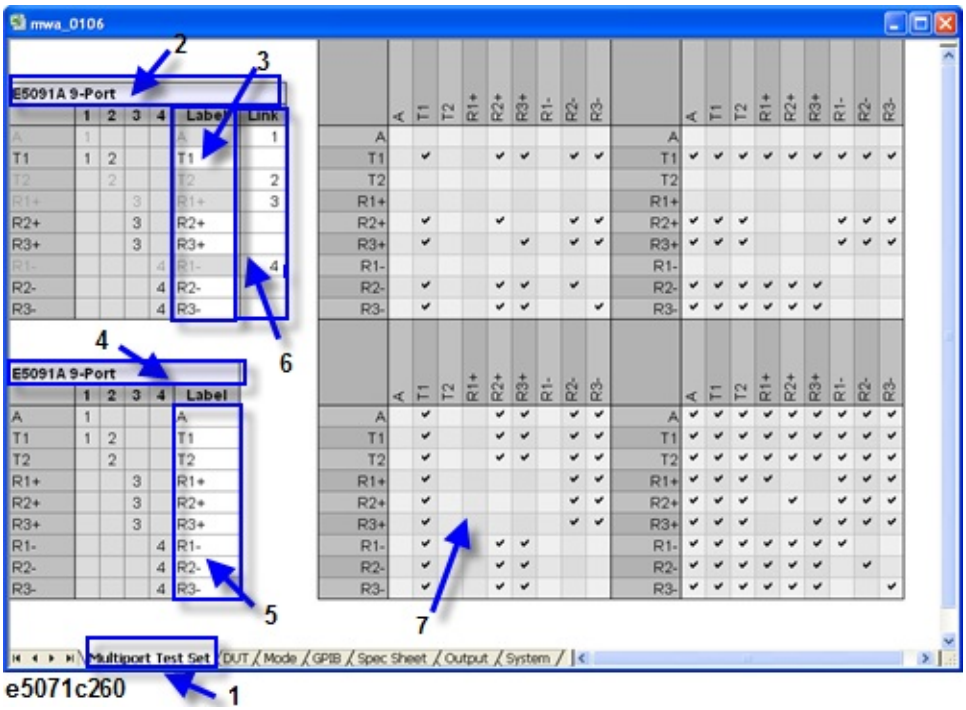
2	E5091A	(E5091A-016)
3	E5091A 16-Port16 E5091A	E5091A 16 (E5091A-016)

4 6 “None” 7

5. 2 “Label” (5)
6. “Link” (6) 1 4 1 4
7. (7)

A-T1 A-T2

MWA



e5071c260

DUT

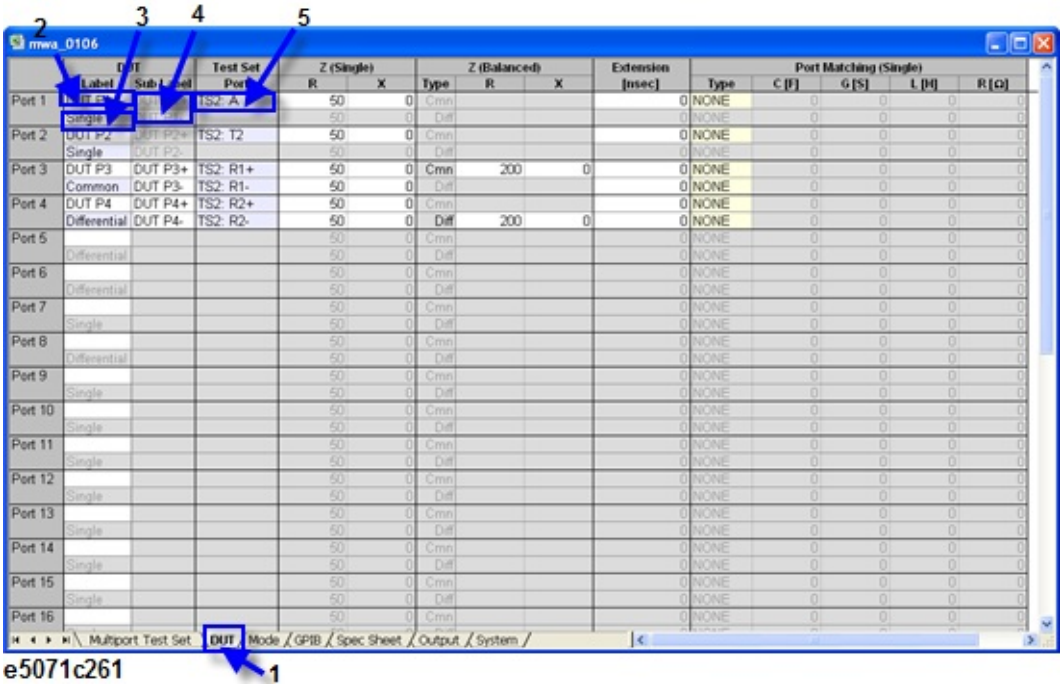
“DUT” DUT

1. “DUT” DUT (1/3) (1)
2. DUT“Label” DUT (1/3) (2) DUT
3. DUT“Label” DUT (1/3) (3)1 3

1	Single	“Single”
2	Common	“Common”
3	Differential	“Differential”

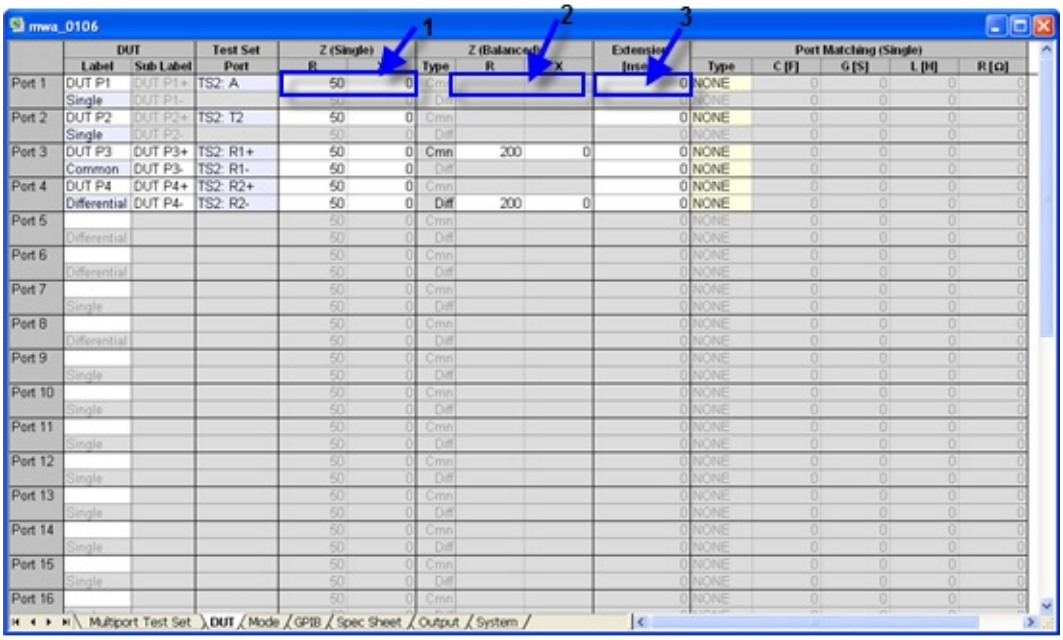
4. “Common”“Differential”“Sub Label” DUT (1/3) (4)
5. DUT (1/3) (5)

DUT (1/3)



6. “Single”“Z (Single)”Z“R”“X” DUT (2/3) (1)“Common”“Differential”“Z (Balanced)”Z“R”“X” DUT (2/3) (2)
7. “Extension” DUT (2/3) (3)“Single”

DUT (2/3)



e5071c262

8. “Port Matching (Single)”“Type” DUT (3/3) (1) CGL R “ **OK**” “ **Cancel**” (2)

CGL R FSHΩ

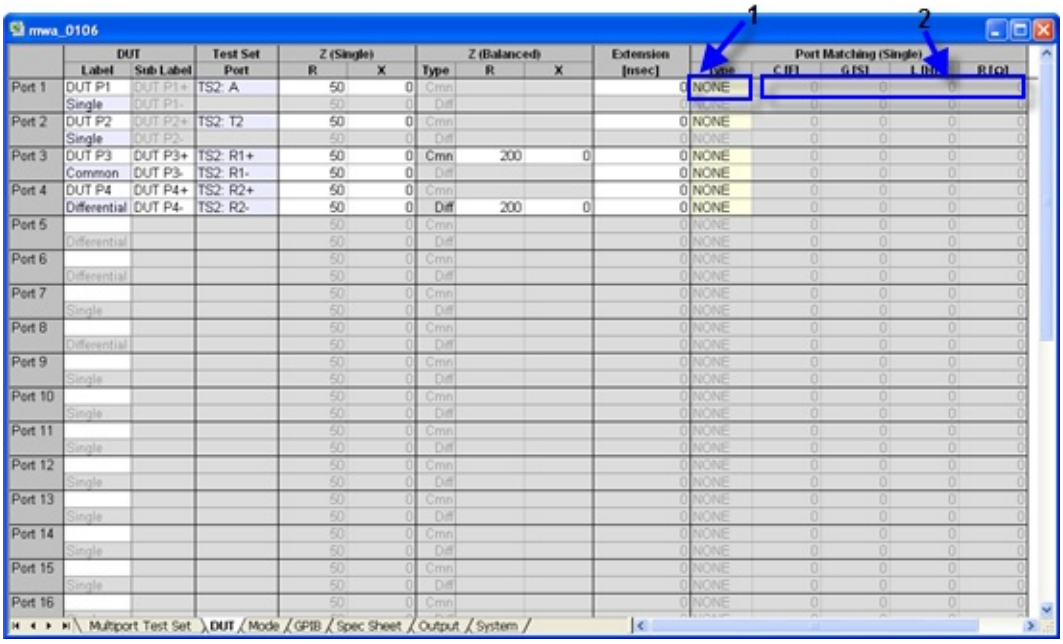
None
SLPC
PCSL
PLSC

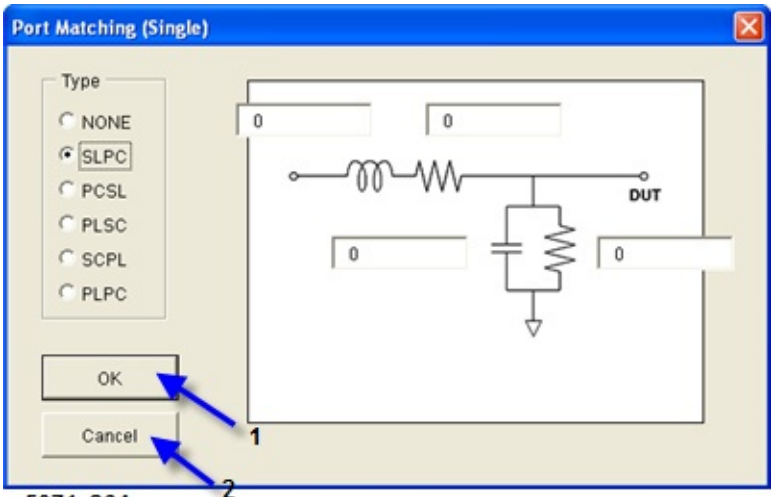
SCPL
PLPC

“C”“G”“L”“R” DUT (3/3) (2)

9. 2 8

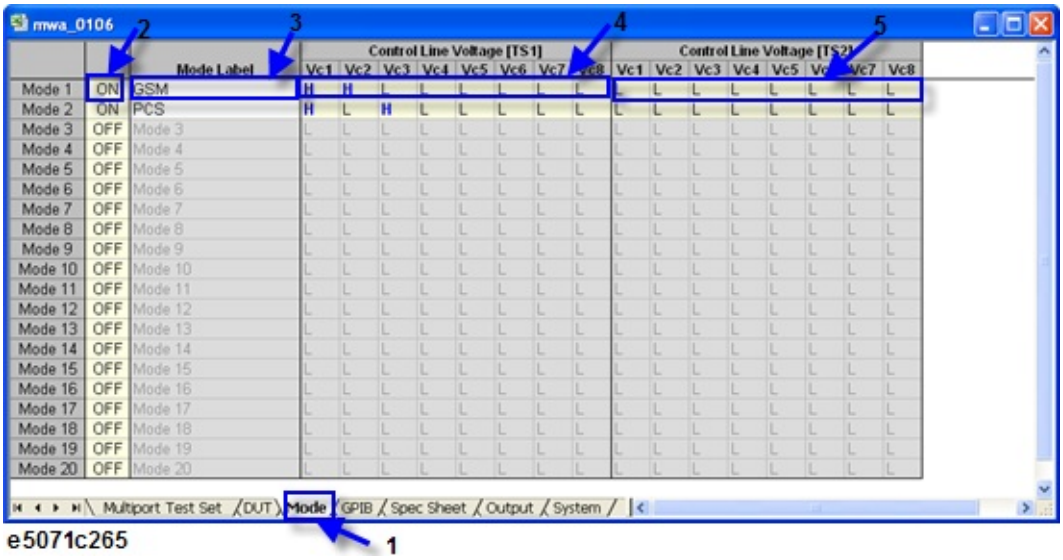
DUT (3/3)




Port Matching (Single)
Type
NONE
SLPC
PCSL
PLSC
SCPL
PLPC
0
0
DUT
0
0
OK
Cancel
1
2
e5071c264

“Mode”

1. “Mode” (1)
2. (2)“ON”
3. “Mode Label” (3)
4. “Control Line Voltage [TS1]” [TS1] Vc1 Vc8 (4)
   L LowH HIGH 0 10 1
5. 4 “Line Voltage [TS2]” [TS2] Vc1 Vc8
   (5)
6. 2 5 20



GPIB

“GPIB” GPIB INIT

1. “GPIP” GPIB (1)
2. “GPIB Command”GPIB “Address” GPIB (3)
   GPIB GPIB

- “Mode” GPIB (2)“Delete”

  “Command Count”

- “Address” -1 GPIB E5071C

3. “GPIB Command”GPIB “Command” GPIB
   (4) GPIB
4. “GPIB Command”GPIB “Wait [ms]” [ms] GPIB
   (5) GPIB ms
5. “GPIB Command”GPIB “Type” GPIB (6)

1	Limit	GPIB
2	Pre	GPIB
3	Post	GPIB

- GPIB INIT GPIB
- INIT

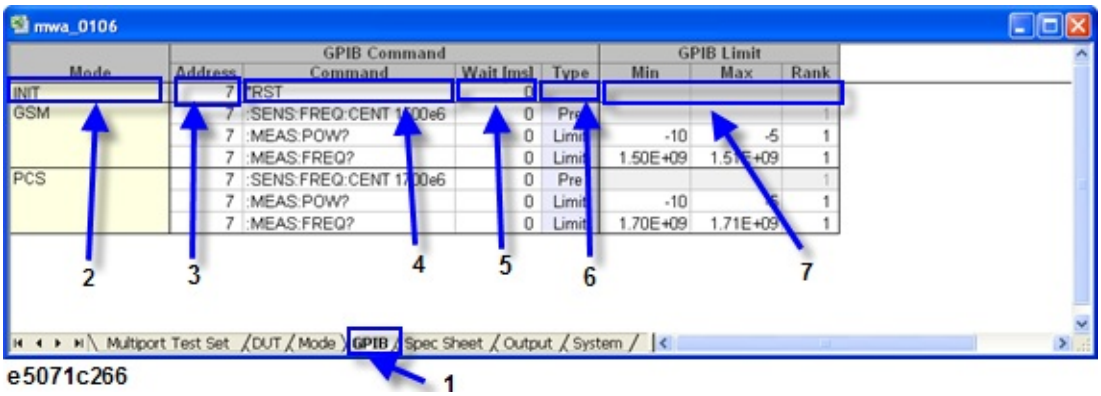
6. GPIB “Limit”“GPIB Limit”GPIB “Min”

“Max”“Rank” GPIB (7) GPIB
“Min”“Max”“Rank”

7. GPIB “Limit”“GPIB Limit”GPIB
“Pre”“Post” GPIB INIT GPIB

7. 2 6 GPIB

GPIB



e5071c266

1. “Spec Sheet” (1/2) (1)
2. “Path”“From” (1/2) (3) DUT

“Mode” (1/2) (2)“Delete Record”

“Path Count...”...“Path Count”

3. 2 “Path”“To” (1/2) (3)
4. “Type” (1/2) (4)

1	IL		LogMag	Inversion	
2	Ripple		LogMag		
3	VSWR		VSWR		
4	RL		LogMag	Inversion	
5	ATT		LogMag	Inversion	
6	Isolation		LogMag	Inversion	
7	BalAmp		LogMag		
8	BalPhase		Phase		

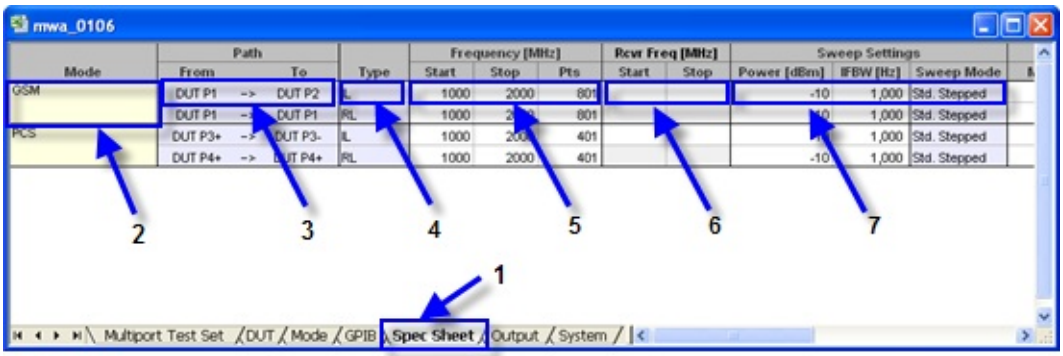
“Limit”“Inversion” 8 IL “Max”
5 “ 5” S -5

2

	RLVSWR
	ILRippleATTIsolation
	ILRippleATTIsolation BalAmpBalPhase

ILATT Isolation

(1/2)



e5071c267

5. “Frequency [MHz]” [MHz]“Start”“Stop” “Pts” (1/2) (5) E5071C
6. FOM “Rcv Freq [MHz]” [MHz]“Start” “Stop” (1/2) (6) E5071C
7. “Sweep Setting”“Power”“IFBW [Hz]”IF [Hz]“Sweep Mode” (1/2) (7)IF IF

1	Std.Stepped	
2	Std.Swept	

8. “Min”“Max” (2/2) (1)“Min”
   -999“Max” 999

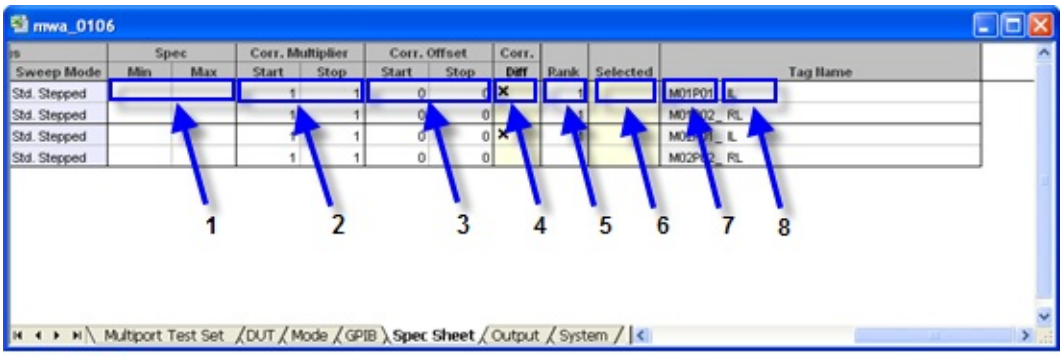
9. “Corr. Multiplier”“Start”“Stop” (2/2)
   (2)“Corr. Offset”“Start”“Stop”
   (3) Ax+BA B

10. “Corr.”“Diff” (2/2) (4)

    dBc

-
-
-
-

(2/2)



e5071c268

11. “Rank” (2/2) (5) 1 255 0

    I/O

12. “Selected” (2/2) (6)

    “Selected”

13. “Tag Name” (2/2) (7) (8) (2/2) (7) (2/2) (8)

14. 2 13

“Output” MWA

1. “Output” (1)

“ **Yes**”

2. “Max Channel” (4)“Max Trace” (5)
   “Max Point” (6)

3.

3. “Output” (2)
4. (3)
5. “ **Save Spec Sheet**” (7)“Save”
   “ **Save**”MWA

mwa_0106

	A	B	C	D	E	F	G	H	I	J
1	Save Spec Sheet		Max Channel	Max Trace	Max Point					
2			32	8	1601					
3										
4	Mode	Tag	Channel	Trace	Segment	Parameter				
5	GSM	M01P01_IL	1	1	1	S21				
6		M01P02_RL	1	2	1	S11				
7	PCS	M02P01_IL	2	1	1	S43				
8		M02P02_RL	3	1	1	S33				
9										
10	Channel 1									
11			Test Set	Port	Freq Mult	Freq Offs				
12		Port 1	2	1						
13		Port 2	2	3						
14										
15			Parameter	Format						
16		Trace 1	S21	MLOGarithmic						
17		Trace 2	S11	MLOGarithmic						
18										
19			Start	Stop	Points	Power	IFBW	Sweep Mode		
20		Segment 1	1000000000	2000000000	801	-10	1000	STEPped		
21										
22	Channel 2									
23			Test Set	Port	Freq Mult	Freq Offs				
24		Port 3	2	4						
25		Port 4	2	7						
26										
27			Parameter	Format						
28		Trace 1	S43	MLOGarithmic						
29										
30			Start	Stop	Points	Power	IFBW	Sweep Mode		
31		Segment 1	1000000000	2000000000	401	-10	1000	STEPped		
32										
33	Channel 3									
34			Test Set	Port	Freq Mult	Freq Offs				
35		Port 3	2	5						
36										
37			Parameter	Format						
38		Trace 1	S33	MLOGarithmic						
39										
40			Start	Stop	Points	Power	IFBW	Sweep Mode		
41		Segment 1	1000000000	2000000000	401	-10	1000	STEPped		
42										

Multiport Test Set / DUT / Mode / GPIB / Spec Sheet / Output / System

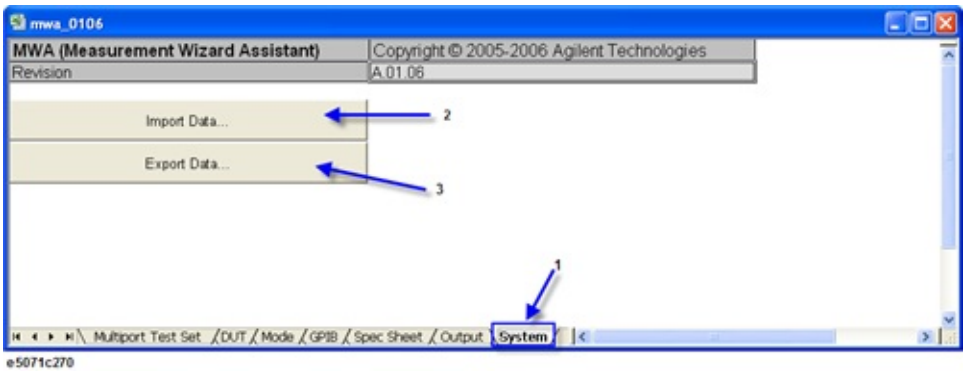
1 2 3 4 5 6 7

e5071c269

/

“System” csv MWA

1. “System” (1)
2. csv “ **Export**” (3)“Export Data”
3. csv “ **OK**” csv
4. csv “ **Import**” (2)“Import Data”
5. cvs “ **OK**”



E5071C VBA

-
- MWA
-
-
-
-
-
-

MWA

MWA E5071C VBA MWA E5091AE5071C
DUT

MWA

E5071C MWA

1. “ **Macro Setup**”
2. “ **Load Project...**”...“”Open
3. “ **mwa_xxxx.vba**”“ **Open**”
4. “ **Select Macro**”>“ **Module1 Main**” 1
5. MWA

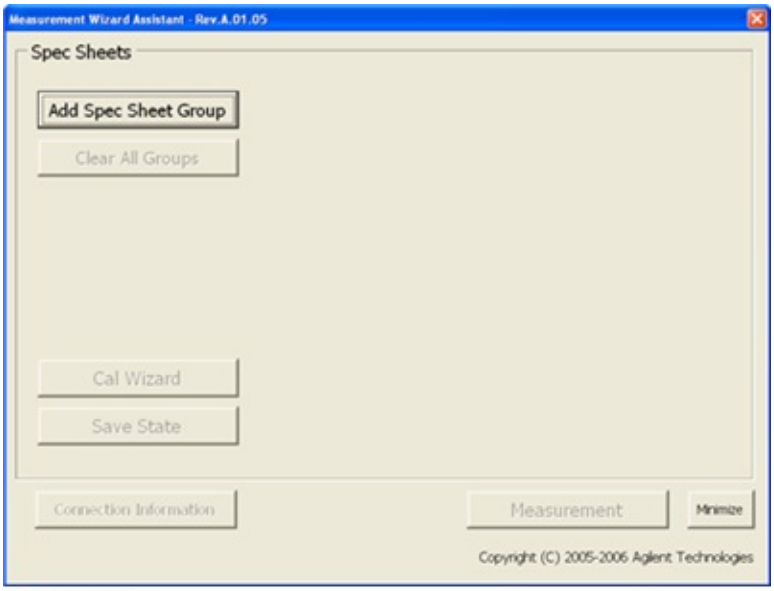
- **mwa_xxxx.vba** **D:\VBA** “ **Load & Run**” >“**mwa_xxxx**”

MWA MWA “Calibration Wizard”“Lot Input”“File Open/Save”/

1.
2. “Spec Sheet Setting”
3. “Connection Information”
4. “Calibration Wizard”
5. “All Spec Sheets Measurement”
6. “Measurement by Spec Sheet”
7. “Results Verification by Mode”

1.

“Connection Information”“Calibration Wizard”“All Spec Sheets Measurement”



e5071c254

- “**Add Spec Sheet Group**”

- “**Clear All Spec Groups**”

- “**Cal Wizard**”

(1/2)

- “**Save State**”

E5071C

“Minimize” E5071C

E5071C

- “**Connection Information**”

- “**Measurement**”

- “**Minimize**”

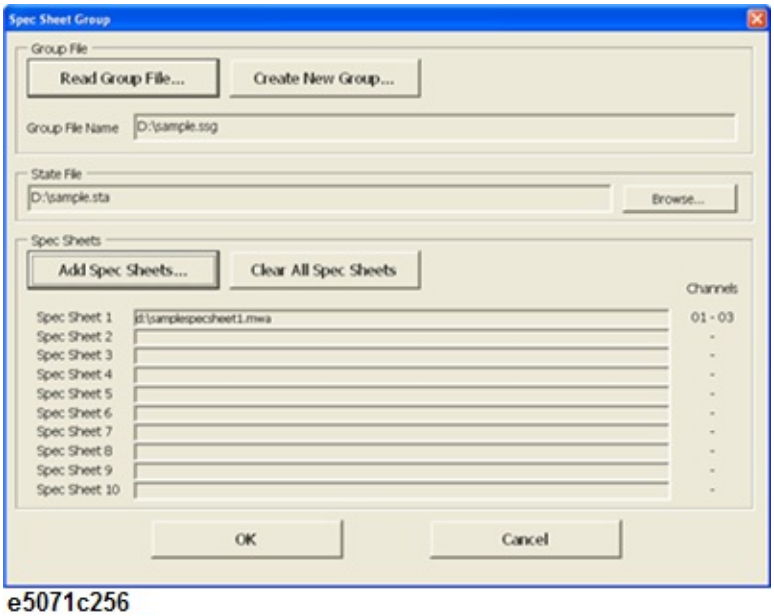
MWA

-

(l)

2. “Spec Sheet Setting”

E5071C



e5071c256

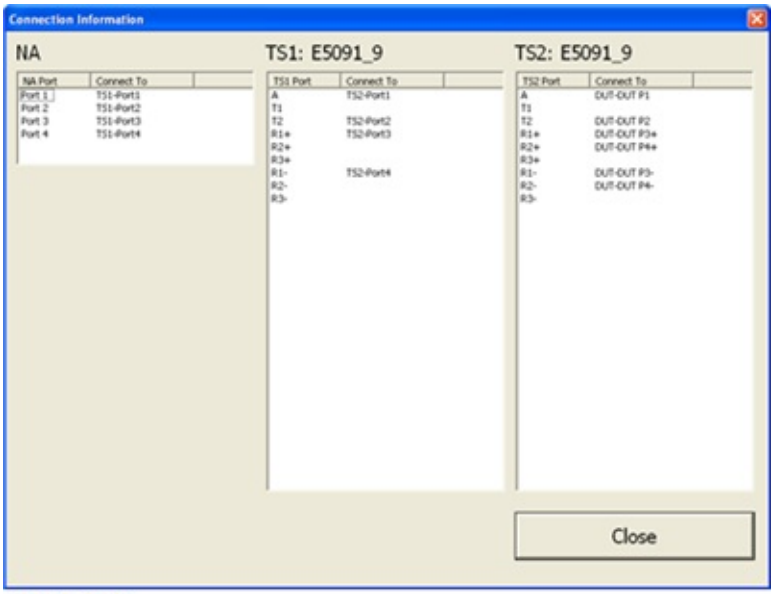
- “**Read Group File...**”…
- “**Create New Group...**”…
- “**Browse...**”…
- “**Add Spec Sheets...**”…
- “**Clear All Spec Sheets**”
- “**OK**”

- “**Cancel**”

3. “Connection Information”

“ **Connection Information**”

E5071C 1 2 DUT



e5071c257

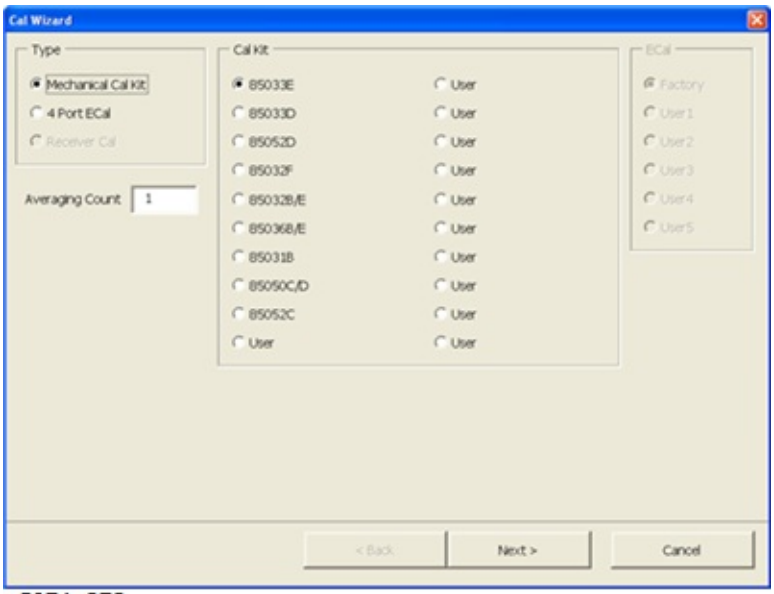
- “**Close**”

4. “Calibration Wizard”

“ **Cal Wizard**”

E5071C

(1/2)



e5071c272

-

  “Mechanical Cal Kit”“4-Port ECal”4 4
  ECal “Receiver Cal”FOM

-

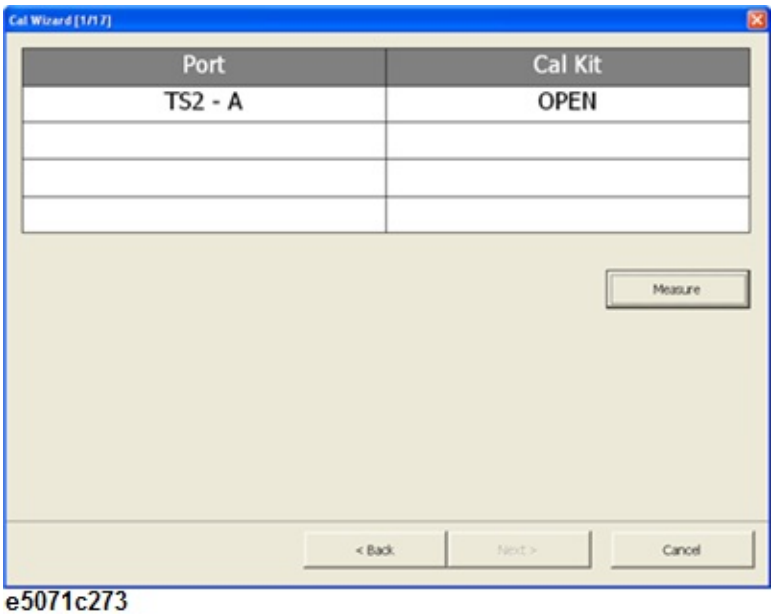
  “Mechanical Cal Kit”

- ECal

  E5071C ECal “4-Port ECal”4

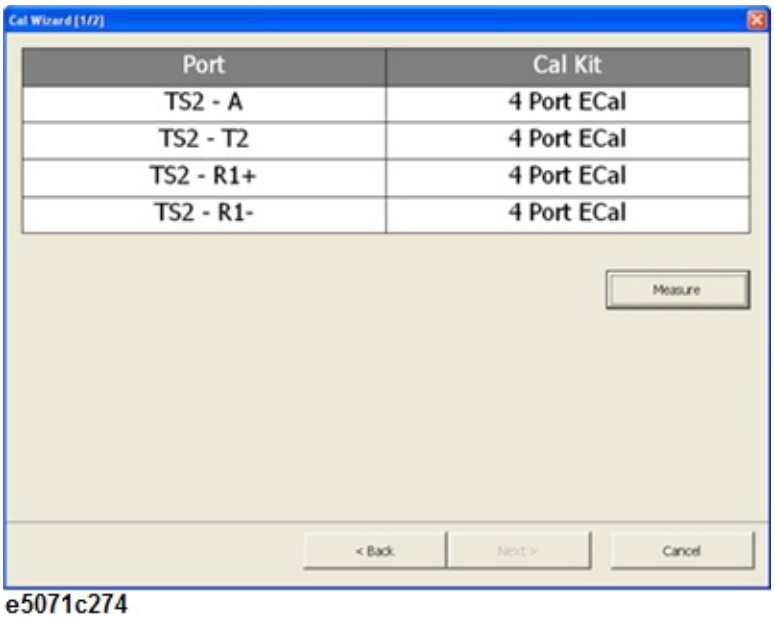
-

(2/2)



e5071c273

(2/2) ECal



e5071c274

(2/2) “Port”“CalKit“”“”

- “**Measure**”

- “**<Back**”<

- “**Next >**” >

- “**Cancel**”

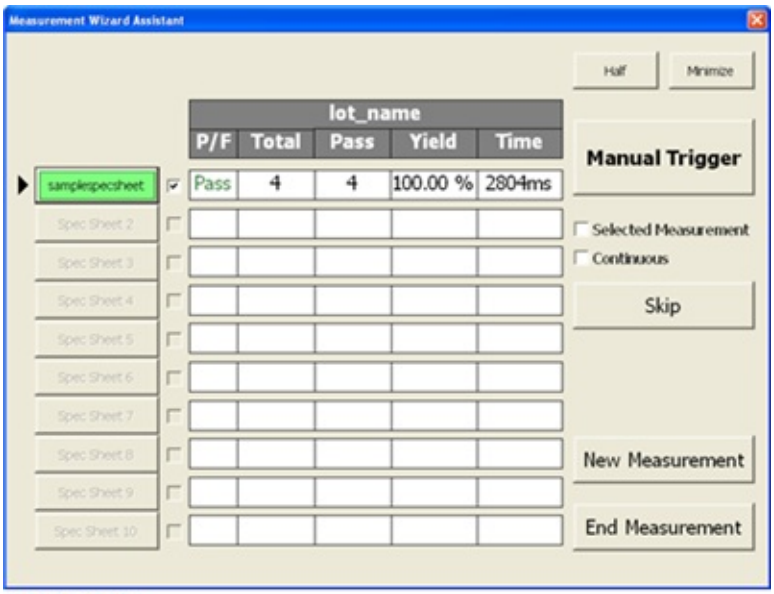
- “**Done**”“ **Cancel**”

5. “Measurement”“All Spec Sheets Measurement” “Measurement by Spec Sheet”“Results Verification by Mode”

“All Spec Sheets Measurement”

“ **Measurement**”“Lot Input”

“ **Manual Trigger**”



e5071c258

- “**Spec Sheet**”

- “Measurement ON/OFF”/

/

- “**Half**”

MWA E5071C “Half”

- “**Minimize**”

MWA

- “**Manual Trigger**”

“ **Spec Sheet**”

- “**Skip**”

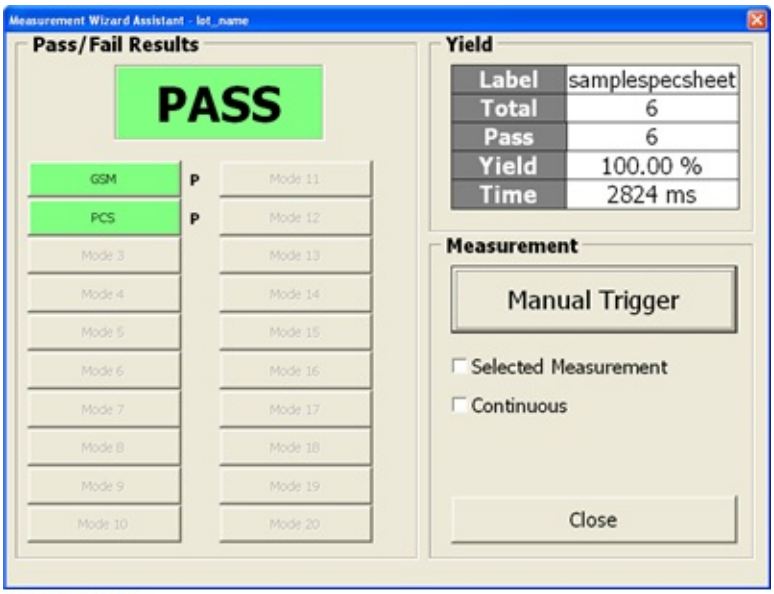
- “**New Measurement**”

- “**End Measurement**”

“Measurement by Spec Sheet”

“All Spec Sheets Measurement”

“ **Manual Trigger**”



e5071c259

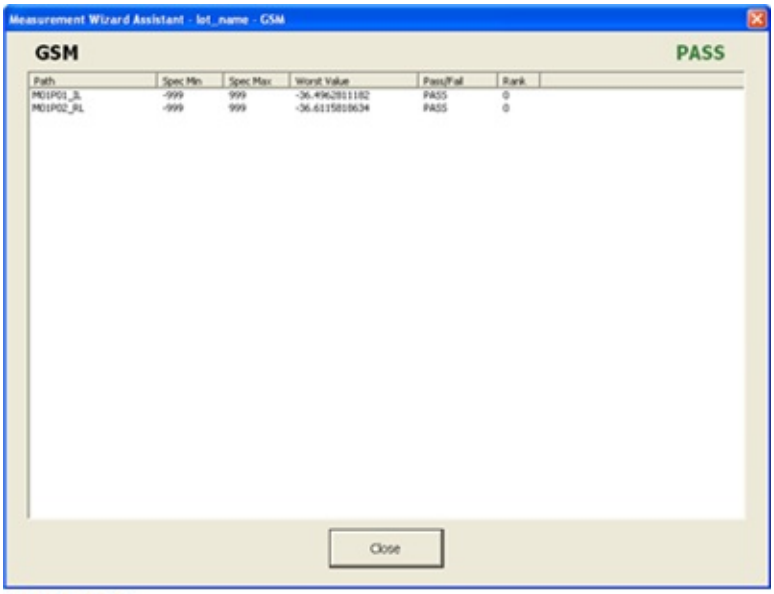
- "**Manual Trigger**"

  "All Spec Sheets Measurement"

- "**Close**"

"Results Verification by Mode"

"Measurement by Spec Sheet"



e5071c277

-

“Path”

“Spec Min”/“Spec Max”

“Worst Value”

“Pass”/“Fail”“Pass”“Fail”

“Rank” 0 200

DUT	/	
0		0
1 - 200		1 - 200

201 - 255		201 - 255

DUT “the result is pass, however, the Rank is not 0”

/

- “**Close**”

6.

“ **half**”“Measurement”

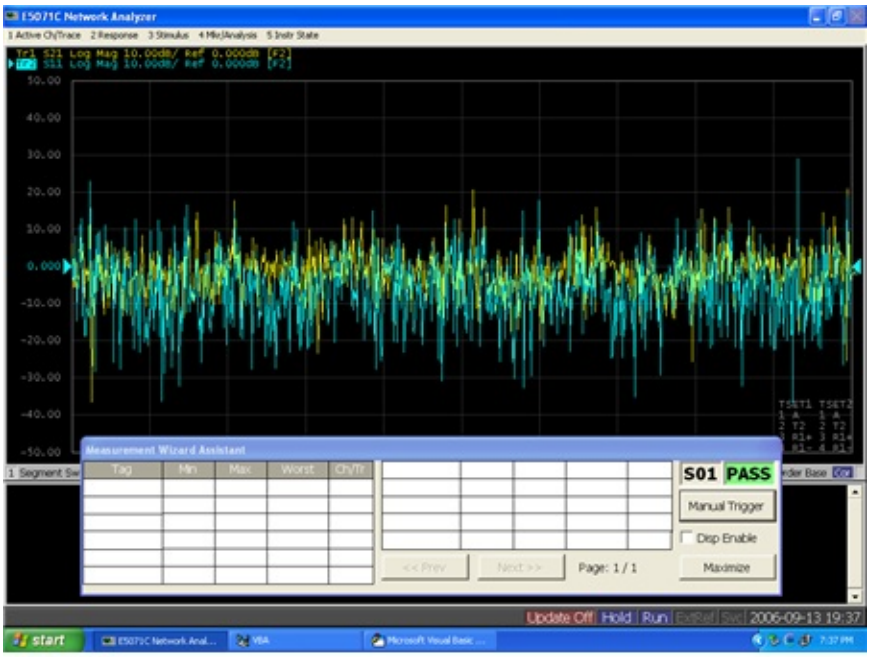
ENA MWA

“ **<<Prev**”<< “ **Next>>**” >>

“ **Half**”“ **Save State**”

“Disp Enable” E5071C “Half” E5071C
E5071C

“Half”



e5071c275

e5071c271

“Spec Sheet Setting”

“Spec Sheet Setting”

1. “ **Add Spec Sheet Group**”“Spec Sheet Setting”

2. “Group File”“ **Create New Group...**”...
   “ **Save**”

   “Group File”“ **Read Group File...**“...
   “ **Open**”

   “Group File Name”

3. “State File”“ **Browse...**”...“ **Open**”

   “State File”

4. “Spec Sheets”“ **Add Spec Sheets...**”...
   “ **Open**”

“Spec Sheets”

- 4
- “ **OK**”

5. “ **OK**”

   “ **Clear All Groups**”“ **Cal Wizard**”“ **Save State**”“ **Connection Information**”

6. “ **Clear All Groups**”

1. “ **Clear All Groups**”

2. “ **Clear All Groups**”“ **Cal Wizard**”“ **Save State**”“ **Connection Information**”
“ **Measurement**”

“ **Clear All Groups**”

1. “ **Clear All Spec Sheets**”
2.

“ **Clear All Spec Sheets**”

“Connection Information”

1. “ **Connection Information**”
2. “Connection Information” E5071CE5091A DUT
3. “ **Close**”“Connection Information”

E5071C DUT

E5071CE5091A DUT

“Calibration Wizard” (1/2) (2/2) (2/2) ECal

1.
2. “ **Cal Wizard**”
3. “Calibration Wizard” (1/2)“Type”

Mechanical Cal Kit	4
4 Port ECal4	4 ECal
Receiver Cal	FOM

4. 3 “Mechanical Cal Kit”“Cal Kit”

3 4 ECal “ECal” ECal

E5071C E5071C

5. “Averaging Count”“ **Next**”
6. “Calibration Wizard” (2/2) ECal ECal
   “ **Measure**”“ **Next >**” >
7. 6 “ **Done**”

MWA

1. “ **Save State**”
2.

“All Spec Sheets Measurement”

“All Spec Sheets Measurement”

1. “ **Measurement**”
2. “ **OK**”

- Output Start/Stop Values/

- Correction (Limit Test)

- Enable Manual Measurement

“All Spec Sheets Measurement”“Measurement by Spec Sheet”“ **Manual Trigger**”“ **Manual T**

- Overwrite Output File

- Max fail rank (1-255) (1-255)

“Max fail rank” 10 1 10 11 255

“ **OK**”“All Spec Sheet Setting”

3. “ **Skip**”
4. “ **Manual Trigger**”

“ **Selected Measurement**”“ **Manual Trigger**”

“ **Continuous**”

5.

“Measurement by Spec Sheet”“Measurement by Spec Sheet”

“Measurement by Spec Sheet”

“Measurement by Spec Sheet”

1. “All Spec Sheets Measurement”“Measurement by Spec Sheet”
2. “ **Manual Trigger**”

“ **Selected Measurement**”“ **Manual Trigger**”

“ **Continuous**”

3. “P”“F”“ **Close**”“All Spec Sheets Measurement”

“Results Verification by Mode”“Results Verification by Mode”

“Results Verification by Mode”

“Results Verification by Mode”

1. “Results Verification by Measurement Mode” “Measurement by Spec Sheet”
2. GPIB “Results Verification by Measurement Mode” “ **Close**”“Measurement by Spec Sheet”

1.

csv

“”.csv

1 1 2 2 ... n n

2.

csv

“”-“”.csv

/ON)

"Index","Rank","mode1-path1",,,"mode1-path2",,,"[GPIB**] gpib_command", ...,"modeX-pathY",,

,,"Start","Stop","Worst", ...,"Meas", ...,"Start","Stop","Worst", ...

1,result_rank,start_val,stop_val,worst_val, ...,gpib_response_val, ...,start_val,stop_val,worst_val, ...

2,result_rank,start_val,stop_val,worst_val, ...,gpib_response_val, ...,start_val,stop_val,worst_val, ...

...

N,result_rank,start_val,stop_val,worst_val, ...,gpib_response_val, ...,start_val,stop_val,worst_val, ...

“mode1-path1”“mode1-path2”GPIB
“GPIB**]gpib_command”“**” GPIB DUT
CSV DUT

DUT - GPIB CSV x3 GPIB

//

-1

“Output Start/Stop value”/

/OFF

"Index","Rank","mode1-path1",,,"mode1-path2",,,"
[GPIB**]gpib_command", ...,"modeX-pathY",,

1,result_rank,worst_val, ...,gpib_response_val, ...,worst_val, ...

2,result_rank,worst_val, ...,gpib_response_val, ...,worst_val, ...

...

N,result_rank,worst_val, ...,gpib_response_val, ...,worst_val, ...

PC

PC

1. GPIB E5071C

DISP:WIND1:TITL:DATA cmd_para

cmd_para

cmd_para	**1**	**2**	
cmd_read	1 2...		
cmd_clear	-		
cmd_begin		/	“Measurement”
cmd_initiate	-		“Measurement”
cmd_skip	-		“Measurement”
cmd_settings			“Measurement”
cmd_end	-		“Measurement”

cmd_para

cmd_para		
cmd_read	1 2 1	**Add Spec Sheet Group**
cmd_clear		**Clear All Groups**
cmd_begin	“All Spec Sheets Measurement” 1 2 2 _____ 2 • Output Start/Stop Values/ “Start/Stop values”/ 0 1 • Correction values “Correction (Limit Test)” 0 1 • Manual Measurement “Enable Manual Measurement” 0 1 • Overwrite Output File	**Measurement** “All Spec Sheets Measurement” **New Measurement**

	“Overwrite Output File” 0 1 • Max Fail Rank “Max fail rank (1-255)” (1-255) 1 255 “Measurement” “All Spec Sheets Measurement”	
cmd_initiate		“All Spec Sheets Measurement” **Manual Trigger** “Measurement by Spec Sheet” **Manual Trigger**
cmd_skip		“All Spec Sheets Measurement” **Skip**
cmd_settings	cmd_begin	
cmd_end		“All Spec Sheets Measurement” **End Measurement**

2. 1 (,)

- 1

DISP:WIND2:TITL:DATA para

- 2

DISP:WIND3:TITL:DATA para

“lot_gsm001”“Output start/stop value”/
“overwrite output file”“Max fail rank” 100

10 output 717;"DISP:WIND2:TITL:DATA lot_gsm001"
20 output 717;"DISP:WIND3:TITL:DATA 1,0,0,1,100"
30 output 717;"DISP:WIND1:TITL:DATA cmd_begin"

DISP:WIND1:TITL:DATA? “cmd_done”
DISP:WIND1:TITL:DATA? “cmd_done”

I/O 24 I/O

MWA E5071C I/O 24 I/O

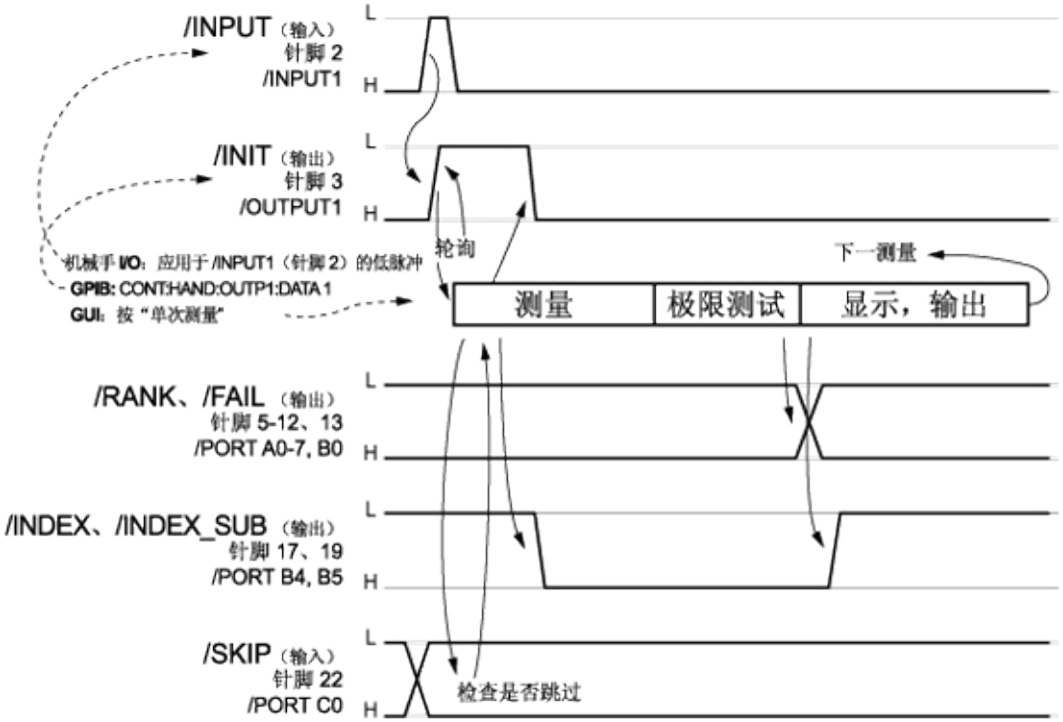
MWA E5071C

2	/INPUT (/INPUT1)		/OUTPUT1 E5071C
3	/INIT (/OUTPUT1)		2 /OUTPUT1 MWA /OUTPUT1 /OUTPUT1 “CONT:HAND:OUTP1:DATA 1”
			/PORT A0 - A7

5	/RANK A0 (/PORT A0)		8 A0 LSBA7 MSB /PORT A0 0
6	/RANK A1 (/PORT A1)		1
7	/RANK A2 (/PORT A2)		2
8	/RANK A3 (/PORT A3)		3
9	/RANK A4 (/PORT A4)		4
10	/RANK A5 (/PORT A5)		5
11	/RANK A6 (/PORT A6)		6
12	/RANK A7 (/PORT A7)		7
13	/FAIL (/PORT B0)		/PORT B0 (/PORT A0 - /PORT A7) 0 /PORT B0
17	INDEX (/PORT B4)		(/PORT A0 - /PORT A7) /PORT B4 /PORT B4

19	/INDEX_SUB (/PORT B5)		/PORT B4
22	/SKIP (/PORT C0)	/	/PORT C0

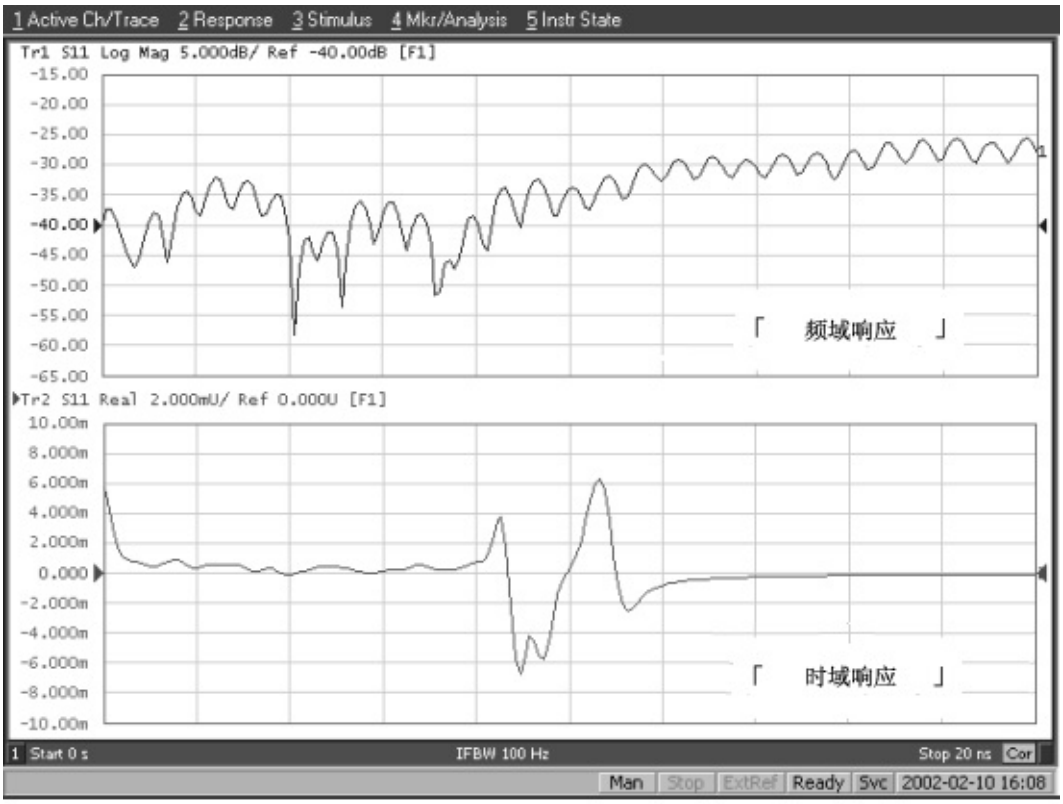
MWA I/O



e5070c038

-
- (TDR)
- E5071C

E5071C 010



e5070aue202

(TDR)

E5071C (TDR)

TDR DUT

E5071C DUT

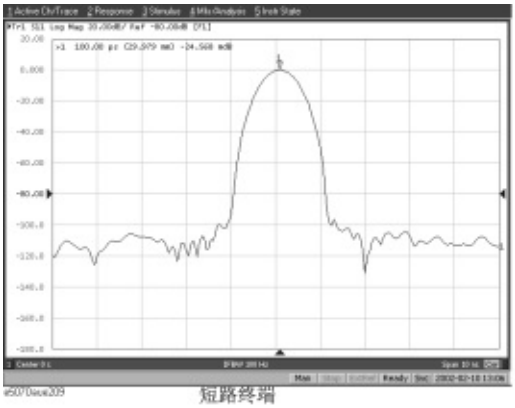
E5071C

E5071C

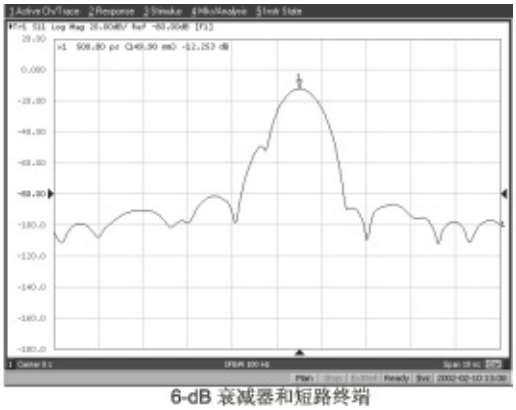
-
-
-

50% 50% 50% 50%
25%

6-dB - 12 dB

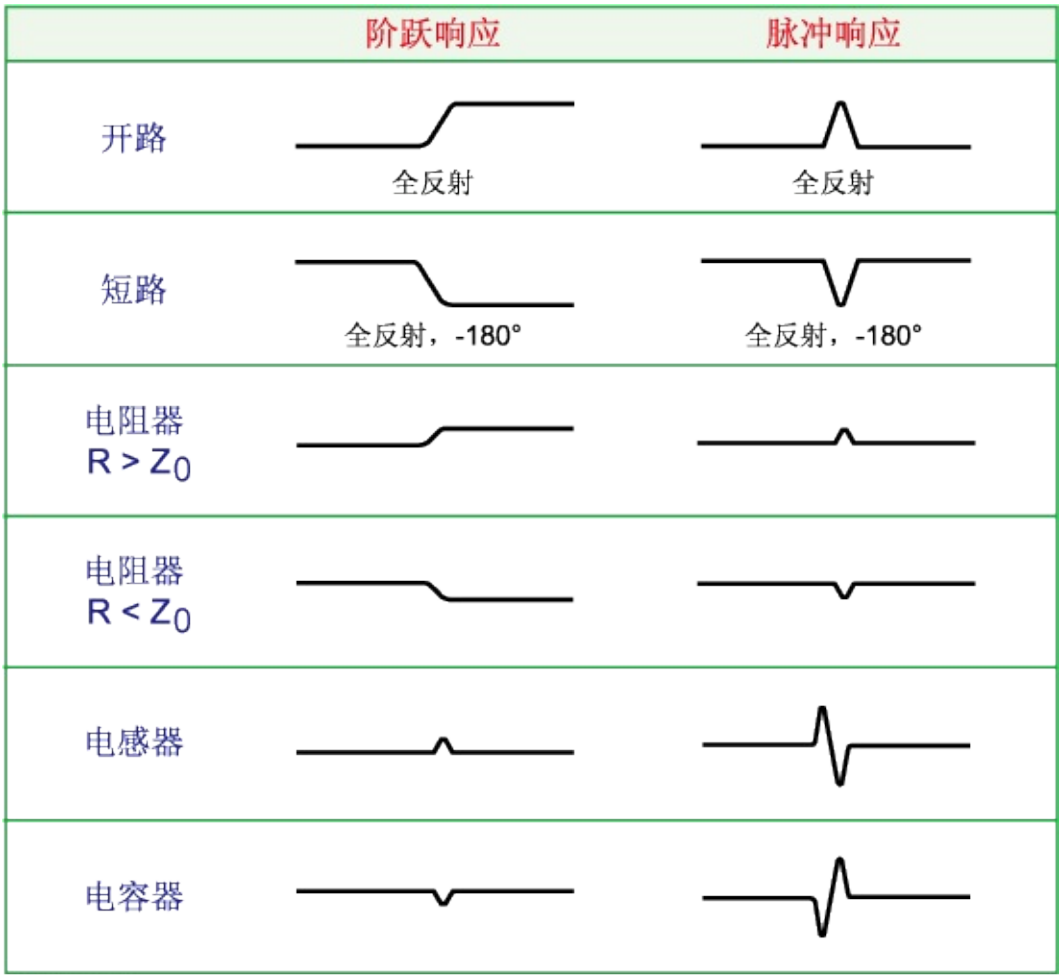


短路终端



6-dB 衰减器和短路终端

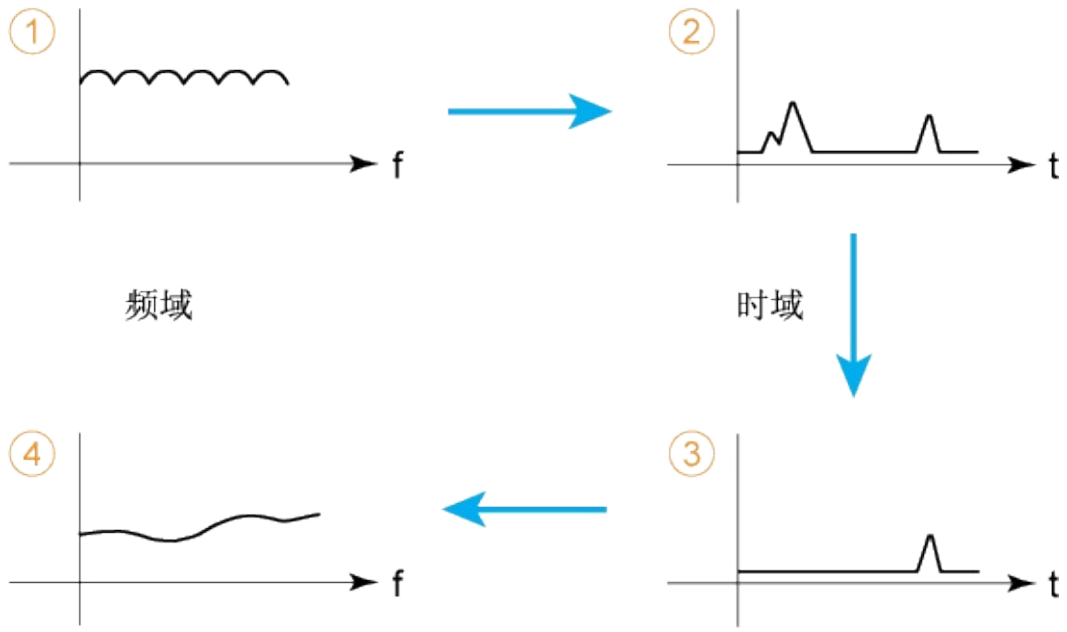
TDR



e5071c415

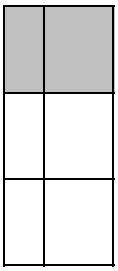
-
-
-
-
-

1.	
2.	
3.	• • •
4.	

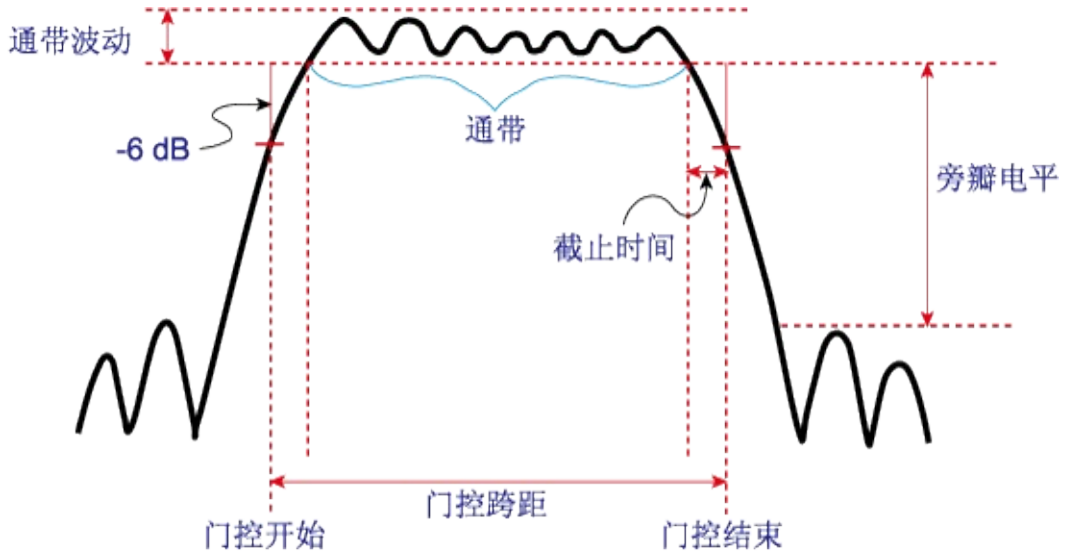


e5071c414

E5071C



1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“ **Gating**”“Gating”
3. “ **Type**”“ **Bandpass**”“ **Notch**”



e5071c416

“”“” 2

	± 0.13 dB	- 48 dB	1.4/	2.8/
	± 0.01 dB	- 68 dB	2.8/	5.6/
	± 0.01 dB	- 57 dB	4.4/	8.8/
	± 0.01 dB	- 70 dB	12.7/	25.4/

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“ **Gating**”“Gating”

3. “ **Shape**”

- 6 dB E5071C

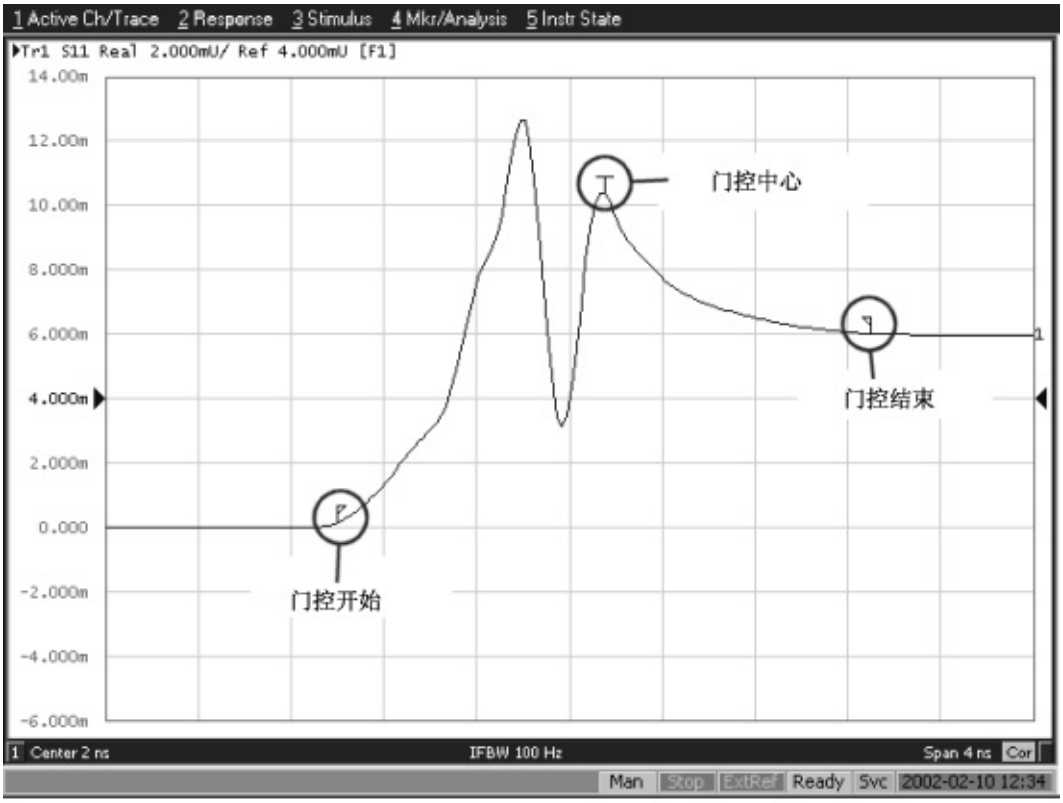
-T_{span}

T_{span}

T_{span}

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“ **Gating**”“Gating”
3.

Start	
Stop	
Center	
Span	



e5070aue205

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“ **Gating**”“Gating”
3. “ **Gating**”“ **ON**”

-
-

	TDR

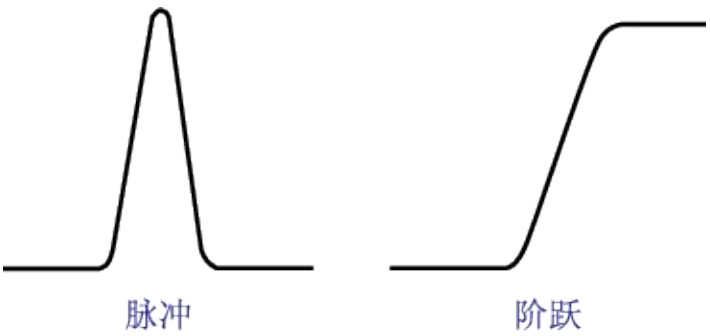
DUT

DUT	DUT	DUT
		TDR
/		

	•
	  •
	  •
	•
	  •	

	• • • SWR • SWR	

E5071C DUT 0 0 0



e5071c417

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“ **Transform**”“Transform”

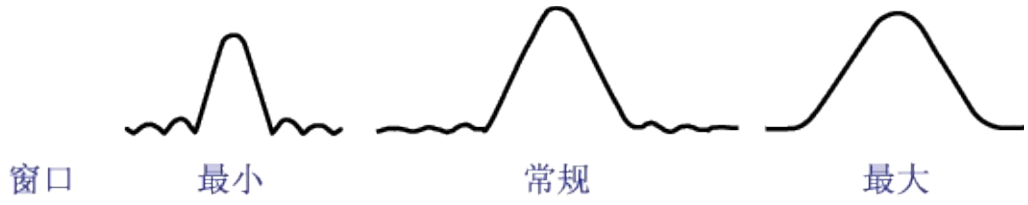
3. “ **Type**”
4. “ **Format**”“Format”

E5071C

-
-

“maximum”“normal”“minimum”
/

		50%		(10 - 90%)
Minimum	- 13 dB		- 21 dB	0.45/
Normal	- 44 dB	0.98/	- 60 dB	0.99/
Maximum	- 75 dB	1.39/	- 70 dB	1.48/



e5071c419

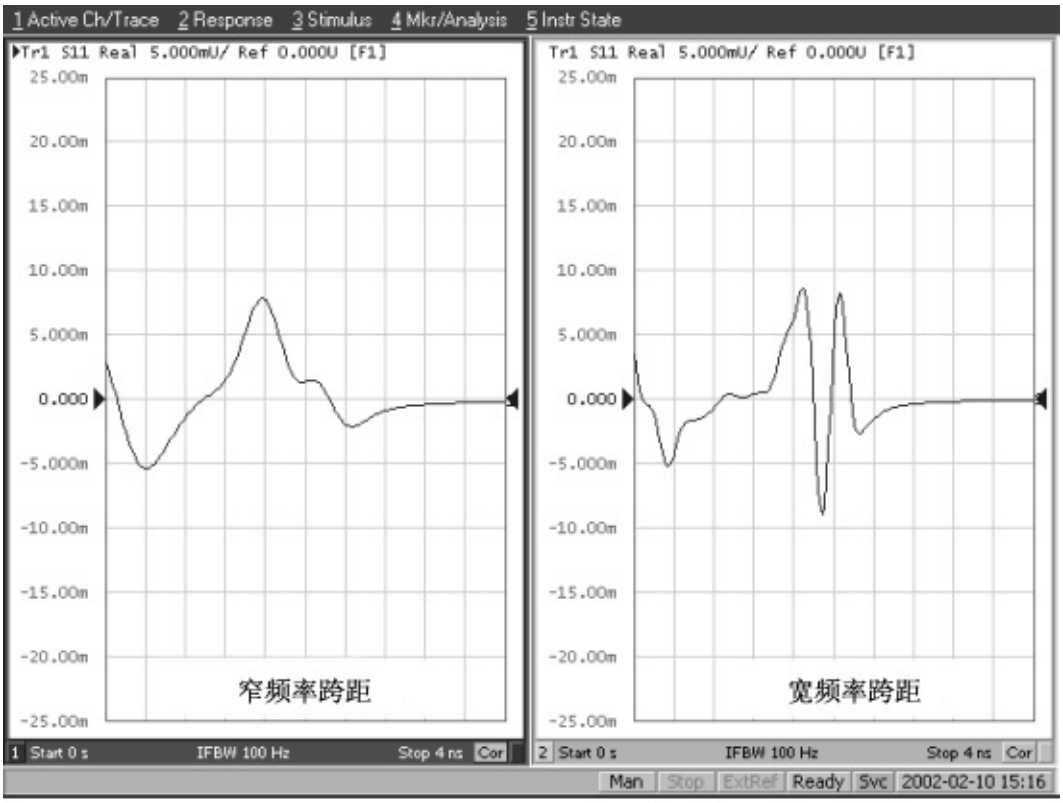
1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace Prev**”

2. “ **Analysis**”“ **Transform**”“Transform”
3. “ **Window**”

Maximum	- b 13
Normal	- b 6
Minimum	- b 0
Rise Time	
Kaiser Beta β	- b - 0 13

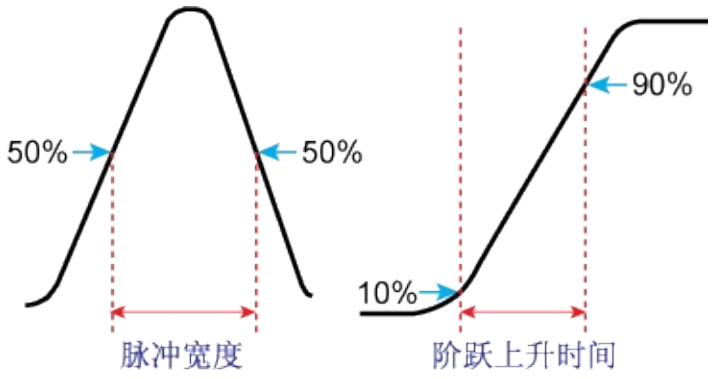
“ **Kaiser Beta” β“Impulse Width** ”“ **Rise Time**”

-
-
-



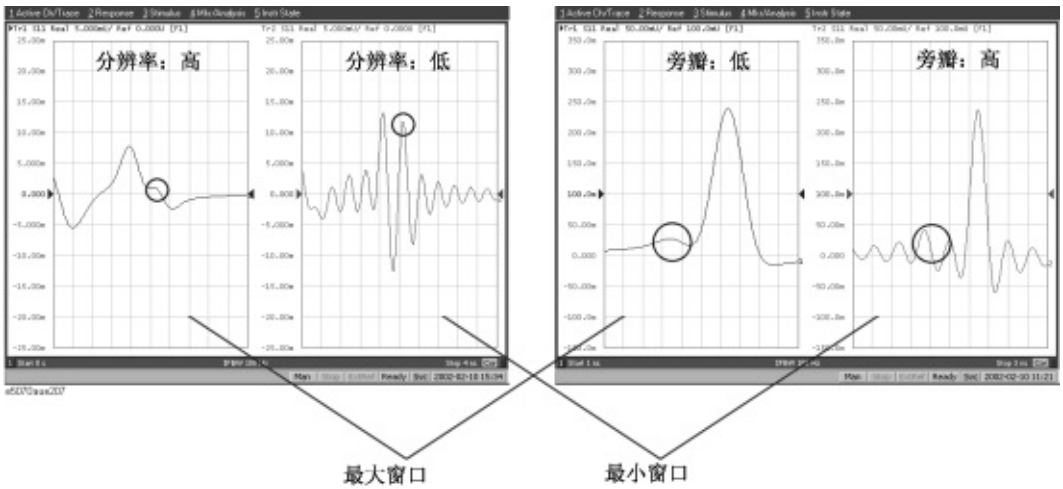
e5070aue208

50% 10% 90%



e5071c418

Minimum	0.45/	0.60/	1.20/
Normal	0.99/	0.98/	1.95/
Maximum	1.48/	1.39/	2.77/



DUT

F_{span} N_{meas} ΔF

$$\Delta F = \frac{F_{span}}{N_{meas} - 1}$$

-1

-
-

T_{span}

$$T_{span} = \frac{1}{\Delta F}$$

Lspan V c (3*E ^{8} m/s)

$$L_{span} = \frac{Vc}{\Delta F}$$

DUT L_{span} L_{span} 1/2

0.66 0.7

1. “ **Channel Next**”/“ **Channel Prev**”
2. “ **Sweep Setup**”“ **Sweep Type**”>“ **Lin Freq**”
3. “ **Start**”/“ **Stop**”“ **Center**”/“ **Span**”
4. “ **Sweep Setup**”“ **Points**”“ **Analysis**” >“ **Transform**”>“ **Set Freq Low Pass**”

> 300 kHz *	= /
< 300 kHz *	300 kHz = 300 kHz *

“ **Set Freq Low Pass**”

E5071C

- - T_{span}
- T_{span}

T_{span}

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“ **Transform**”“Transform”
3.

Start	
Stop	
Center	
Span	

1. “ **Channel Next**”/“ **Channel Prev**”“ **Trace Next**” /“ **Trace prev**”
2. “ **Analysis**”“ **Transform**”“Transform”
3. “ **Transform**”“ **ON**”
4.
   -
   -

- GPIB
- HTTP

- /
- LCD
-

-
-
-
- /
-
-

GPIB

- E5071C / GPIB
- USB/GPIB

E5071C GPIB

E5071C / GPIB

GPIB GPIB E5071C E5071C / GPIB

1. “ **System**”
2. “ **Misc Setup**”>“ **GPIB Setup**”GPIB >“ **Talker/Listener Address**”/
3.

USB/GPIB

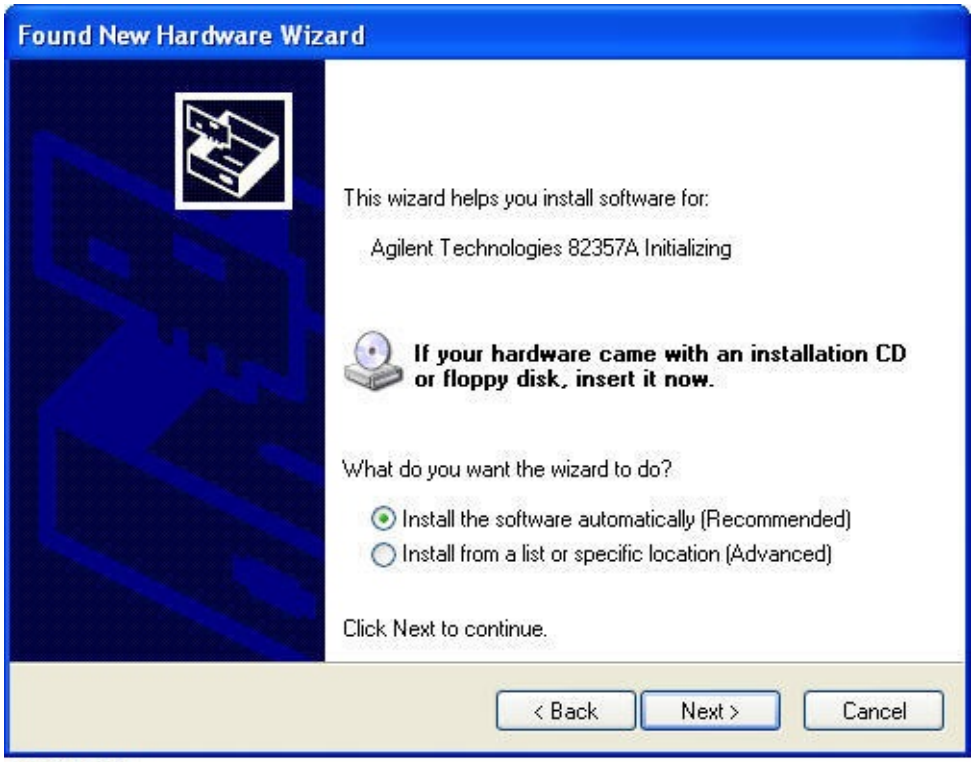
E5071C USB/GPIB E5071C USB GPIB

USB/GPIB

USB/GPIB

1. E5071C USB USB/GPIB
2. “ **No, not this time**”“ **Next**”

3. “ **Install the software automatically (Recommended)**”
   “ **Next**”



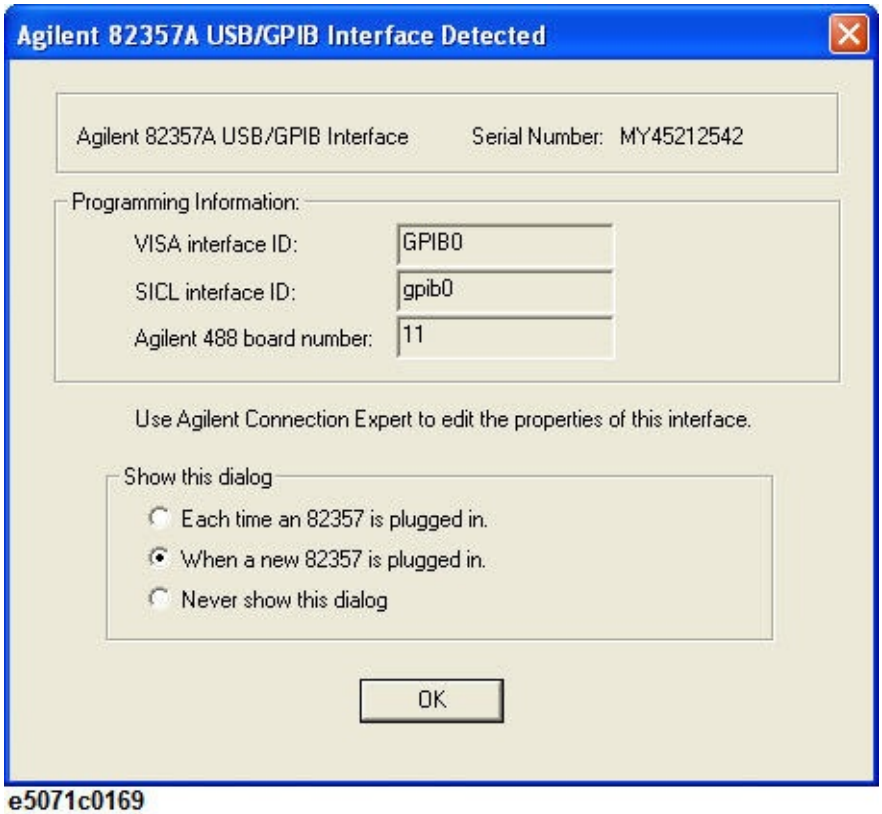
e5071c161

4. “ **Finish**”

e5071c163

5. “**Found New Hardware Wizard**” 2 4

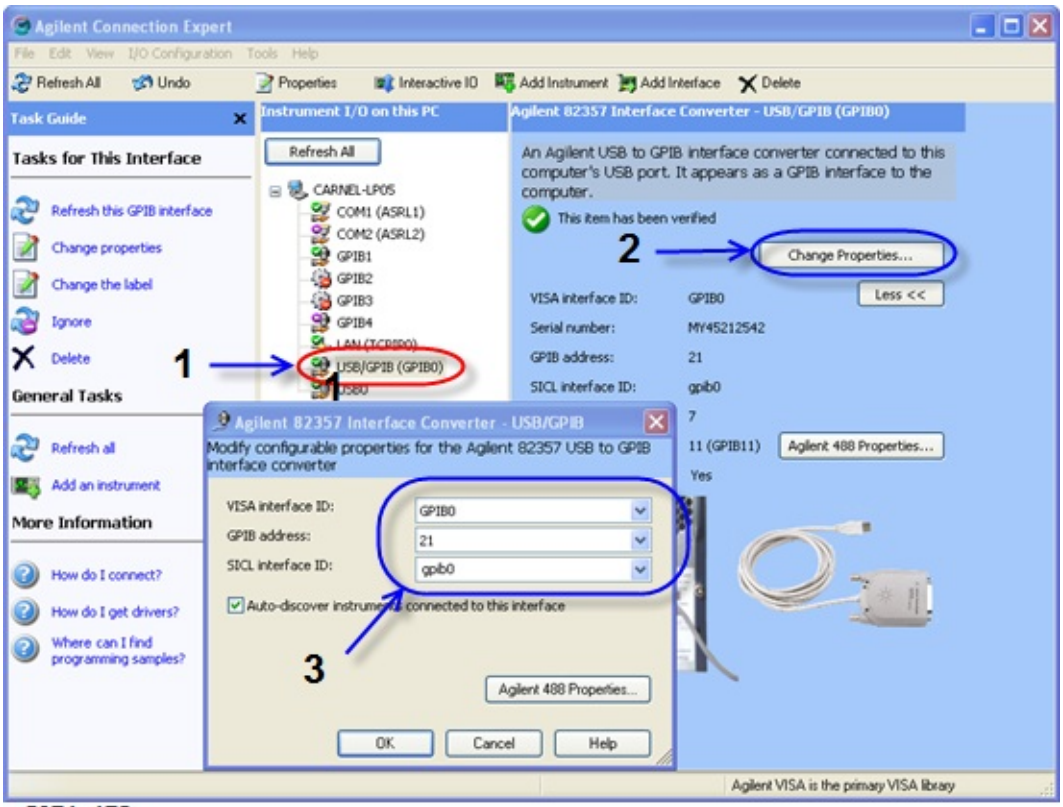
6. “ **When a new 82357 is plugged in**” 82357 “ **OK**”



e5071c0169

USB/GPIB / USB/GPIB

1. “ **System**”“ **Misc Setup**”>“ **GPIB Setup**”GPIB >“ **System Controller Configuration**”
2. “Agilent Connection Expert”Agilent Windows XP “Start”“Agilent Connection Expert”Agilent



e5071c172

3. USB/GPIB (GPIBx)
4. “Change Propetries...”...
5. USB/GPIB

HTTP

- Web
- VNC

(http) E5071C IP PC Web E5071C Web
E5071C PC

- Java Script
- Internet Explorer
- Netscape

E5071C

E5071C

Web

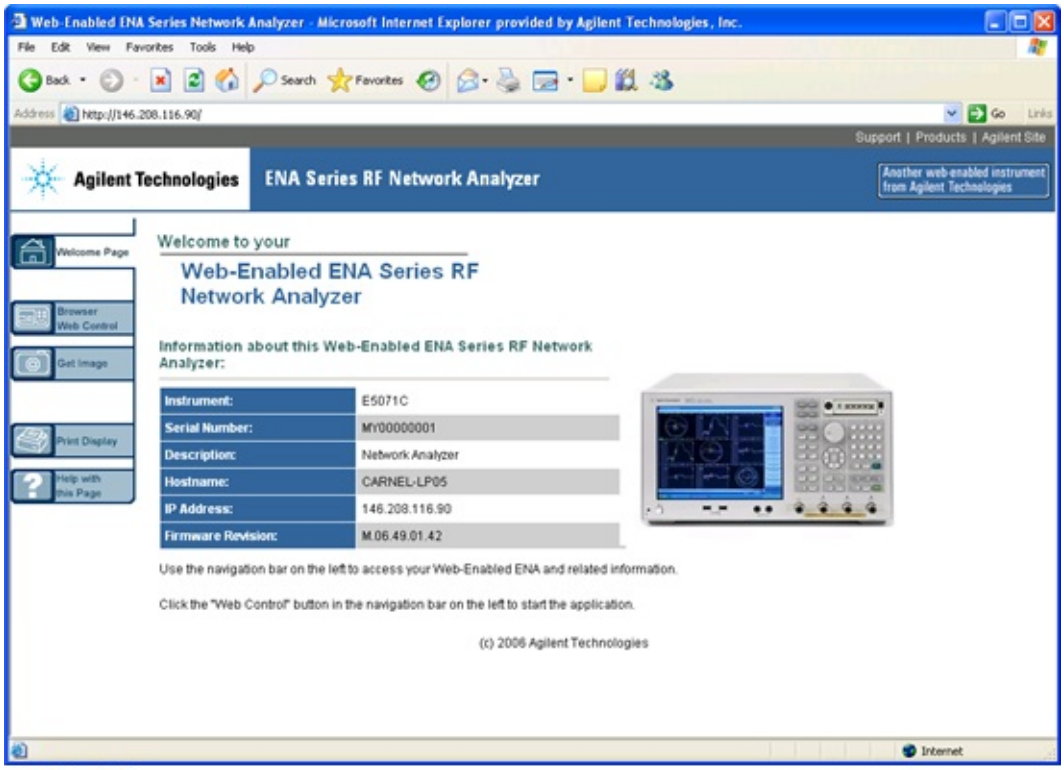
E5071C Web PC

1. “ **System**”
2. “ **Misc Setup**”>“ **Network Setup**”
3. “ **Web Server**”Web

PC Java Runtime Environment E5071C Java Runtime Environment http://www.java.com/

PC

1. PC Web
2. IP
3.



e5071c173

VNC

VNC Web VNC

VNC VNC http://www.realvnc.com

1. “ **System**”
2. “ **Misc Setup**”>“ **Network Setup**”
3. “ **VNC Server Configuration**”VNC “VNC Server Properties”VNC

//

//

1. “ **System**”>“ **Misc Setup**”>“ **Clock Setup**”
2. “ **Show Clock**”//

LCD

E5071C LCD

LCD

1. “ **System**”
2. “ **Backlight**”/
3. LCD
4. “ **Preset**” LCD “ **Preset**”

E5071C

1. “ **System**”
2. “ **Service Menu**”>“ **Test Menu**”>“ **Adjust Touch Screen**”
3.
4. x x x
5.

LED 1

DUT DUT DUT

	DUT	50 Ω DUT
1	1 3	2 4
2	2 3	1 4
3	3 1	2 4
4	4 1	2 3

:INIT:CONT OFF

“ **Trigger**”>“ **Single**”

“**System**”>“ **Service**”>“ **Init Src Ctrl [ON]**” []

“**System**”>“ **Service**”>“ **Init Src Port [1|2|3|4]**” [1|2|3|4]

SCPI 1

:INIT:CONT OFF

:SYSTem:ISPControl[:STATe] ON

:SYSTem:ISPControl:PORT 1

“ **System**”>“ **Service**”>“ **Init Src Ctrl [OFF]**” []

SCPI SYSTem:ISPControl[:STATe] OFF

VBA SCPI.SYSTem.ISPControl.STATe = False

-
-
-
-

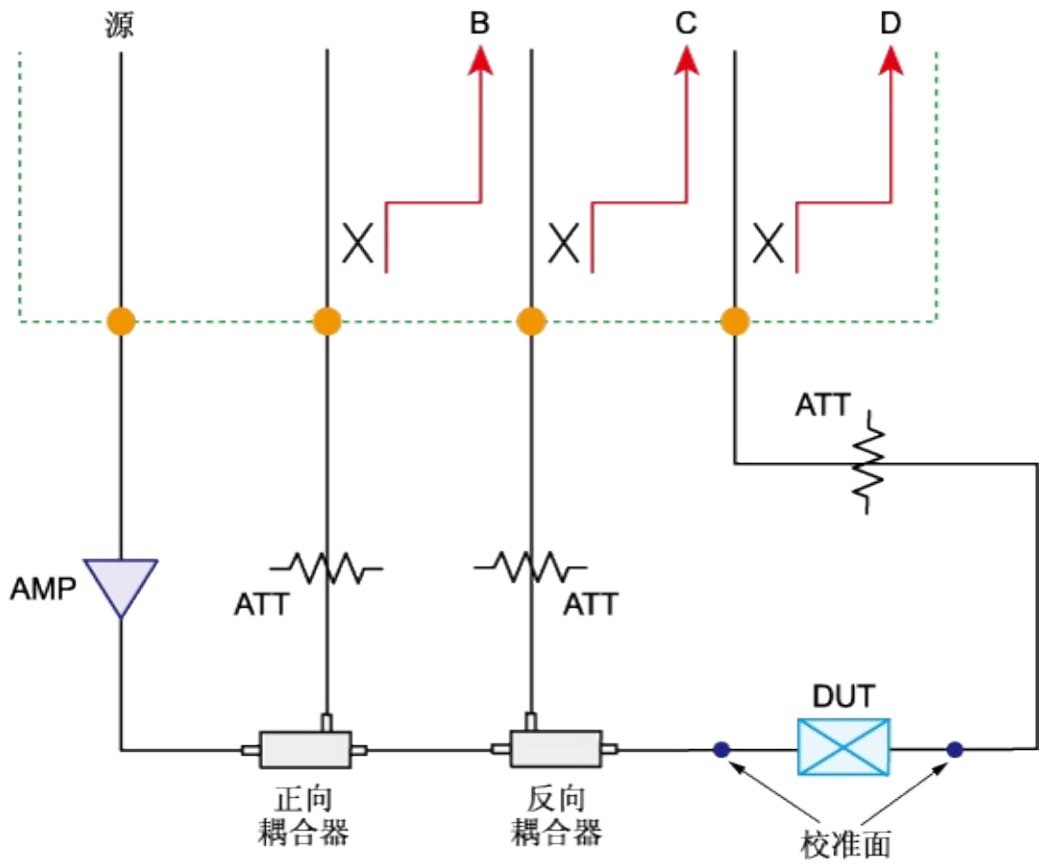
E5071C // E5071C S

E5071C

E5071C DUT E5071C

S21 S11“Mode 1” 1

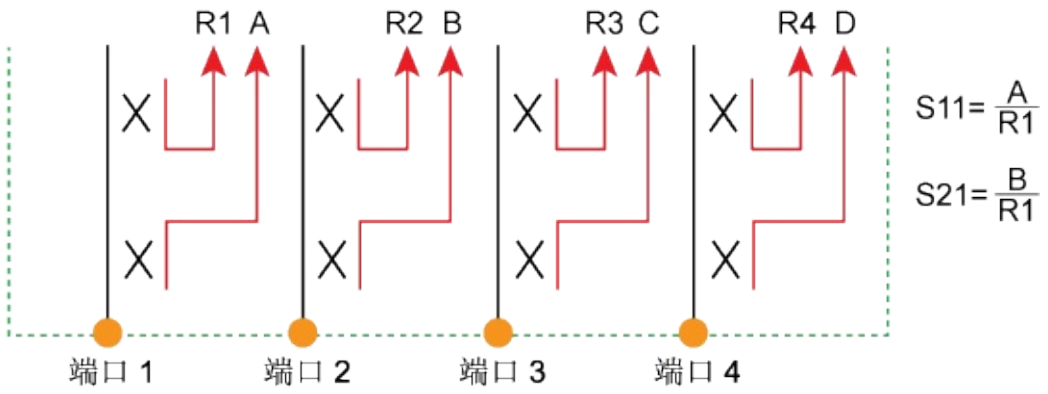
1



e5071c474

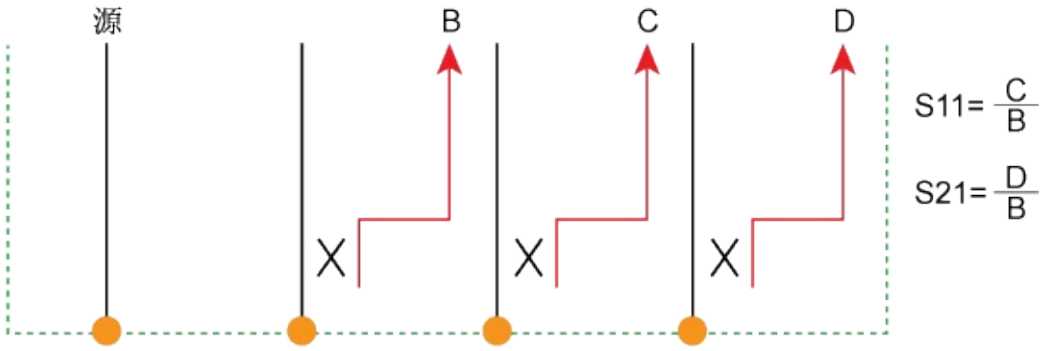
“OFF”“Mode 1” 1“Mode 2” 2

- “OFF” 4 S



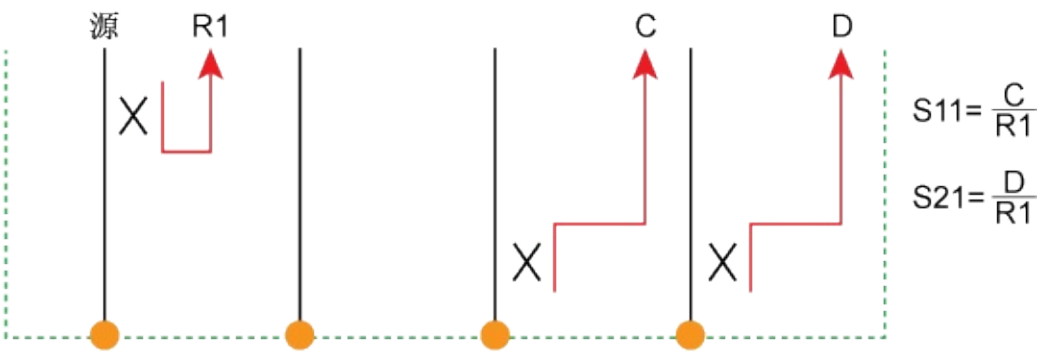
e5071c475

- “Mode 1” 1“Mode 2” 2 2 T/R
  1. Mode 1 1



e5071c476

  2. Mode 2 2



e5071c477

1. “ **System**”>“ **Service Menu**”>“ **External Test Set**”

2.

ECal

“Mode 1” 1“Mode 2” 2 E5071C

“Mode 1” 1“Mode 2” 2 E5071C

-
-
-
-
- 1

ECal “Mode 1” 1“Mode 2” 2 4
4 ECal 1/2/3 4

E5071C	**1/2/3**	**4**
ECal	A	B
	B	C
	C	D
	D	A

“Mode 1” 1“Mode 2” 2 E5071C 15-3 15-4
S11/S21

“Mode 1” 1“Mode 2” 2

-
-

E5071C

1. “ **System**”
2. “ **Service Menu**”>“ **Enable Options**”
3.

E5071C

1. “ **System**”
2. “ **Firmware Revision**”
3. “Firmware Revision”

/

/

/

1. “ **System**”
2. “ **Misc Setup**”>“ **Key Lock**”
3. /

Front Panel & Keyboard Lock	/
Touch Screen & Mouse Lock	/

“Standby”

-
-

E5071C

	• •
	• •

1. “ **System**”
2. “ **Misc Setup**”>“ **Beeper**”>“ **Beep Complete**” /
3. “ **Test Beep Complete**”

1. “ **System**”
2. “ **Misc Setup**”>“ **Beeper**”>“ **Beep Warning**”/
3. “ **Test Beep Warning**”

- /
-
-

/

E5071C “ **OK**”“ **Cancel**”

1. “ **System**”
2. “ **Misc Setup**” >“**Preset Setup**”
3. “ **Confirm**”/

E5071C

1. “ **System**”
2. “ **Misc Setup**”>“ **Preset Setup**”>“ **State**”
3.

Factory	
User	
Cancel	

E5071C

1. “ **Save/Recall**”/
2. “ **Save State**”>“ **User Pres**”

Windows XP

- Windows XP
- /
-
-
-
- PC E5071C
- Windows
- USB
-

Windows XP

- USB
-
- LAN
-
-

Windows XP

USB

ENA USB USB USB

USB Windows

USB ID

USB ECAL ENA “ **Next**”
USB

LAN

Windows DHCP IP LAN LAN LAN
“Hardware Wizard”

Windows Windows

Windows XP

/

E5071C

-
- VBA

E5071C “**Date & Time**”/

Windows XP

-
-

Windows XP

“Auto Log On”
“Auto Log On”

1. “ **User Accounts**”
2. “ **Change an Account**”
3. “agena”
4. “ **Create a password**”
5.
6. “ **Create Password**”

CTRL+ALT+DELETE

CTRL+ALT+DELETE

1. “ **User Accounts**”
2. “ **Change the way users log on or off**”
3. “ **Use the Welcome screen**”
4. “ **Apply Options**”
5. Windows XP

“agena”“administrator”“administrator”
“e507xa”“agena”

1. “ **User Accounts**”
2. “ **Create a new account**”
3. “ **Next**”
4. “ **Computer Administrator**”
5. “ **Create Account**”

-
- Windows

Windows XP

Windows XP E5071C

E5071C E5071C

e5071c060

Windows

“ **Start**”>“ **All Programs**”>“ **Accessories**” >“**Accessibility**”>“ **On-Screen Keyboard**”

E5071C LAN LAN

E5071C LAN

- /
- IP
- .

Windows XP PC

Windows XP

/

/ E5071C

/

1. LAN E5071C LAN
2. “ **System**”>“ **Misc Setup**”>“ **Network Setup**” >“**Network Configuration**”“ **Network and Dial-up Connections**”
3.

   -

     “ **Network and Dial-up connections**”“Local Area Connection”

   -

     “ **Network and Dial-up connections**”“Local Area Connection”“ **Local Area Connection Status**” “ **Disable**”

4. “ **Network and Dial-up Connections**”

IP

IP

1. “ **System**”>“ **Misc Setup**”>“ **Network Setup**” >“**Network Configuration**”
2. “ **Network and Dial-up connections**”“Local Area Connection”“ **Local Area Connection Status**”
3. “ **Properties**” “ **Local Area Connection Properties**”
4. “ **Internet Protocol (TCP/IP)**”Internet (TCP/IP) “ **Properties**”
5. “ **Internet Protocol (TCP/IP) Properties**”Internet (TCP/IP) “ **Use the following IP address**” IP “ **IP address**”IP “ **subnet mask**”“ **gateway address**”
6. IP DHCP “ **Obtain an IP address automatically**” IP
7. “ **Internet Protocol (TCP/IP) Properties**”Internet (TCP/IP) “ **OK**”
8. “ **Local Area Connection Properties**”“ **OK**”
9. “ **Local Area Connection Status**”“ **Close**”
10. “ **Network and Dial-up Connections**”

1. “ **System**”>“ **Misc Setup**”>“ **Network Setup**” >“**Network Identification**”
2. “ **Properties**”
3. “**Computer Name**”
4. “ **OK**”
5. “ **System Properties**”“ **OK**”
6. “ **Yes**” E5071C

E5071C

PC E5071C

E5071C LAN LAN PC E5071C D

- PC
- PC E5071C

Windows XP

PC

PC

1. “ **Save/Recall**”/>“ **Explorer**”
2. “ **USER (D:)**” (D:)“File”“ **Properties**” “USERS(D:)Properties” (D:)
3. “ **Sharing**”
4. “ **New Share**” “New Share”
5. “ **Share Name**” PC “ **OK**”
6. “USERS(D:)Properties” (D:) “ **OK**”

PC E5071C

PC Windows XP PC E5071C D

PC

1. “Start”“ **Explorer**”
2. “ **Tools**”>“ **Map Network Drive”** “Map Network Drive”
3. **\\C_NAME\S_NAME** “ **OK**”
4. “ **OK**”

C_NAME E5071C **S_NAME** D PC

6. PC PC “ **agena**”

Windows

Windows

1. Windows
2. “General”
    - “**On (recommended)**”
    - “**Off (not recommended)**” Windows

Windows “ **Windows Firewall**”Windows

Windows XP

USB

/ USB

- USB
- USB

Windows XP

USB

1. “ **Save/Recall**”/
2. “ **Explorer...**”... **D:\Agilent\Service**
3. “ **DisableUsbStorage.exe**”
4. “SUCCEEDED”“ **OK**” USB E5071C
   “Hardware Wizard” USB

USB

1. “ **Save/Recall**”/
2. “ **Explorer...**”...
3. **D:\Agilent\Service** “ **EnableUsbStorage.exe**”
4. “SUCCEEDED”“ **OK**”

- USB **DisableUsbStorage.exe** E5071C **EnableUsbStorage.exe** USB

-

USB E5071C /

-
-
-
-

Windows XP

E5071C (USB) E5071C

E5071C Mini-DIN (USB) E5071C

E5071C _____ E5071C

E5071C Mini-DIN (USB) E5071C

- .
- .
- .
- .
- /
- .
- .
- .
- .
- .
- .
- .

VBA

- VBA
-
- E5071C
-
-
-
-

-
- COM
-

- .
- .
- .
- .
- /
- .
- .
- .
- .
- .
- .
- .

-
- GPIB
- LAN
- USB
- SCPI

PC	GPIB/	E5071CGPIB GPIB
	LAN	E5071CLAN LAN
	USB	E5071CUSB USB
E5071C		E5071C VBA.E5071C
	GPIB	E5071C VBAE5071CGPIB

GPIB

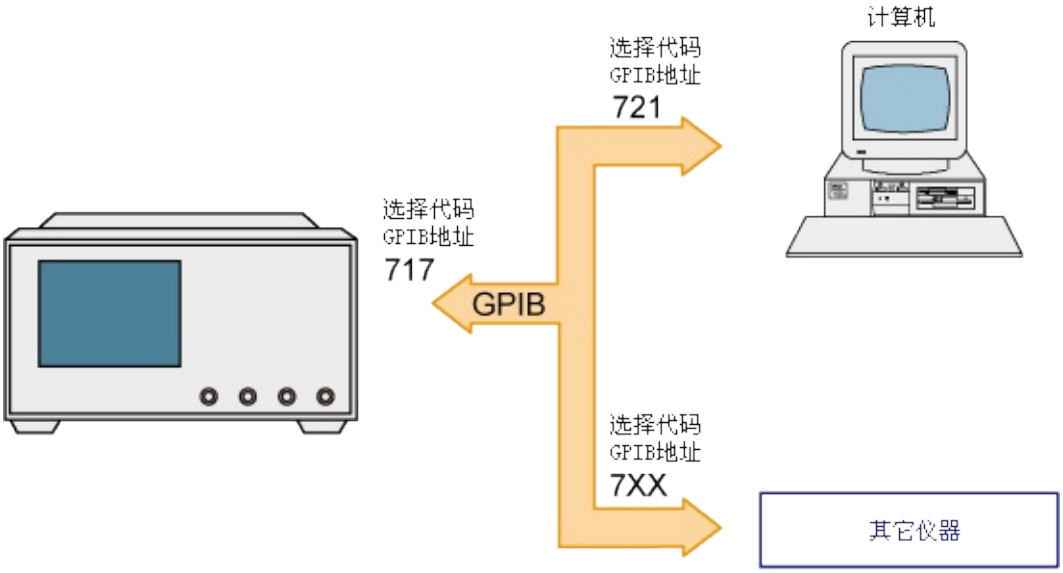
- GPIB
-

GPIB

GPIBIEEE488.1 IEC-625 IEEE488.2
GPIBAgilent E5071CE5071CE5071CGPIB

GPIBE5071CGPIB

GPIB



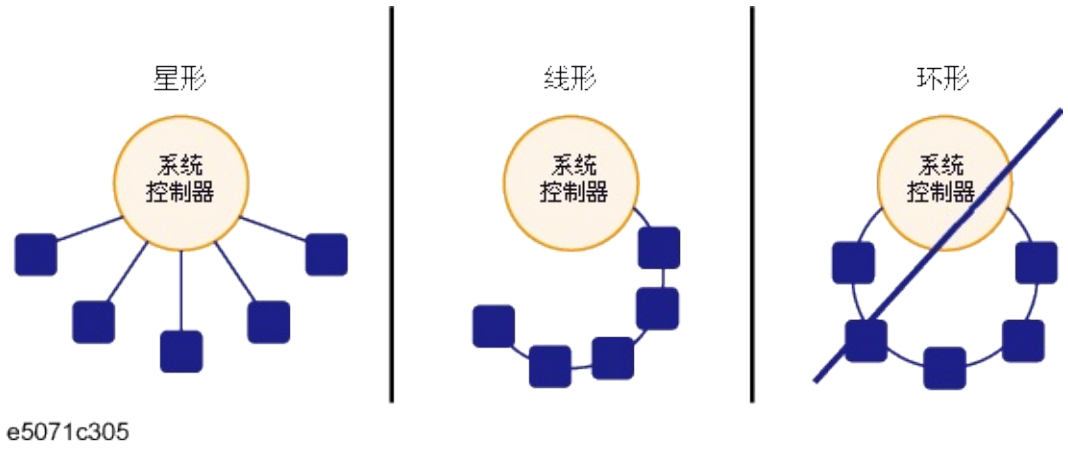
e5071c337

E5071CE5071CSRQGPIBSRQE5071CE5071C

- E5071C
- PCLANAgilent I/O
- /
- GPIB

- GPIB15
- 4mGPIB2mx20m

- 45
-



(Device selector)

GPIB/

7GPIB7GPIB17717GPIB717

E5071CGPIB

GPIB E5071CGPIB.

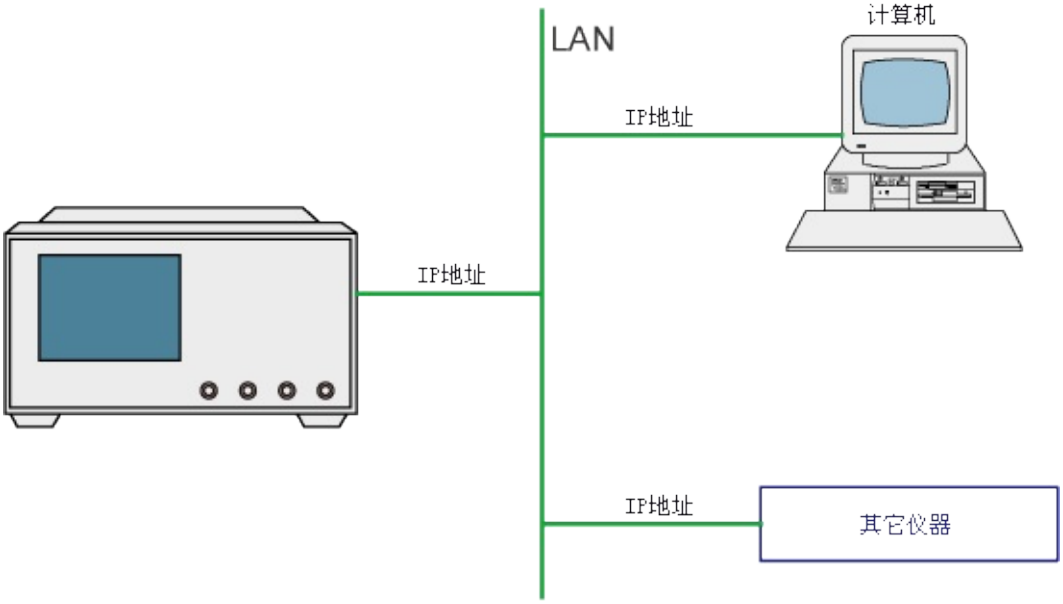
LAN

-
-
-
- SICL-LAN
- CVisual Basic
- Agilent VEE
- (Telnet Server)

LANSICL-LANE5071CE5071C

LANE5071CLAN

LAN



e5071c338

- E5071C
- PCLAN
- /
- LAN

SICL-LAN

SICL-LANE5071CSICL-LANSICLSICL
VISAE5071CUNIXCWindowsVisual

E5071C

E5071CSICL-LAN

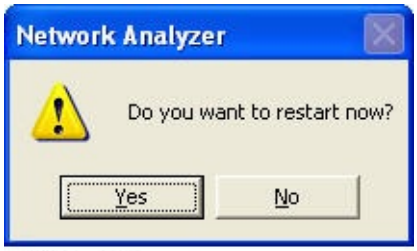
1. E5071CSICL-LAN

   System > **Misc Setup** > **Network Setup** > **SICL-LAN Server [ON]**

2. E5071CGPIBSICL-LAN"XX"
   System > **Misc Setup** > **Network Setup** > **SICL-LAN Address [XX]**

3. E5071CSICL-LAN

4. **Yes**



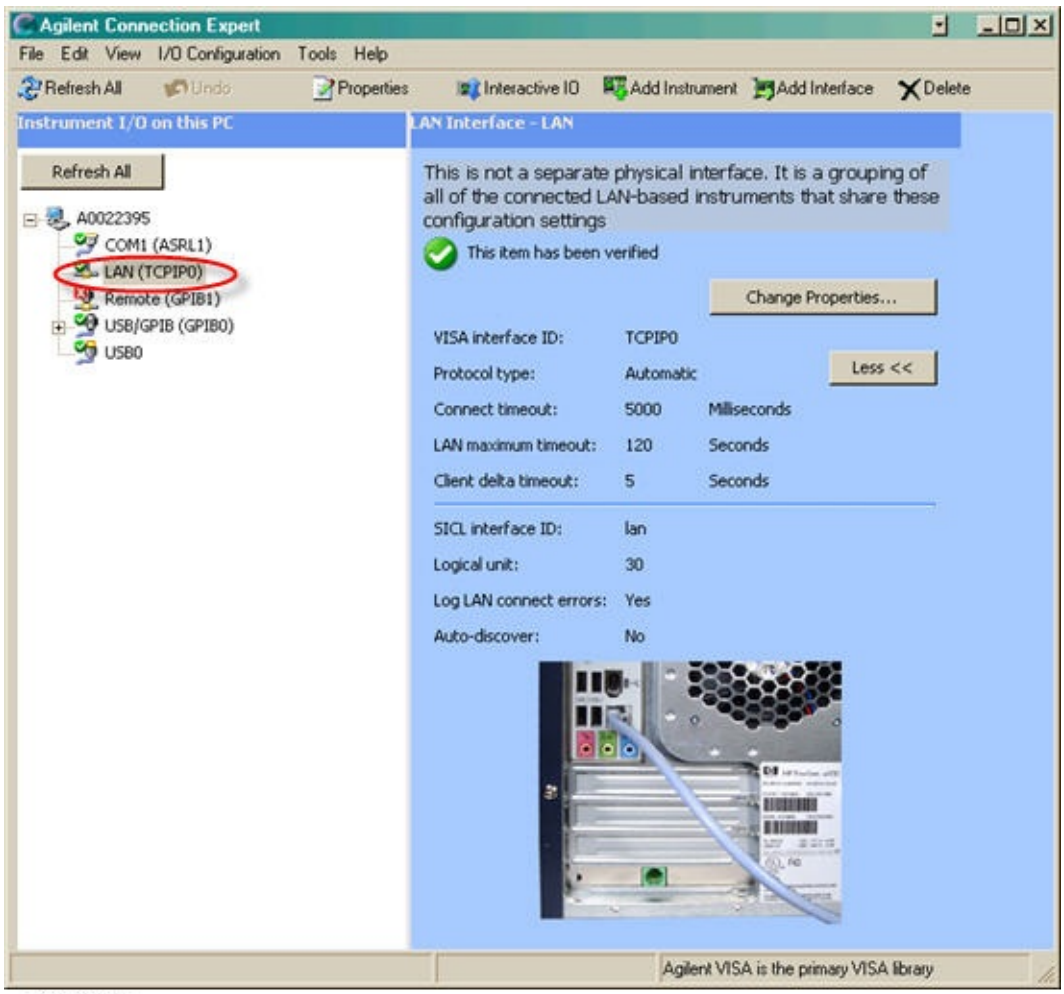
e5071c138

TCP/IPE5071CI/O/Windows

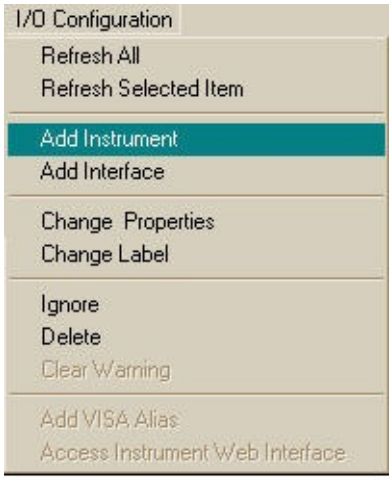
- PCAgilent I/O LibrariesAgilent/Agilent I/O Libraries Suite 14.2

1. PC **Program** > **Agilent I/O Libraries Suite** > **Agilent Connection Expert** Agilent Connection Expert

2. Agilent Connection Expert **Instrument I/O on this PC LAN(TCPIP0)I/O Configuration** > **Add Instrument**

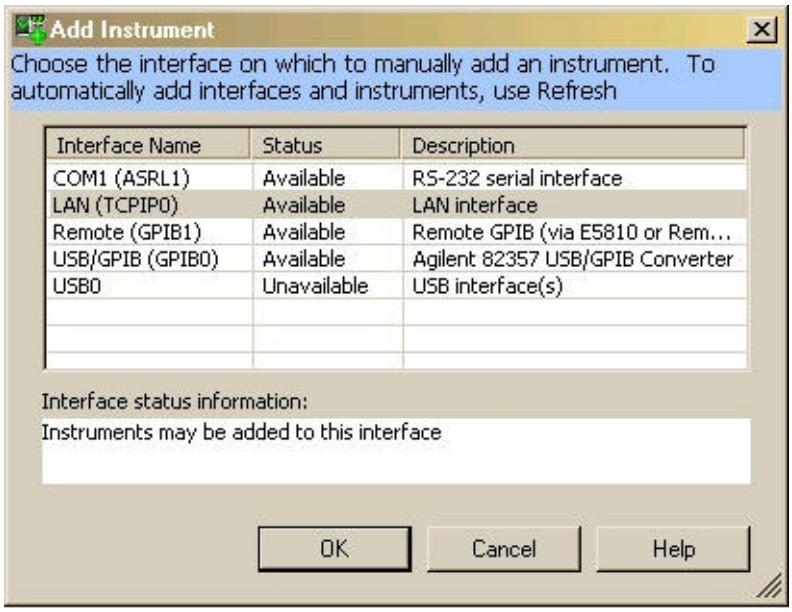


e5071c139



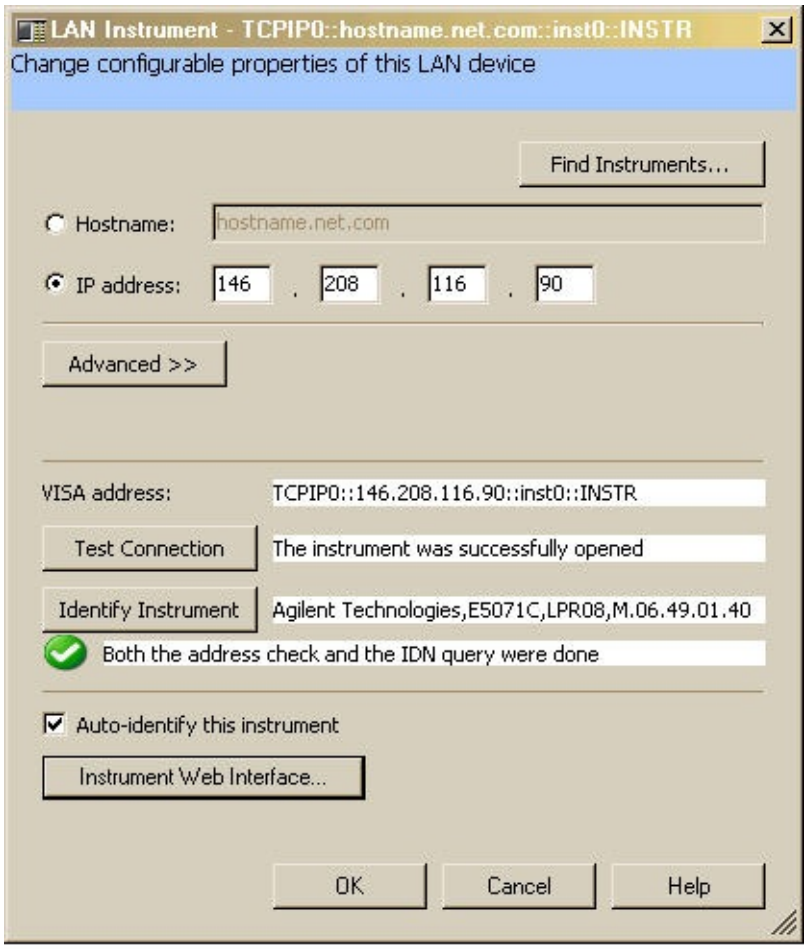
e5071c140

3. **Add Instrument LAN OK**



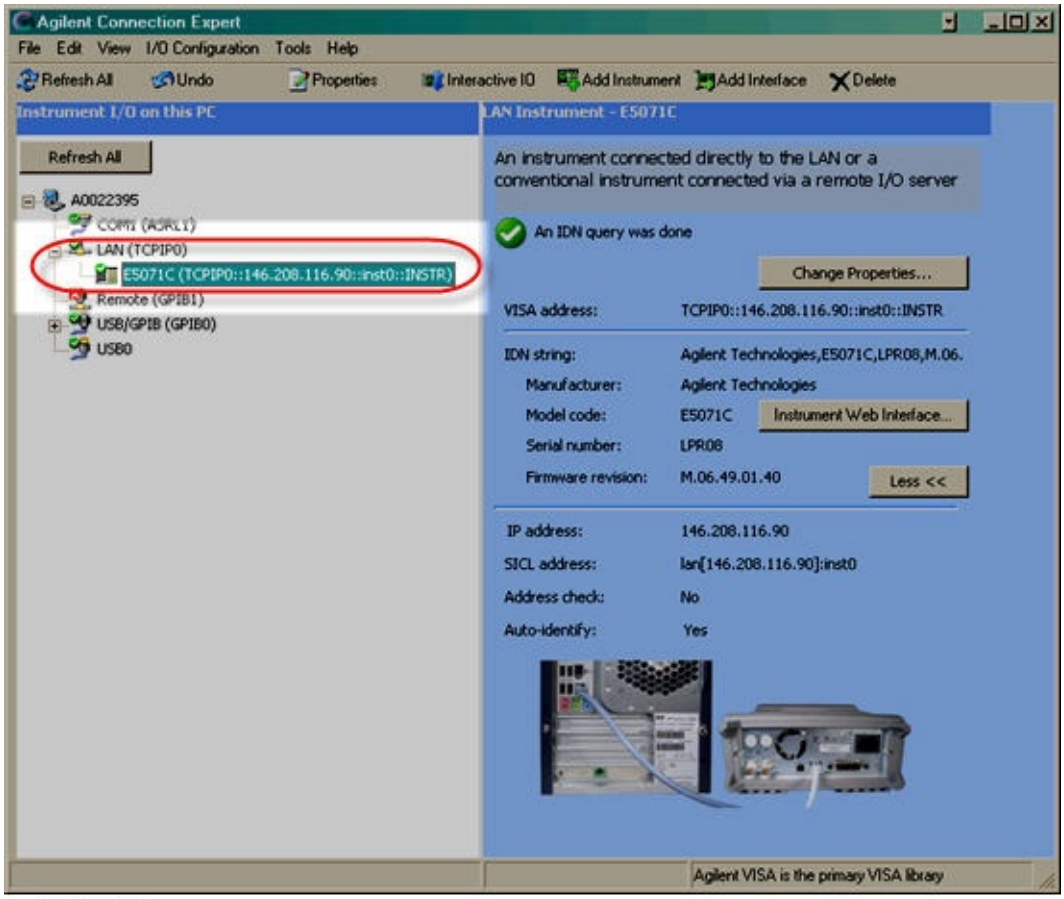
e5071c141

4. LAN Instrument PropertiesE5071CIP **OK** Agilent I/O Libraries Suite



e5071c142

5. Agilent Connection Expert **Instrument I/O on this PC LAN(TCPIP0)** E5071C


Agilent Connection Expert
File Edit View I/O Configuration Tools Help
Refresh All
Undo
Properties
Interactive IO
Add Instrument
Add Interface
Delete
Instrument I/O on this PC
Refresh All
A0022395
COM1 (ASRL1)
LAN (TCPIP0)
E5071C (TCPIP0::146.208.116.90::inst0::INSTR)
Remote (GPIB1)
USB/GPIB (GPIB0)
USB0
LAN Instrument - E5071C
An instrument connected directly to the LAN or a conventional instrument connected via a remote I/O server
An IDN query was done
Change Properties...
VISA address: TCPIP0::146.208.116.90::inst0::INSTR
IDN string: Agilent Technologies,E5071C,LPR08,M.06.
Manufacturer: Agilent Technologies
Model code: E5071C
Instrument Web Interface...
Serial number: LPR08
Firmware revision: M.06.49.01.40
Less <<
IP address: 146.208.116.90
SICL address: lan[146.208.116.90]:inst0
Address check: No
Auto-identify: Yes
Agilent VISA is the primary VISA library

e5071c143

CVisual Basic

SICLE5071CUNIXCWindowsVisual C++Visual

Agilent VEE

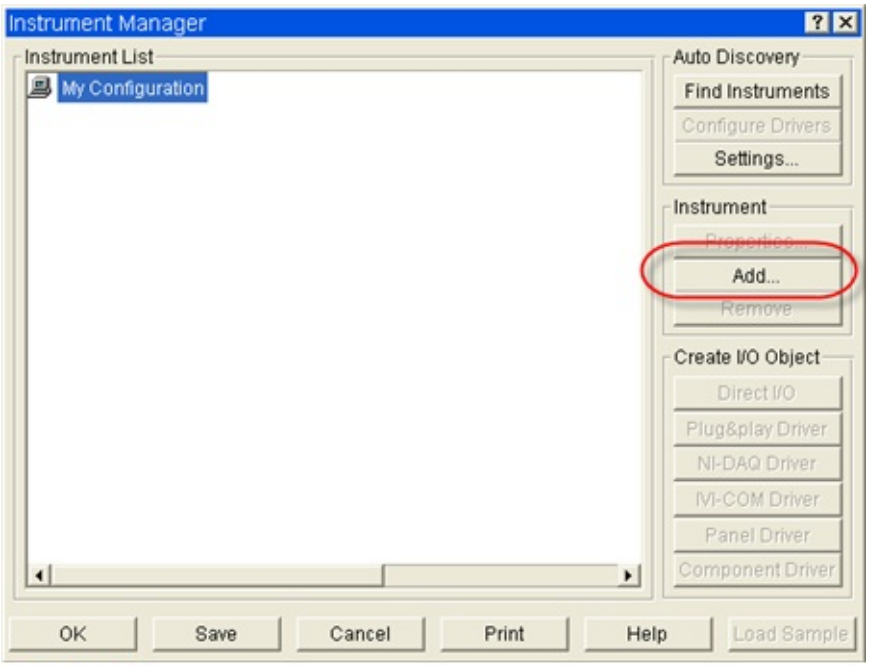
Agilent VEEI/OE5071CE5071CSICL-LAN17IP146.208.116.90

Agilent VEEPCAgilent VEE Pro 7.5 Windows

1. Agilent VEE**I/O** **Instrument Manager**

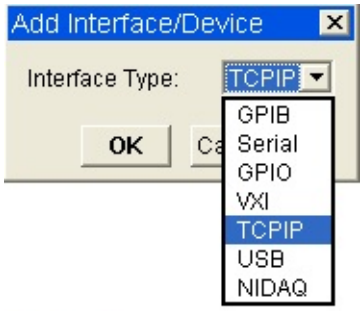
e5071c146

2. **Instrument Manager** **Add...**



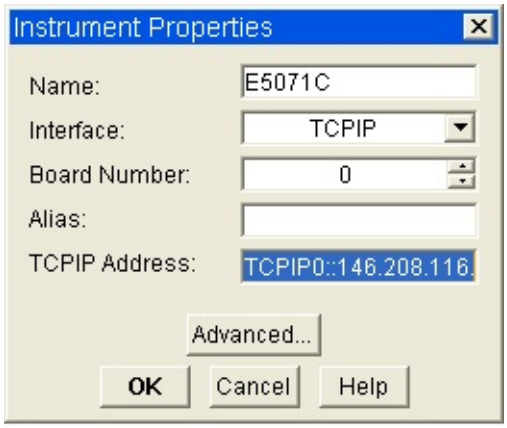
e5071c147

3. **TCPIPOK**



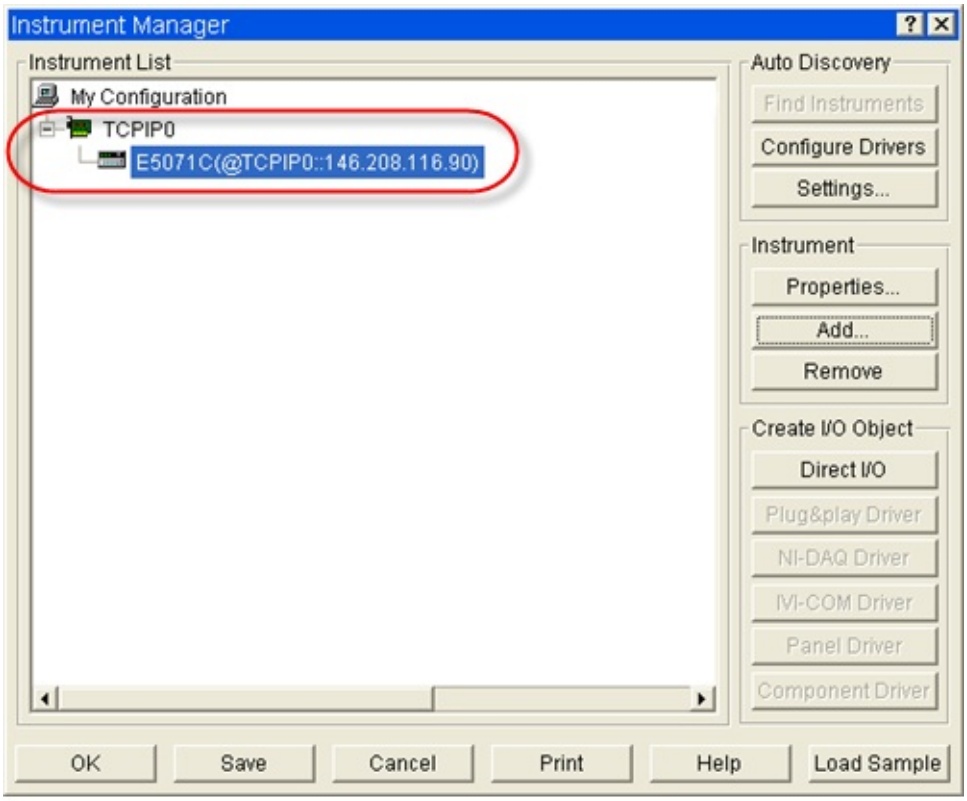
e5071c148

4. **Instrument Properties** NameENAE5071C **TCIP Address**TCPIP0::<IP><IP >E5071CIPE5071CIP 146.208.116.90 **TCPIP Address**TCPIP0::146.208.116.90 **OK**



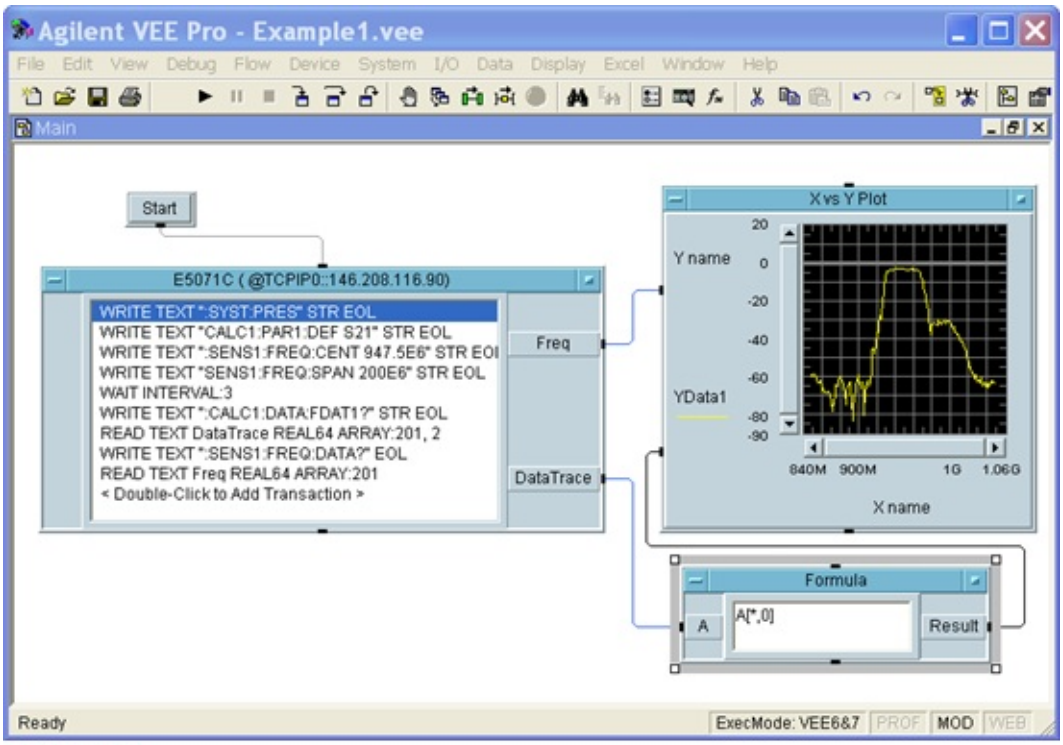
e5071c149

5. Instrument managerE5071C



e5071c150

I/O


Agilent VEE Pro - Example1.vee
File Edit View Debug Flow Device System I/O Data Display Excel Window Help
Main
Start
E5071C (@TCPIP0::146.208.116.90)
WRITE TEXT ":SYST:PRES" STR EOL
WRITE TEXT "CALC1:PAR1:DEF S21" STR EOL
WRITE TEXT ":SENS1:FREQ:CENT 947.5E6" STR EOI
WRITE TEXT "SENS1:FREQ:SPAN 200E6" STR EOL
WAIT INTERVAL:3
WRITE TEXT ":CALC1:DATA:FDAT1?" STR EOL
READ TEXT DataTrace REAL64 ARRAY:201, 2
WRITE TEXT ":SENS1:FREQ:DATA?" EOL
READ TEXT Freq REAL64 ARRAY:201
< Double-Click to Add Transaction >
Freq
DataTrace
X vs Y Plot
Y name
YData1
20
0
-20
-40
-60
-80
-90
840M 900M 1G 1.06G
X name
Formula
A
A[*,0]
Result
Ready
ExecMode: VEE6&7 PROF MOD WEB

e5071c151

E5071C

235025E5071C23telnetTELNET5025

- telnet235025 Windows
- 235025E5071Ctelnet235025

E5071C

E5071C

System > **Misc Setup** > **Network Setup** > **Telnet Server [ON]**

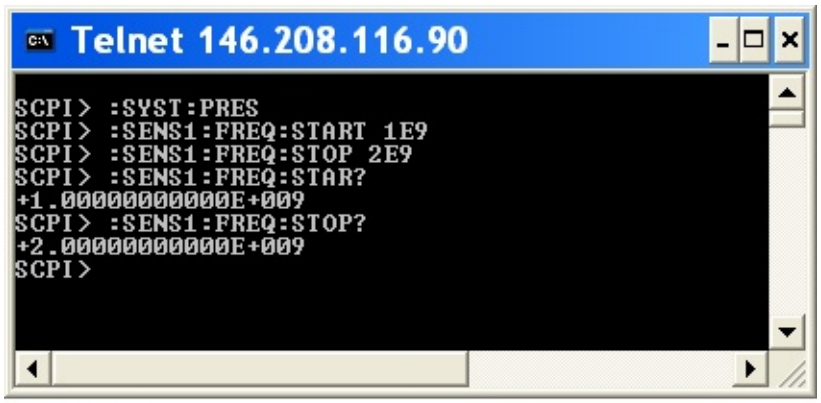
telnet23

E5071CSCPItelnettelnet23

telnetWindowsE5071CIP146.208.116.90e5071c

1. MS-DOS
2. MS-DOStelnet 146.208.116.90telnet e5071c
3. telnet
4. E5071C
5. :SYST:PRES :SENS{1-36}:FREQ:STAR1 GHz:SENS{1-36}:FREQ:STOP command2 GHz

   telnet



e5071c153

6. telnetConnectDisconnectE5071CEditExit telnetUNIXtelnettelnetE5071Ctelnet

5025

E5071C5025

GPIB

CVisual Basic

E5071CUNIXCWindowsC++Visual Basic

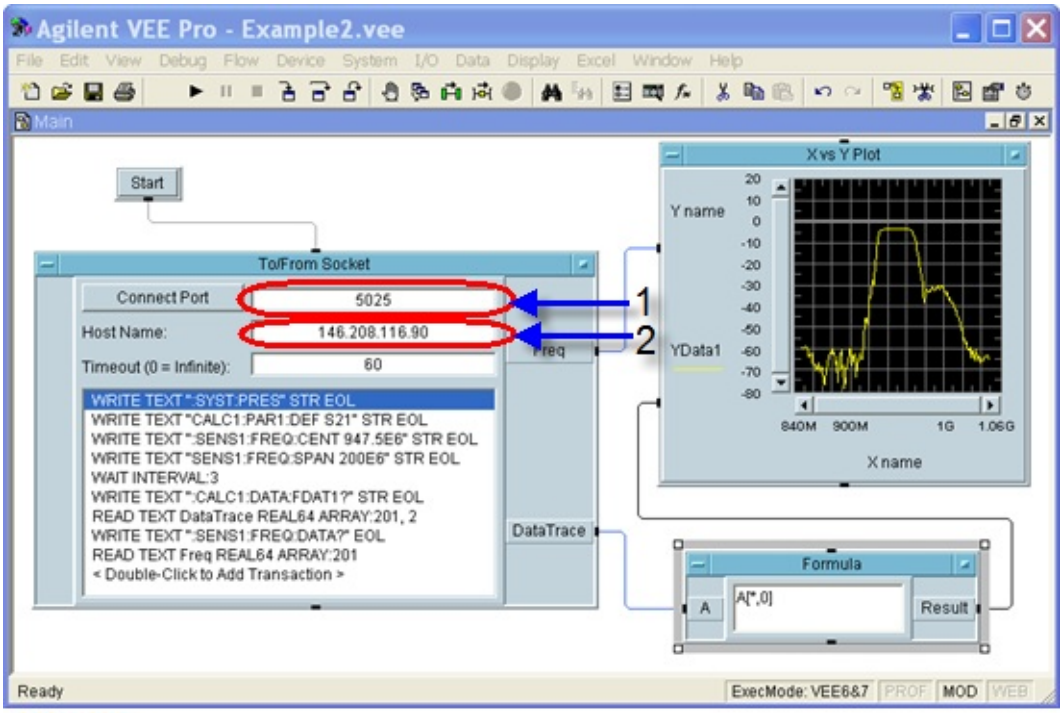
TCP/IPUNIX BSDWindowsBSDWindows WinSockWinSock1.1WinSock2.0

"Controlling Using Telnet Server"WinSock

Agilent VEE

Agilent VEE/5025E5071CE5071CIP146.208.116.905025
Host Name 1 E5071CIP 2

Agilent VEE


Agilent VEE Pro - Example2.vee
File Edit View Debug Flow Device System I/O Data Display Excel Window Help
Main
Start
To/From Socket
Connect Port
5025
Host Name:
146.208.116.90
Timeout (0 = Infinite):
60
WRITE TEXT ":SYST:PRES" STR EOL
WRITE TEXT "CALC1:PAR1:DEF S21" STR EOL
WRITE TEXT ":SENS1:FREQ:CENT 947.5E6" STR EOL
WRITE TEXT "SENS1:FREQ:SPAN 200E6" STR EOL
WAIT INTERVAL:3
WRITE TEXT ":CALC1:DATA:FDAT1?" STR EOL
READ TEXT DataTrace REAL64 ARRAY:201, 2
WRITE TEXT ":SENS1:FREQ:DATA?" EOL
READ TEXT Freq REAL64 ARRAY:201
< Double-Click to Add Transaction >
Freq
DataTrace
1
2
X vs Y Plot
Y name
YData1
20
10
0
-10
-20
-30
-40
-50
-60
-70
-80
840M
900M
1G
1.06G
X name
Formula
A
A[*,0]
Result
Ready
ExecMode: VEE6&7
PROF
MOD
WEB

e5071c152

USB

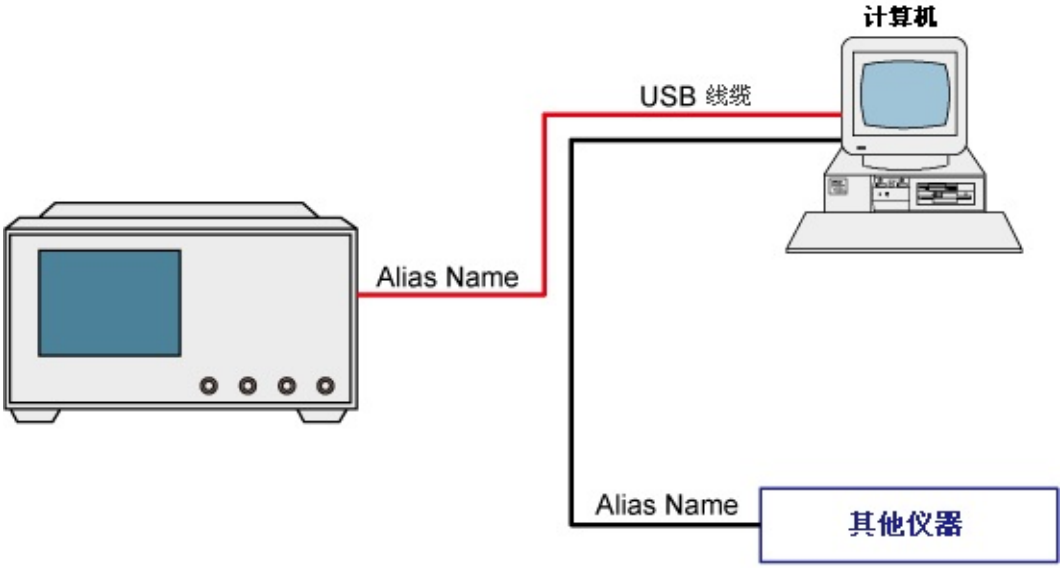
-
-

USBUSBGPIBUSBTMC-USB488USB 2.0.

USB“alias”GPIB

USBE5071CUSB

USB



e5071c339

- E5071C
- USBAPC
- USB
- USBE5071CA/4-prongB/4-prong

USB

USBUSBtype AE5071CUSBtype

	Type A: USB
	Type B: USB (USBTMC)

E5071C

E5071CUSBUSB

USBE5071CI/OUSBUSBUSB

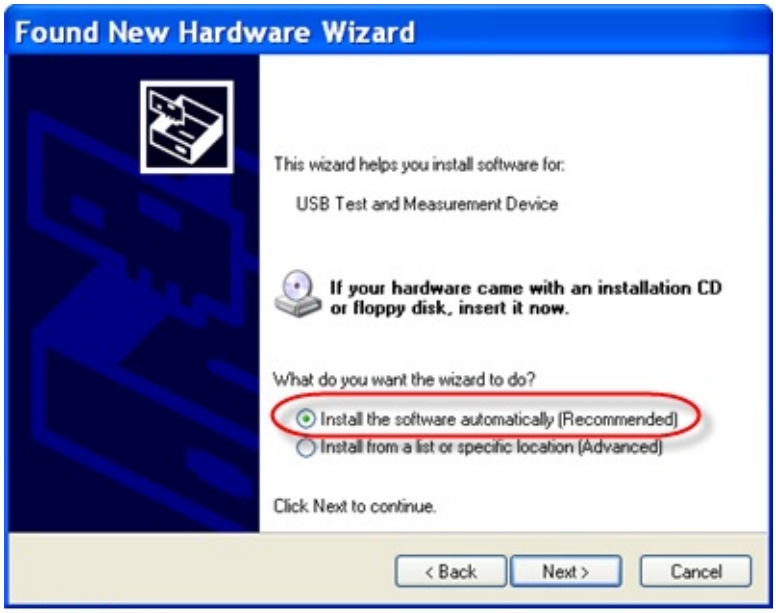
- E5071C
- Agilent I/O Agilent I/O14.2

1. USBE5071C

1. USB **No** **Next**

e5071c155

2. **Next**



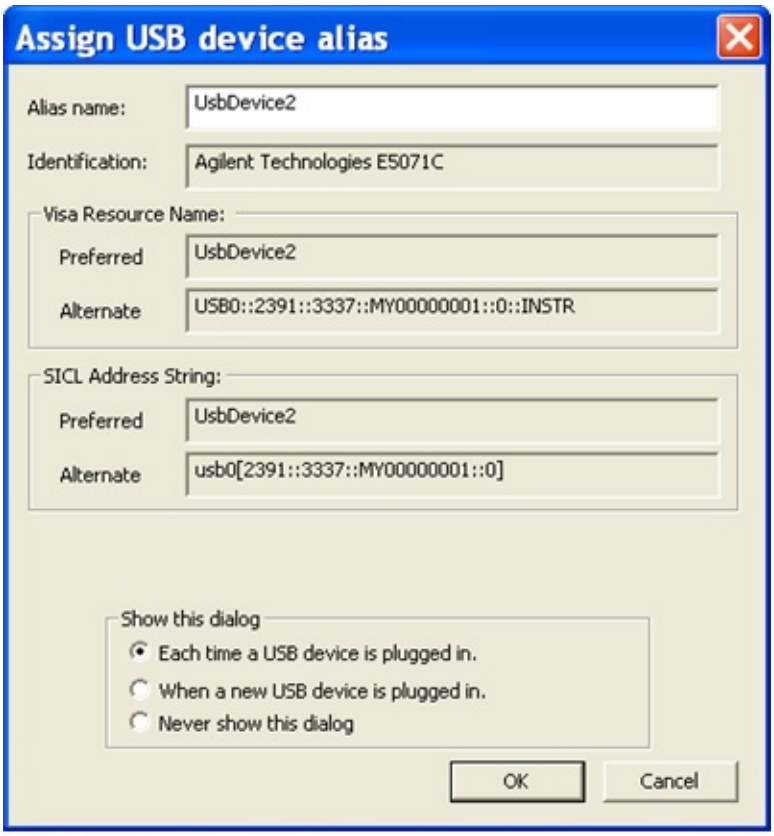
e5071c156

3. Next E5071B

e5071c157

2.

E5071C



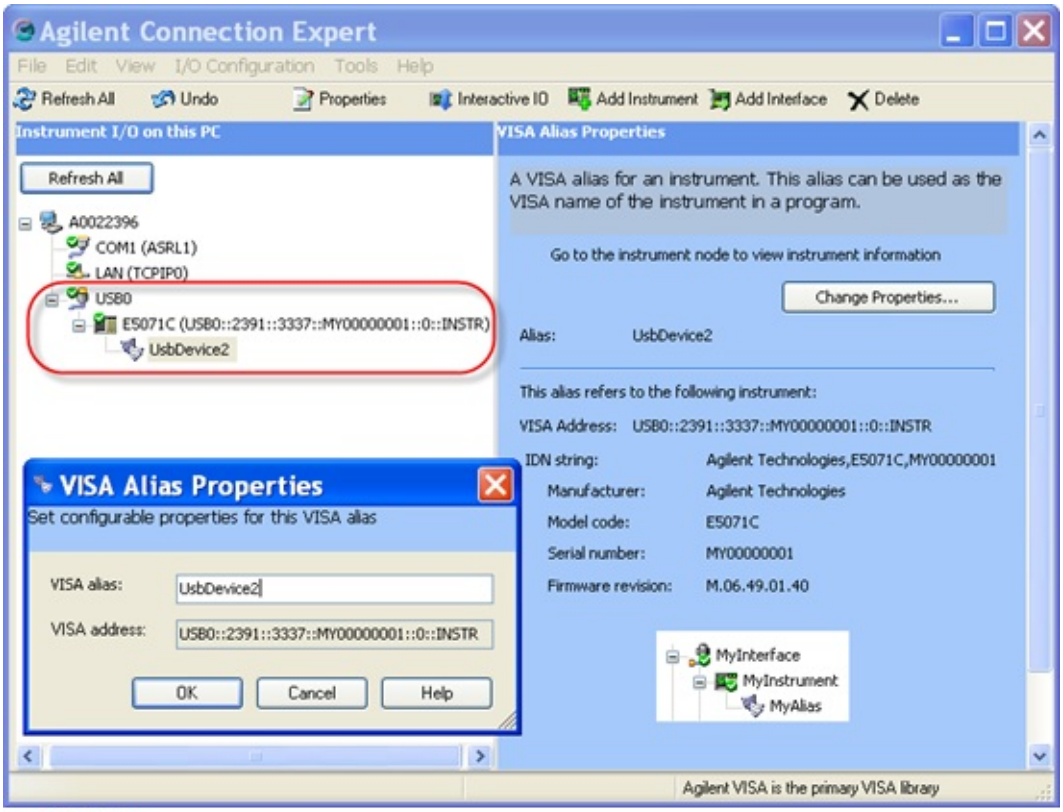
e5071c158

- 127ASCII /
-
- “”

3.

Agilent I/O14.2

1. **> Agilent I/O > Agilent**
2. **PCI/OUSB0I/O**



e5071c145

CVisual Basic

WindowsVisual C++ Visual BasicSICL/VISAE5071CE5071C
SICL VISAAgilent I/OAgilent I/O14.2

SICL/VISA

OPENE5071CENA_USBIF

SICL	id = iopen("ENA_USBIF")
VISA	viOpen(...,"ENA_USBIF",...)

SICL/VISASICLVISA

Agilent VEE

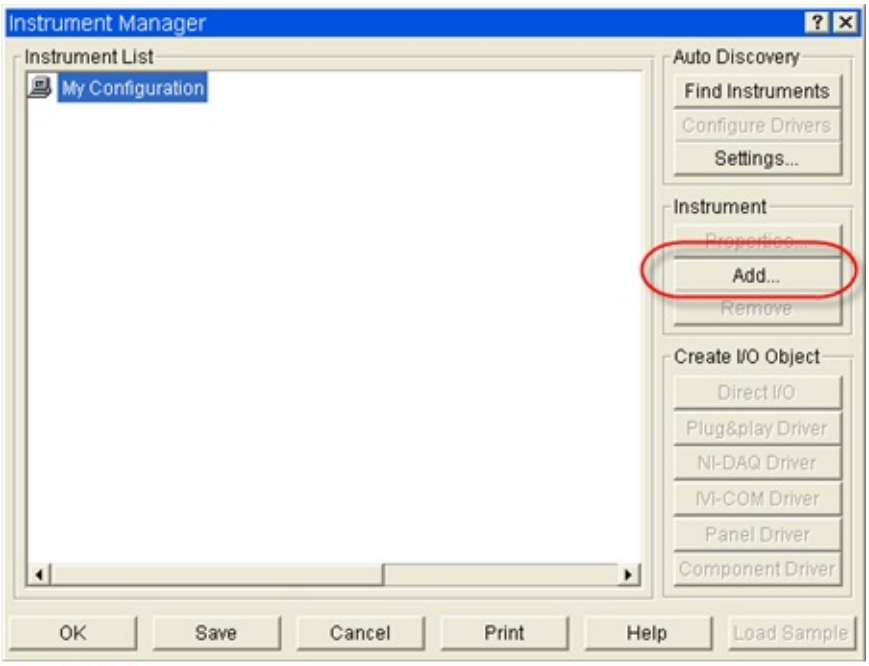
Agilent VEEI/OE5071CE5071CENA_USBIF

PCAgilent VEEWindowsAgilent VEE Pro 7

1. Agilent VEE**I/O**

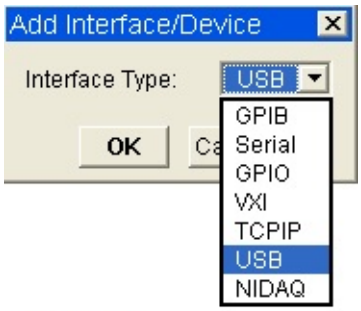
e5071c146

2. **Add.....**



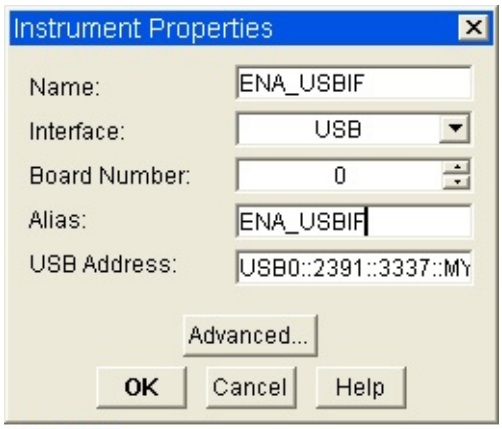
e5071c147

3. **USBOK**



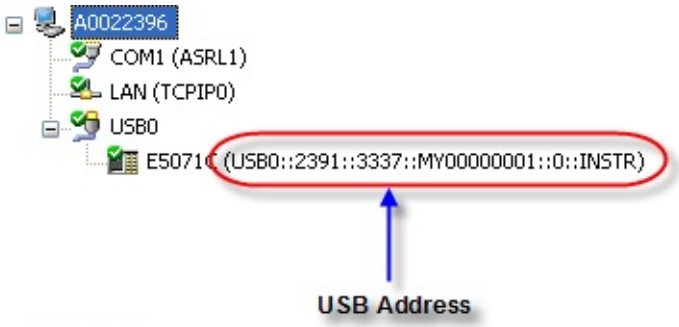
e5071c154

4. ENA_USBIFE5071C_USBUSB
   OK



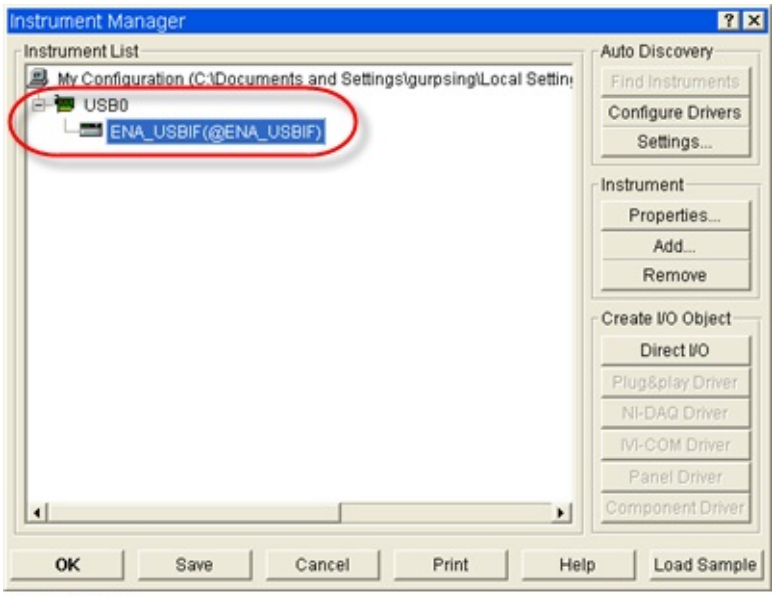
e5071c181

5. AgilentUSB



e5071c159

5. E5071C


Instrument Manager
Instrument List
My Configuration (C:\Documents and Settings\gurpsing\Local Settin
USB0
ENA_USBIF(@ENA_USBIF)
Auto Discovery
Find Instruments
Configure Drivers
Settings...
Instrument
Properties...
Add...
Remove
Create I/O Object
Direct I/O
Plug&play Driver
NI-DAQ Driver
IVI-COM Driver
Panel Driver
Component Driver
OK
Save
Cancel
Print
Help
Load Sample

e5071c182

SCPI

-
-
-

E5071CSCPI

E5071C

E5071C E5071C:

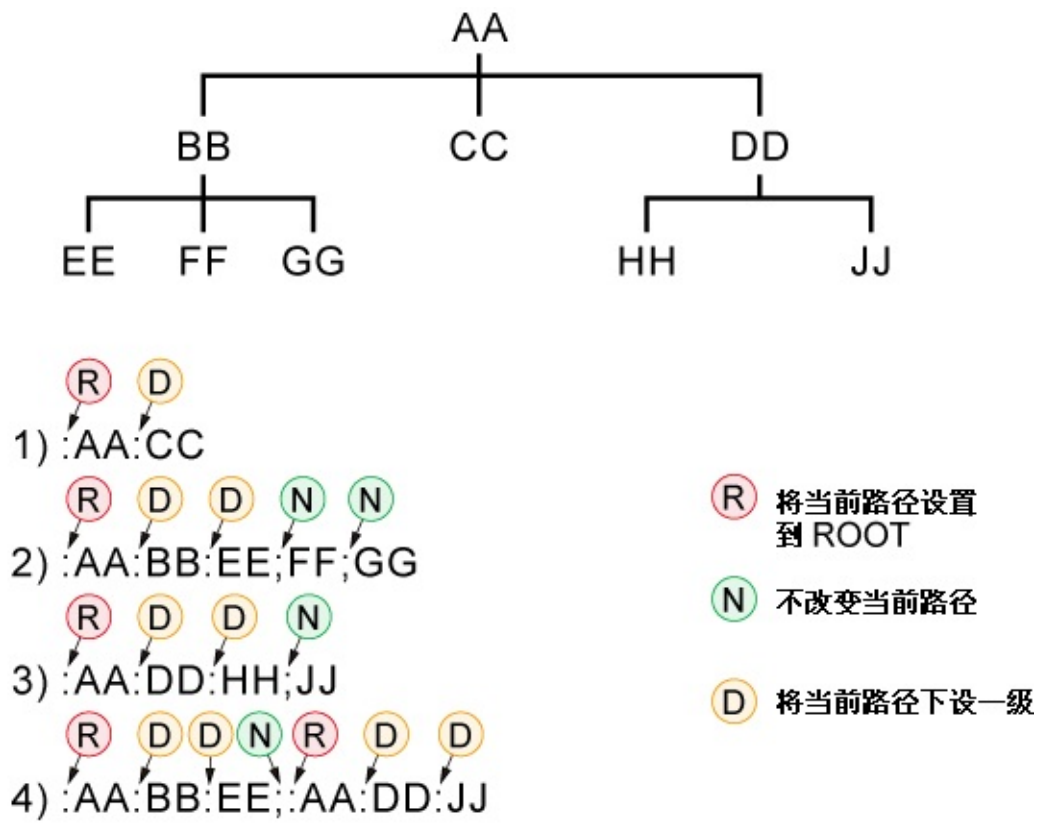
IEEE

IEEE488.2IEEE488.2*

“““”DOS

<>

(:)


AA
BB
CC
DD
EE
FF
GG
HH
JJ
R D
1) :AA:CC
R D D N N
2) :AA:BB:EE;FF;GG
R D D N
3) :AA:DD:HH;JJ
R D D N R D D
4) :AA:BB:EE;:AA:DD:JJ
R 将当前路径设置到 ROOT
N 不改变当前路径
D 将当前路径下设一级

e5071c350

GPIB

/

/

<new line> <^END><new line><^END>
HTBasicOUTPUT

(ASCII32)()

() *CLS :STAT:PRES

OUTPUT 717;"*CLS;:STAT:PRES"

E5071C GPIB

- :SYST:KLOC:KBD
- :SYST:KLOC:MOUS

- /
- .
- .
- .

/

E5071C

:DISP:WIND{1-36}:ACT

:DISP:SPL

:CALC{1-36}:PAR{1-36}:SEL

:CALC{1-36}:PAR:COUN

-
-
-
-
- Z0

313

:CALC{1-36}:PAR:COUN

S

:CALC{1-36}:PAR{1-36}:DEF

S

4

-
-
-
-

:SENS{1-36}:SWE:TYPE

/

:SENS{1-36}:SWE:GEN

/

/

:OUTP

/

	:SENS{1-36}:FREQ:STAR
	:SENS{1-36}:FREQ:STOP
	:SENS{1-36}:FREQ:CENT
	:SENS{1-36}:FREQ:SPAN

:SENS{1-36}:SWE:POIN

	:SENS{1-36}:SWE:TIME
/	:SENS{1-36}:SWE:TIME:AUTO

:SENS{1-36}:SWE:DEL

:SENS{1-36}:BAND

:SENS{1-36}:BWID

:SOUR{1-36}:POW

1

:SOUR{1-36}:POW:PORT:COUP

:SOUR{1-36}:POW:PORT{1-4}

:SOUR{1-36}:POW:SLOP:STAT

:SOUR{1-36}:POW:SLOP

:SENS{1-36}:SEGM:DATA

CSV

:MMEM:LOAD:SEGM

CSV

:MMEM:STOR:SEGM

	:SOUR{1-36}:POW:STAR
	:SOUR{1-36}:POW:STOP
	:SOUR{1-36}:POW:CENT
	:SOUR{1-36}:POW:SPAN

CW

:SENS{1-36}:FREQ

/

/	:SENS{1-36}:AVER
	:SENS{1-36}:AVER:COUN
	:SENS{1-36}:AVER:CLE

Z0

3.01

Z0

:SENS:CORR:IMP

-
-
-

E5071CLCD(echo

19

:DISP:SPL

:CALC{1-36}:PAR:COUN112212

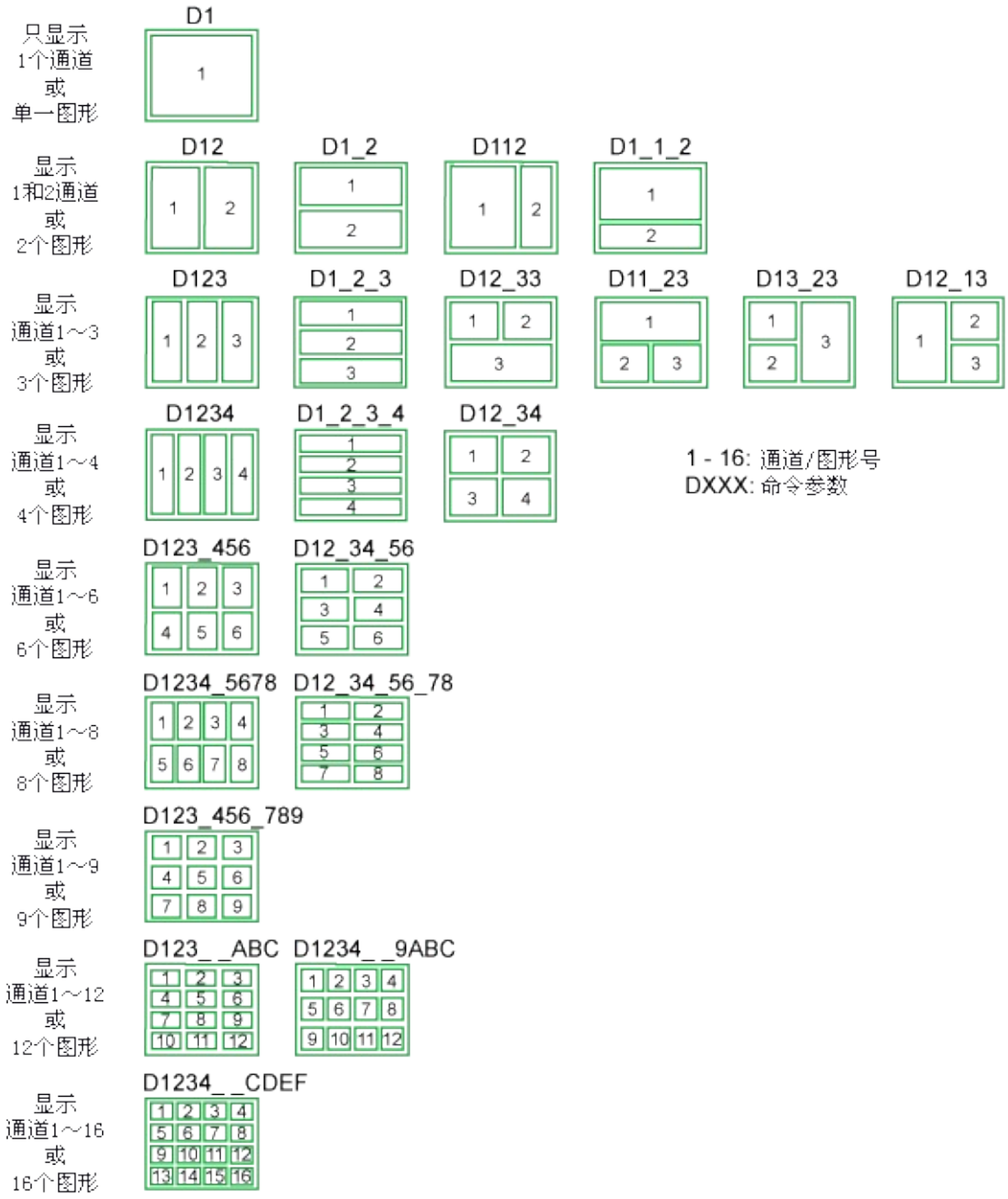
19

:DISP:WIND{1-36}:SPL

:DISP:MAX

:DISP:WIND{1-36}:MAX

/



e5071c314

/

LCD

-
-

-
-
-
-

:DISP:TABL

:DISP:TABL:TYPE

/

LCD

:DISP:SKEY

	:DISP:WIND{1-36}:TRAC{1-36}:STAT
	:DISP:WIND{1-36}:TRAC{1-36}:ANN:MARK:POS:X

:CALC{1-36}:MATH:MEM

:CALC{1-36}:MATH:FUNC

/

:CALC{1-36}:SMO

:CALC{1-36}:SMO:APER

-
  -

-
-
-
- SWR
-
-
-
-
-
-

:CALC{1-36}:FORM

////SWR////

	:DISP:WIND{1-36}:Y:DIV
/	:DISP:WIND{1-36}:TRAC{1-36}:Y:PDIV
	:DISP:WIND{1-36}:TRAC{1-36}:Y:RPOS
	:DISP:WIND{1-36}:TRAC{1-36}:Y:RLEV

:DISP:WIND{1-36}:LAB

/

/

:DISP:WIND{1-36}:TRAC{1-36}:Y:PDIV

Auto Scale/

:DISP:WIND{1-36}:TRAC{1-36}:Y:AUTO

:DISP:ECHO

:DISP:ECHO:CLE

/

/LCD

:DISP:ENAB

/

LCD

:DISP:ANN:FREQ

:DISP:WIND{1-36}:TITL

:DISP:WIND{1-36}:TITL:DATA

/

:DISP:CLOC

:SYST:DATE

:SYST:UPR

/LCD

/LCD

:SYST:BACK

LCD

:DISP:IMAG

	:DISP:COL{1-2}:TRAC{1-36}:DATA
	:DISP:COL{1-2}:TRAC{1-36}:MEM
	:DISP:COL{1-2}:GRAT{1-2}
	:DISP:COL{1-2}:LIM{1-2}
	:DISP:COL{1-2}:BACK

:DISP:COL{1-2}:RES

:MMEM:STOR

:MMEM:LOAD

-
-
-
-
-

- .
- .
- .
- .
- /
- .

:SENS{1-36}:CORR:COLL:CKIT

		:SENS{1-36}:CORR:COLL:METH:OPEN
		:SENS{1-36}:CORR:COLL:METH:SHOR
		:SENS{1-36}:CORR:COLL:METH:THRU
		:SENS{1-36}:CORR:COLL:METH:ERES
1		:SENS{1-36}:CORR:COLL:METH:SOLT1
2		:SENS{1-36}:CORR:COLL:METH:SOLT2
3		:SENS{1-36}:CORR:COLL:METH:SOLT3
4		:SENS{1-36}:CORR:COLL:METH:SOLT4
2TRL		:SENS{1-36}:CORR:COLL:METH:TRL2
3TRL		:SENS{1-36}:CORR:COLL:METH:TRL3
4TRL		:SENS{1-36}:CORR:COLL:METH:TRL4

34343TRL4TRL3TRL4

:SENS{1-36}:CORR:COLL:METH:TYPE?

“”“”“”“”

:SENS{1-36}:CORR:TRIG:FREE

	:SENS{1-36}:CORR:COLL:OPEN
	:SENS{1-36}:CORR:COLL:SHOR
	:SENS{1-36}:CORR:COLL:LOAD
	:SENS{1-36}:CORR:COLL:THRU
	:SENS{1-36}:CORR:COLL:ISOL
TRL	:SENS{1-36}:CORR:COLL:TRLT
TR	:SENS{1-36}:CORR:COLL:TRLR
TRL/	:SENS{1-36}:CORR:COLL:TRLL

*OPC?

	(a)	a		[a]		
	(a)		a	[a]		
	(a-b)				a-b	[a-b]
(a-b)		b	b	b	a-b	[a-b]
(a)		a	a	a		
2(a-b)		a, b	a ,b	a, b	a-b, b-a	[a-b], [b-a]
3(a-b-c)		a, b, c	a, b, c	a, b, c	a-b, b-a a-c, c-a b-c, c-b	[a-b], [b-a] [a-c], [c-a] [b-c], [c-b]
3						
(1-2-3)		1, 2, 3	1, 2, 3	1, 2, 3	1-2, 2-1 1-3, 3-1 [2-3], [3-2]	[1-2], [2-1] [1-3], [3-1] [2-3], [3-2]
(1-2-4)		1, 2, 4	1, 2, 4	1, 2, 4	1-2, 2-1 1-4, 4-1 [2-4], [4-2]	[1-2], [2-1] [1-4], [4-1] [2-4], [4-2]
(1-3-4)		1, 3, 4	1, 3, 4	1, 3, 4	1-3, 3-1 [1-4], [4-1] 3-4, 4-3	[1-3], [3-1] [1-4], [4-1] [3-4], [4-3]
(2-3-4)		2, 3, 4	2, 3, 4	2, 3, 4	2-3, 3-2 [2-4], [4-2] 3-4, 4-3	[2-3], [3-2] [2-4], [4-2] [3-4], [4-3]

4	1, 2, 3, 4	1, 2, 3, 4	1, 2, 3, 4	1-2, 2-1 1-3, 3-1 1-4, 4-1 2-3, 3-2 2-4, 4-2 3-4, 4-3	[1-2], [2-1] [1-3], [3-1] [1-4], [4-1] [2-3], [3-2] [2-4], [4-2] [3-4], [4-3]
4	1, 2, 3, 4	1, 2, 3, 4	1, 2, 3, 4	1-2, 2-1 1-3, 3-1 [1-4], [4-1] [2-3], [3-2] [2-4], [4-2] 3-4, 4-3	[1-2], [2-1] [1-3], [3-1] [1-4], [4-1] [2-3], [3-2] [2-4], [4-2] [3-4], [4-3]

m1amm-n1-2a-bmn

1/2/3/4	:SENS{1-36}:CORR:COLL:SAVE
3/43/4TRL	:SENS{1-36}:CORR:COLL:SIMP:SAVE

3-/4

3.50A.06.503-/4

3-/4

3-/4

3-/43-/43-/4

-
  -
  -
  -
  - N
-
  -
  -
  -
  -
  - /
  -

3-/4TRL

- A.06.50
- 3/4TRL/3/4TRL//
- 3/4TRL3/4

/

/

:SENS{1-36}:CORR:STAT

: :SENS{1-36}:CORR:COLL:SAVE: :SENS{1-36}:CORR:COLL:SIMP:SAVE

Ecal1-/2-/3-/4

EcalEcalE5071CEcal

EcalUSBEcal

Ecal

1	:SENS{1-36}:CORR:COLL:ECAL :SOLT1
2	:SENS{1-36}:CORR:COLL:ECAL :SOLT2
3	:SENS{1-36}:CORR:COLL:ECAL :SOLT3
4	:SENS{1-36}:CORR:COLL:ECAL :SOLT4
	:SENS{1-36}:CORR:COLL:ECAL :ERES
	:SENS{1-36}:CORR:COLL:ECAL :THRU

Ecal

EcalEcalEcal

Ecal/

:SENS{1-36}:CORR:COLL:ECAL:ISOL

Ecal

EcalEcal

:SENS{1-36}:CORR:COLL:ECAL:UCH

EcalEcalE5071C

:SENS:CORR:COLL:ECAL:ORI

:SENS:CORR:COLL:ECAL:PATH

:SENS{1-36}:CORR:TYPE{1-36}?

33SOLT3443TRL3TRL4TRL4TRL

:SENS{1-36}:CORR:COLL:CKIT

:SENS{1-36}:CORR:COLL:CKIT:LAB

:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}:TYPE

:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}:LAB

C0	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :C0
C1	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :C1
C2	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :C2
C3	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :C3
L0	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :L0
L1	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :L1
L2	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :L2
L3	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :L3
	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :DEL
	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :LOSS
Z0	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :Z0

	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}: ARB
	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :FMIN
	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}: FMAX
	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21} :CHAR

C0C3	**L0L3**			**Z0**					
*			**	*	***				
		*	**	*	***				
			**	*	***				
			**	*	***				
			**	*	****				

*

/

:MMEM:STOR:CKIT{1-20}

:MMEM:LOAD:CKIT{1-20}

:SENS{1-36}:CORR:COLL:SUBC

:SENS{1-36}:CORR:COLL:CKIT:ORD

:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}:FMIN

:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}:FMAX

:SENS{1-36}:CORR:COLL:CKIT:ORD:OPEN

:SENS{1-36}:CORR:COLL:CKIT:ORD:SHOR

:SENS{1-36}:CORR:COLL:CKIT:ORD:LOAD

:SENS{1-36}:CORR:COLL:CKIT:ORD:THRU

:SENS{1-36}:CORR:COLL:CKIT:ORD:TRLT

:SENS{1-36}:CORR:COLL:CKIT:ORD:TRLR

TRL/

:SENS{1-36}:CORR:COLL:CKIT:ORD:TRLL

:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}:CHAR

:MMEM:STOR

:MMEM:LOAD

/

/

:SENS{1-36}:CORR:COEF

:SENS{1-36}:CORR:COEF:METH:ERES

:SENS{1-36}:CORR:COEF:METH:OPEN

:SENS{1-36}:CORR:COEF:METH:SHOR

:SENS{1-36}:CORR:COEF:METH:SOLT1

:SENS{1-36}:CORR:COEF:METH:SOLT2

:SENS{1-36}:CORR:COEF:METH:SOLT3

:SENS{1-36}:CORR:COEF:METH:SOLT4

:SENS{1-36}:CORR:COEF:METH:THRU

:SENS{1-36}:CORR:COEF:SAVE

	ES	**ER**	**ED**	**EL**	**ET**	**EX**
		*	*			
		*	*			
					*	*
	*	*	*		*	*

1	*	*	*			
2	*	*	*	*	*	*
3	*	*	*	*	*	*
4	*	*	*	*	*	*
2TRL	*	*	*	*	*	
3TRL	*	*	*	*	*	
4TRL	*	*	*	*	*	

23,

1.

: SENS{1-36}:CORR:COEF:METH:xxxx

2.

: SENS{1-36}:CORR:COEF

3.

: SENS{1-36}:CORR:COEF:SAVE

:SENS{1-36}:CORR:OFFS:COLL:OPEN

:SENS{1-36}:CORR:OFFS:COLL:CLE

:SENS{1-36}:CORR:OFFS:CLE

- .
- .
- .
- /

E5071C

E5071CGPIBUSB/GPIBE5071C

E5071CGPIB

:SYST:COMM:GPIB:PMET:ADDR

50 MHz

E5071C

A	:SOUR:POW:PORT:CORR:COLL:ASEN:RCF
B	:SOUR:POW:PORT:CORR:COLL:BSEN:RCF

A	:SOUR:POW:PORT:CORR:COLL:TABL:ASEN:DATA
B	:SOUR:POW:PORT:CORR:COLL:TABL:BSEN:DATA

CSV

A	:MMEM:LOAD:ASCF
B	:MMEM:LOAD:BSCF

A	:MMEM:STOR:ASCF
B	:MMEM:STOR:BSCF

/ /

DUT

:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:TABL:LOSS:DATA

CSV

:MMEM:LOAD:PLOS{1-4}

:MMEM:STOR:PLOS{1-4}

/

:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:TABL:LOSS

E5071C100

:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:AVER

:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:NTOL

:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL

*OPC

/

:SOUR{1-36}:POW:PORT{1-4}:CORR

:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL

/

6416 4/

:SOUR{1-36}:POW:PORT{1-4}:CORR:DATA

- .
- .
- /

E5071C

- THRU

REC

:SENS{1-36}:CORR:REC{1-4}:COLL:ACQ

-
-
-

/

:SENS{1-36}:CORR:REC{1-4}

:SENS{1-36}:CORR:REC{1-4}:COLL:ACQ

-
-

E5071C

:SENS{1-36}:OFFS

- :SENS{1-36}:CORR:OFFS:COLL:METH:SMIX2
- :SENS{1-36}:CORR:OFFS:COLL:METH:SOLT1

“SMIX2”2“SOLT1”1

“SMIX2”

:SENS{1-36}:CORR:OFFS:COLL:OPEN

:SENS{1-36}:CORR:OFFS:COLL:OPEN

- :SENS{1-36}:CORR:OFFS:COLL:OPEN
- :SENS{1-36}:CORR:OFFS:COLL:SHOR
- :SENS{1-36}:CORR:OFFS:COLL:LOAD
- :SENS{1-36}:CORR:OFFS:COLL:THRU

:SENS{1-36}:CORR:OFFS:COLL:PMET

1

:SENS{1-36}:CORR:OFFS:COLL:SAVE

/

:SENS{1-36}:CORR:STAT

Ecal

Ecal1

Ecal

1	:SENS{1-36}:CORR:OFFS:COLL:ECAL:SOLT1
	:SENS{1-36}:CORR:OFFS:COLL:ECAL:SMIX2

1

EcalEcal

EcalEcalEcal

-
-

E5071CNErEsEdEtEx

- C?C!
-

Ex0

20

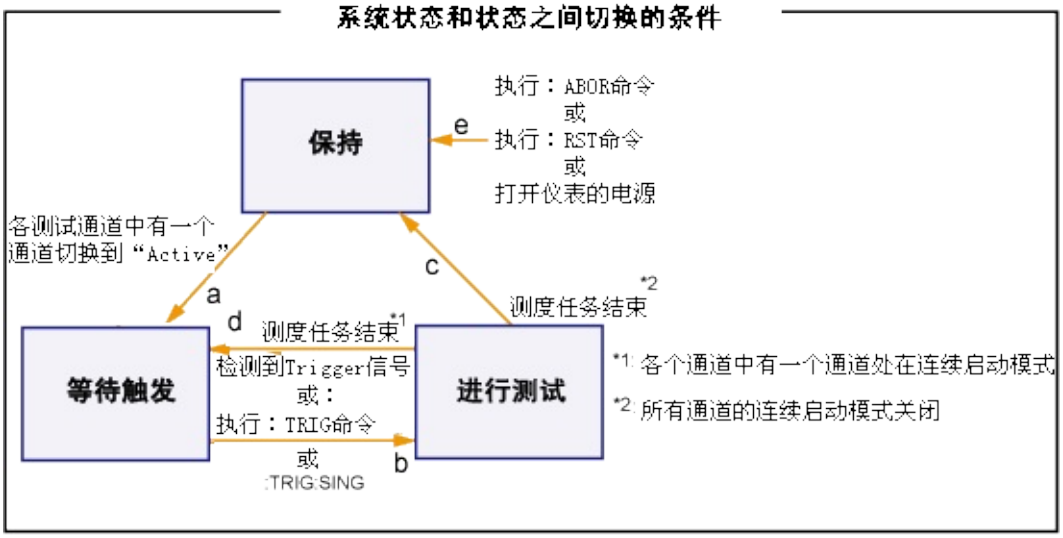
:SENS{1-36}:CORR:COLL:PART:SAVE

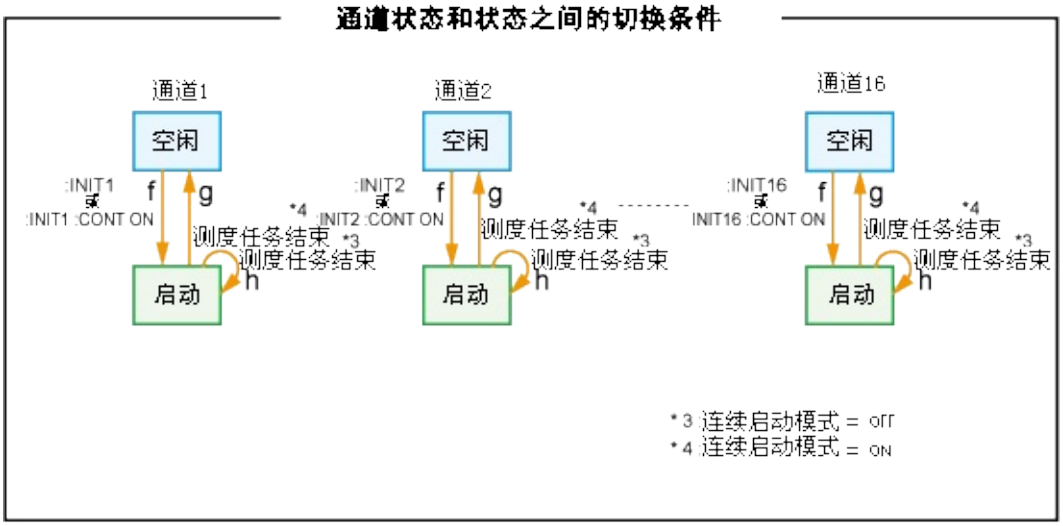
28

-
-
-
-
-

-
-
-

/“”“”“”“”“”“”


系统状态和状态之间切换的条件
保持
e
执行：ABOR命令
或
执行：RST命令
或
打开仪表的电源
各测试通道中有一个
通道切换到“Active”
a
c
测度任务结束*2
d
测度任务结束*1
等待触发
检测到Trigger信号
或：
执行：TRIG命令
或
:TRIG:SING
b
进行测试
*1: 各个通道中有一个通道处在连续启动模式
*2: 所有通道的连续启动模式关闭


通道状态和状态之间的切换条件
通道1
空闲
:INIT1
或
:INIT1 :CONT ON
f
g
*4
测度任务结束
*3
测度任务结束
h
启动
通道2
空闲
:INIT2
或
:INIT2 :CONT ON
f
g
*4
测度任务结束
*3
测度任务结束
h
启动
通道16
空闲
:INIT16
或
INIT16 :CONT ON
f
g
*4
测度任务结束
*3
测度任务结束
h
启动
*3 连续启动模式 = off
*4 连续启动模式 = on

e5071c312

“”

“e”“”“”1“”“”
“”“”

- :ABOR
- *RST

“”“”“”f“”“”a”

“”

“”“”“”B”

- :TRIG
- :TRIG:SING

:TRIG:SOUR

	Ext Trig
	*TRG
	Trigger >**Trigger**

“”

“”:SENS{1-36}:SWE:DEL“”“”

“”“c”

“”“d”

“”

“”“f”

- :INIT{1-36}
- :INIT{1-36}:CONT("ON" specified)

“”

“”

: INIT{1-36}:CONT

“”“g”

“”“h”

-
-

1. :INIT{1-36}:CONT
2. :TRIG:SOUR “Internal”

1. :INIT{1-36}:CONT
2. :TRIG:SOUR “Bus Trigger”
3.

	***OPC?**	
*TRG		
:TRIG		
:TRIG:SING		

4. 3

-
- :TRIG:SING Command
-

E5071C

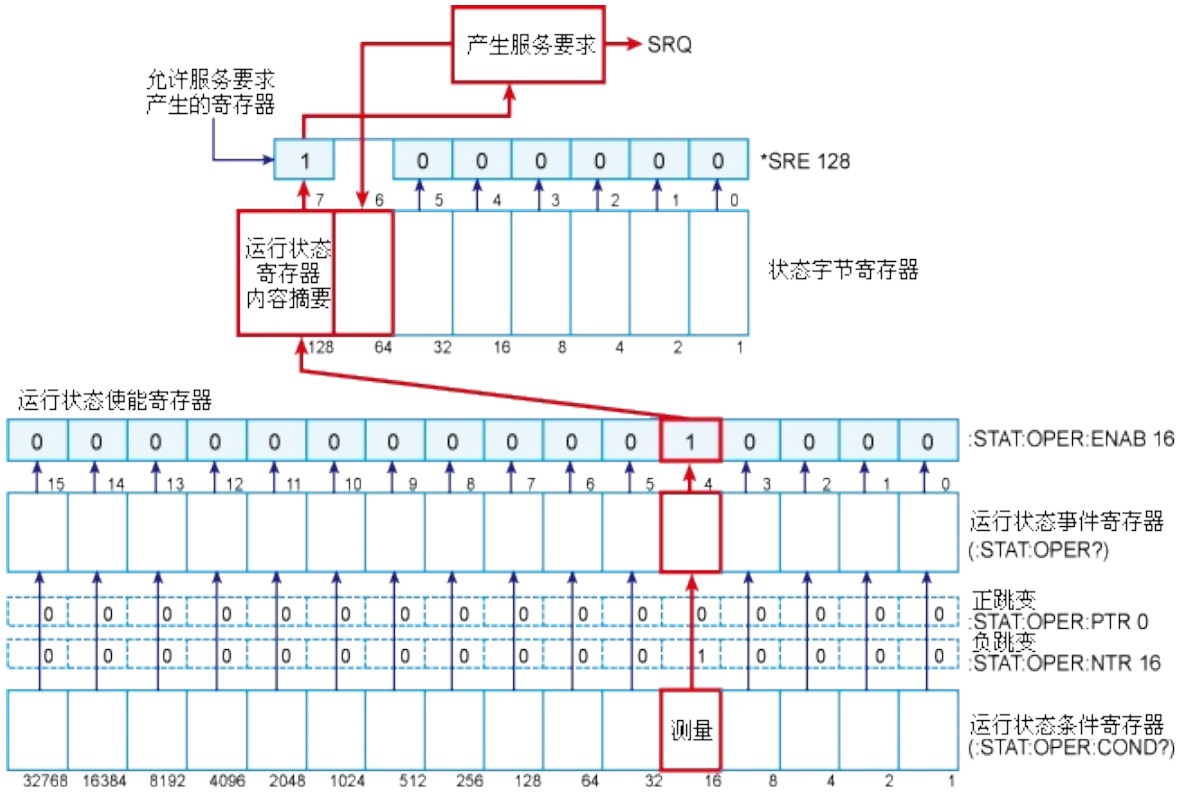
SRQ

SRQ

- *SRE
- :STAT:OPER:ENAB
- :STAT:OPER:PTR
- :STAT:OPER:NTR

SRQ

1. E5071C4110SRQ
2.
3. SRQ
4. SRQ



e5071c356

SRQ

:TRIG:SING

: TRIG:SING *OPC

OPC?

HTBasicWAIT

-
-

/

/

:TRIG:POIN

/

:TRIG:EXT:DEL

-
-

/

:TRIG:AVER

×

/

-
-
-
-

-
- ASCII
-
-
-

-

ASCIIIEEE 64IEEE 32

ASCII

- :CALC{1-36}:BLIM:REP?
- :CALC{1-36}:DATA:FDAT
- :CALC{1-36}:DATA:FMEM
- :CALC{1-36}:DATA:SDAT
- :CALC{1-36}:DATA:SMEM
- :CALC{1-36}:FUNC:DATA?
- :CALC{1-36}:LIM:DATA
- :CALC{1-36}:LIM:REP?
- :CALC{1-36}:LIM:REP:ALL?
- :CALC{1-36}:RLIM:DATA
- :CALC{1-36}:RLIM:REP?
- :SENS{1-36}:FREQ:DATA?
- :SENS{1-36}:SEGM:DATA
- :SOUR:POW:PORT:CORR:COLL:TABL:ASEN:DATA
- :SOUR:POW:PORT:CORR:COLL:TABL:BSEN:DATA
- :SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:TABL:LOSS
- :SOUR{1-36}:POW:PORT{1-4}:CORR:DATA

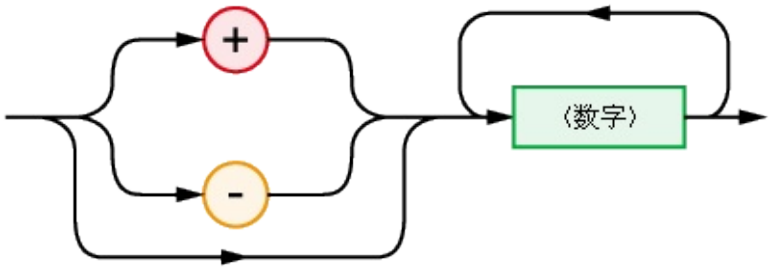
:FORM:DATA

:SYST:PRES*RST

ASCII

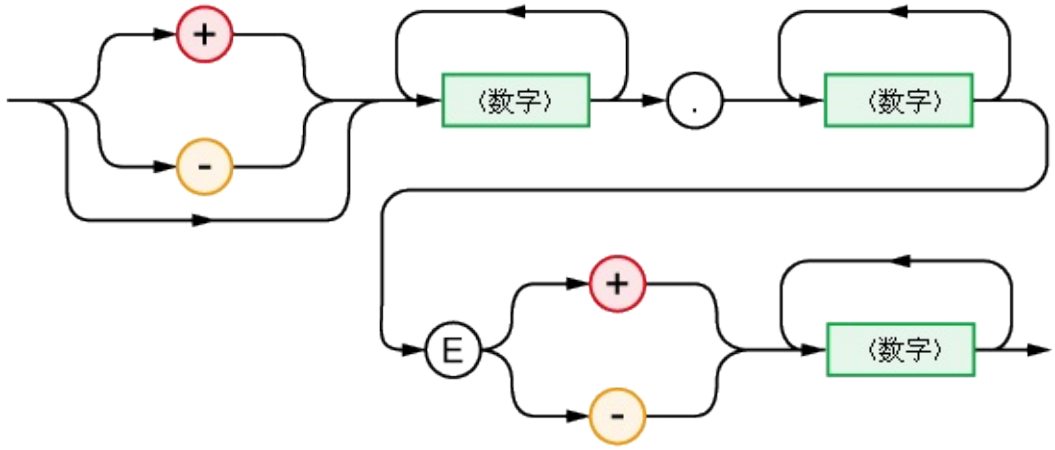
ASCIIASCIIIEEE 488.2

201“+201”“201”



e5071c351

1000“+1.00000000000E+003”

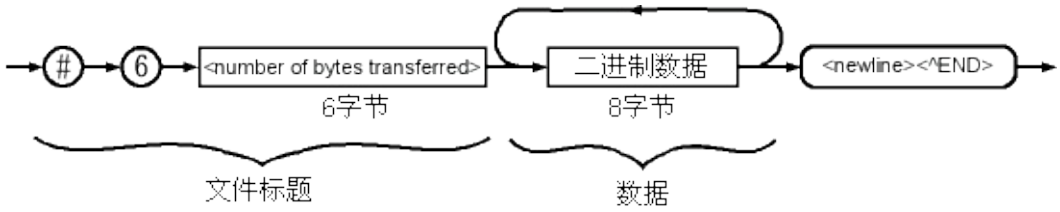


e5071c352

IEEE 64IEEE 32

IEEE 64

IEEE 64

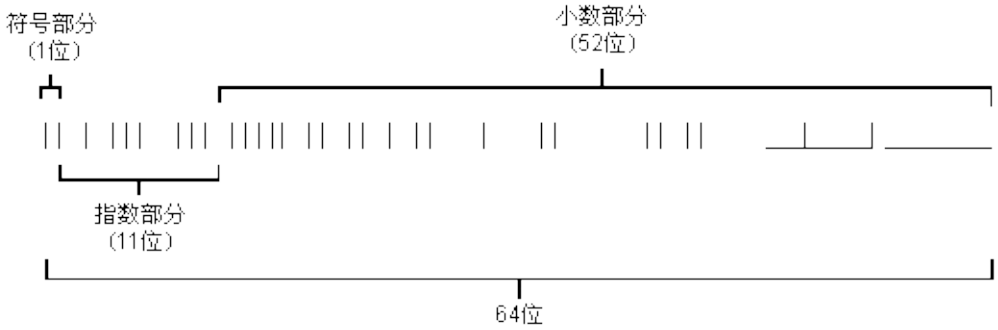


4287ape024

(#)<number of bytes transferred>6<number transferred>8<number of bytes line>^END^

IEEE 754 64

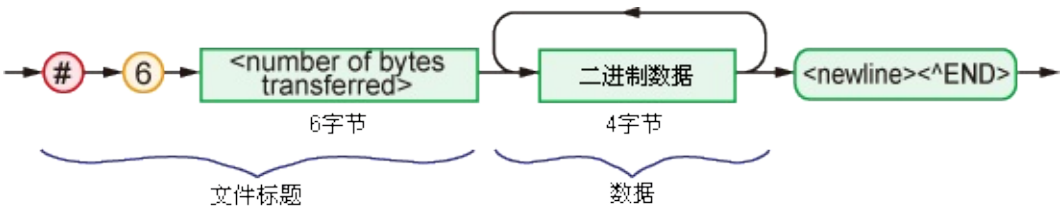
64



IEEE 32

IEEE 32

IEEE 32

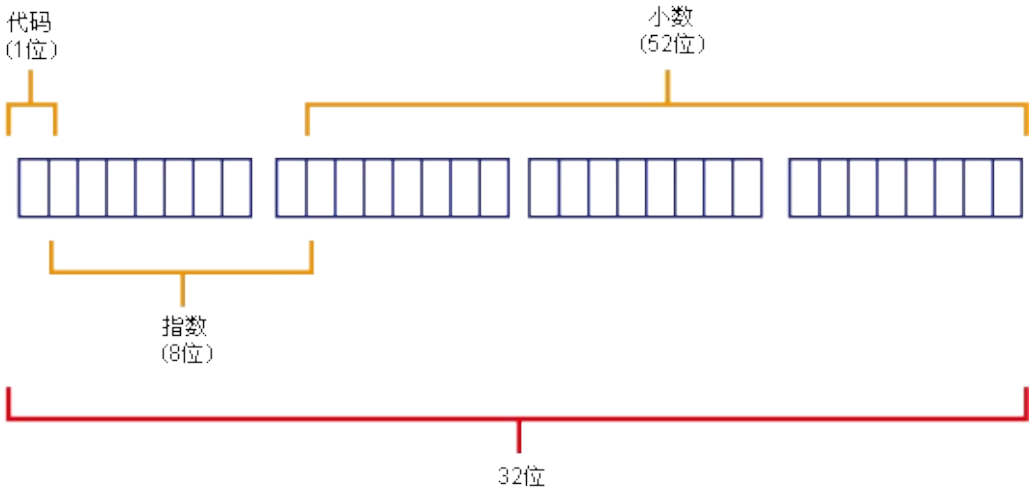


e5071c439

(#)<number of bytes transferred><number transferred>4<number of bytes line>^END^

IEEE 754 32

32



e5071c440

NORMal

MSB6432

SWAPped

LSB6432

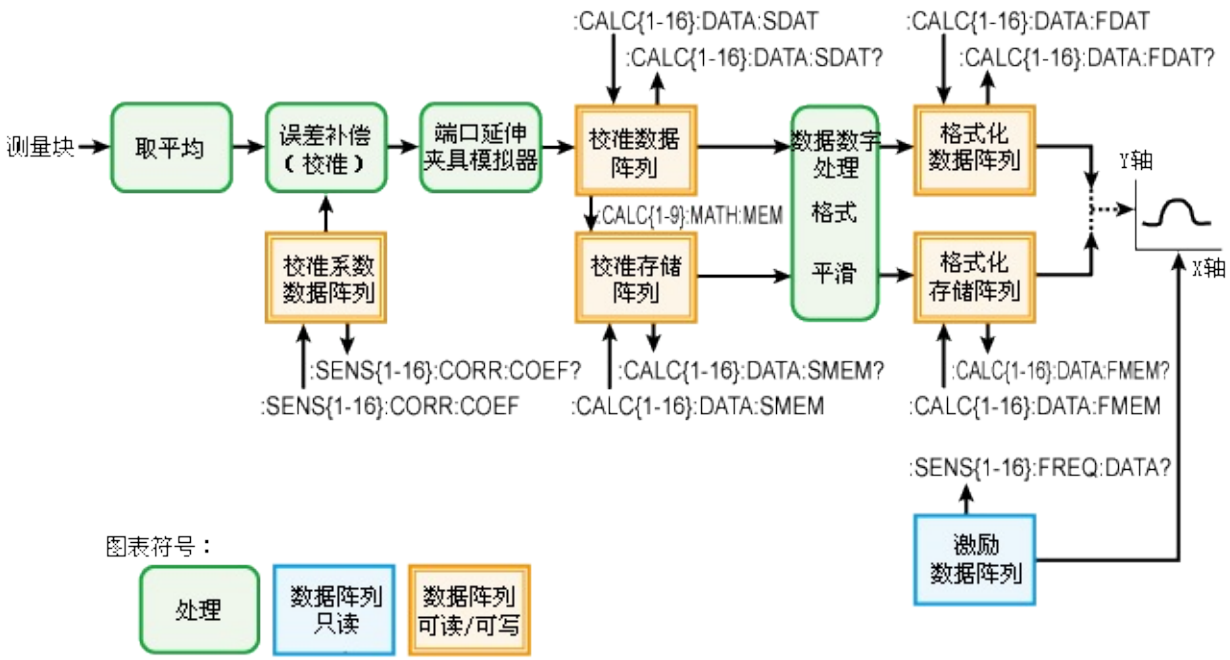
:FORM:BORD

:SYST:PRES*RST

-
-

E5071C

E5071C



e5071c442

S/

8199(9 * 9 = 81)/

:CALC{1-36}:DATA:SDAT

:CALC{1-36}:MATH:MEM

8199(9 * 9 = 81)/

:CALC{1-36}:DATA:SMEM

		0
		0
		0
/		
(R+jX)		
(G+jB)		

/		
		0
SWR	SWR	0
		0
		0
		0

8199(9 * 9 = 81)/

:CALC{1-36}:DATA:FDAT

8199(9 * 9 = 81)/

:CALC{1-36}:DATA:FMEM

1616

:SENS{1-36}:FREQ:DATA?

16

:SENS{1-36}:CORR:COEF

SENS{1-36}:CORR:COEF:SAVE)

-
-
-

ASCII

Ascii

-
-

/LCD/

/ASCII

ASCII(write_a.htb)ASCII(write_b.htb)write_a.htb
write_b.htb

1 :MMEM:STOR:FDAT11

Ascii

-
-
-
-
-
-
-

- .
- /
- /
- .
- .

/

:CALC{1-36}:MARK{1-10}

/

:CALC{1-36}:MARK:REF

:CALC{1-36}:MARK{1-10}:X

:CALC{1-36}:MARK{1-10}:Y?

-
-
-
-

:CALC{1-36}:MARK:FUNC:DOM

	:CALC{1-36}:MARK:FUNC:DOM :STAR
	:CALC{1-36}:MARK:FUNC:DOM :STOP

:CALC{1-36}:MARK:FUNC:DOM:COUP

8

-
-
- 3
  -
  -
  -
- 3
  -
  -

•

:CALC{1-36}:MARK{1-10}:FUNC:TYPE

peak excursion

T

	:CALC{1-36}:MARK{1-10}:FUNC:PEXC
	:CALC{1-36}:MARK{1-10}:FUNC:PPOL

	:CALC{1-36}:MARK{1-10}:FUNC:TARG
	:CALC{1-36}:MARK{1-10}:FUNC:TTR

:CALC{1-36}:MARK{1-10}:FUNC:EXEC

:CALC{1-36}:MARK{1-10}:FUNC:TRAC

:CALC{1-36}:FUNC:DOM

	:CALC{1-36}:FUNC:DOM:STAR
	:CALC{1-36}:FUNC:DOM:STOP

:CALC{1-36}:FUNC:DOM:COUP

5

-
-
-
-
-

3

-
-
-

:CALC{1-36}:FUNC:TYPE

peak excursion

	:CALC{1-36}:FUNC:PEXC
	:CALC{1-36}:FUNC:PPOL

	:CALC{1-36}:FUNC:TARG
	:CALC{1-36}:FUNC:TTR

:CALC{1-36}:FUNC:EXEC

:CALC{1-36}:FUNC:DATA?

:CALC{1-36}:FUNC:POIN?

-
-
-
-

E5071C

- BW=-
- ={+}/2
- QQ=cent/BW
-

:CALC{1-36}:MARK{1-10}:BWID:THR

:CALC{1-36}:MARK{1-10}:BWID:DATA?

LCD

:CALC{1-36}:MARK:BWID

E5071C

-
-
-

:CALC{1-36}:MST:DATA?

LCD

:CALC{1-36}:MST

-
-
-
-
- 4/
- 3/4
- 3/4
- 3/4
-

-
-
-
- 4/
- 3/4
- 3/4
- 3/4

:CALC{1-36}:FSIM:STAT

(DUT)

/

:CALC{1-36}:FSIM:SEND:PMC:STAT

"None"

:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}

C	:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:PAR:C
G	:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:PAR:G
L	:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:PAR:L
R	:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:PAR:R

:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:USER:FIL

50Ω

/

:CALC{1-36}:FSIM:SEND:ZCON:STAT

ZO50Ω

Z0

:CALC{1-36}:FSIM:SEND:ZCON:PORT{1-4}:Z0

:CALC{1-36}:FSIM:SEND:ZCON:PORT{1-4}:REAL

:CALC{1-36}:FSIM:SEND:ZCON:PORT{1-4}:IMAG

(DUT)

/

:CALC{1-36}:FSIM:SEND:DEEM:STAT

"None"

:CALC{1-36}:FSIM:SEND:DEEM:PORT{1-4}

:CALC{1-36}:FSIM:SEND:DEEM:PORT{1-4}:USER:FIL

4/

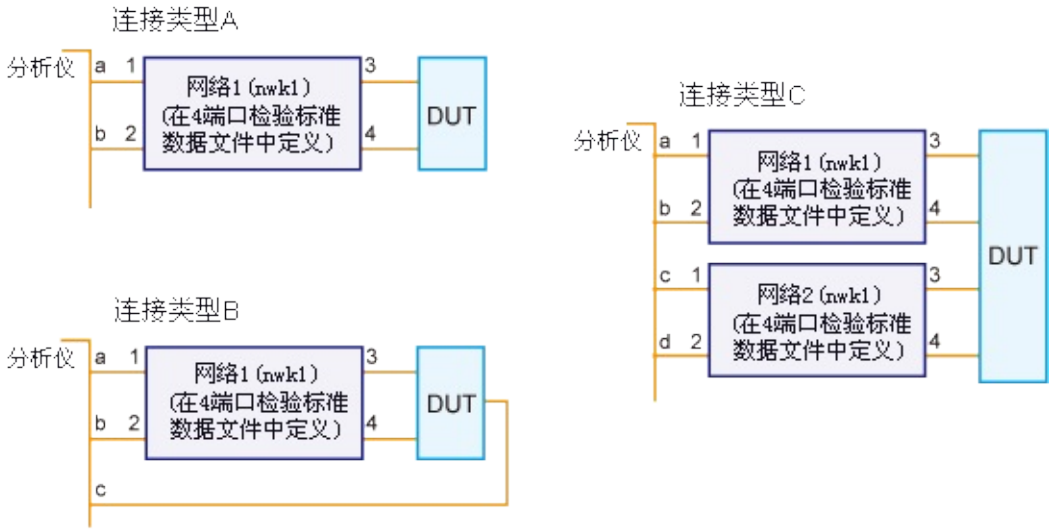
4/(DUT)4

/

4/

:CALC{1-36}:FSIM:EMB:STAT

:CALC{1-36}:FSIM:EMB:TYPE



e5071c313

a~d

A	:CALC{1-36}:FSIM:EMB:TOP:A:PORT
B	:CALC{1-36}:FSIM:EMB:TOP:B:PORT
C	:CALC{1-36}:FSIM:EMB:TOP:C:PORT

/

:CALC{1-36}:FSIM:EMB:NETW{1-2}:TYPE

/
:CALC{1-36}:FSIM:EMB:NETW{1-2}:FIL

-

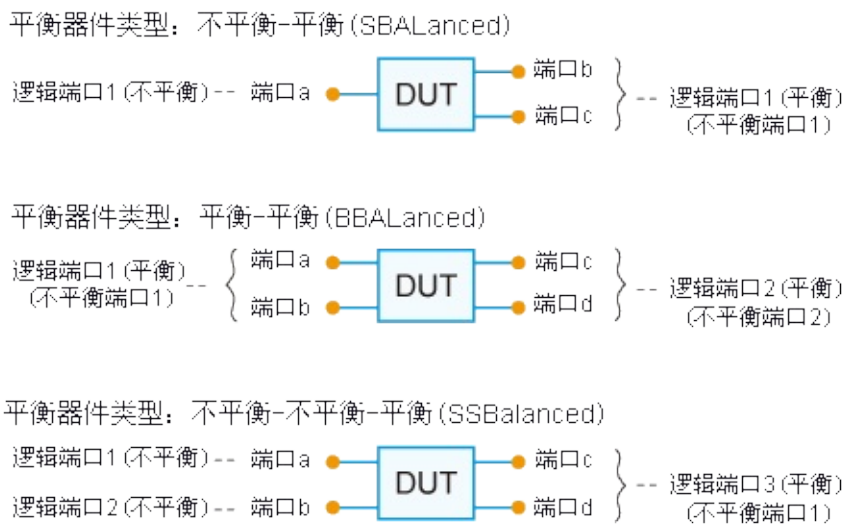
--SCMRR

/

--

:CALC{1-36}:FSIM:BAL:PAR{1-36}:STAT

:CALC{1-36}:FSIM:BAL:DEV



e5071c302

a~d

-(SBALanced)	:CALC{1-36}:FSIM:BAL:TOP :SBAL
-(BBALanced)	:CALC{1-36}:FSIM:BAL:TOP :BBAL

--(SSBalanced)	:CALC{1-36}:FSIM:BAL:TOP :SSB

-	:CALC{1-36}:FSIM:BAL:PAR{1-36} :SBAL
-	:CALC{1-36}:FSIM:BAL:PAR{1-36} :BBAL
--	:CALC{1-36}:FSIM:BAL:PAR{1-36} :SSB

--(DUT)

/

:CALC{1-36}:FSIM:BAL:DMC:STAT

"None"

:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}

C	:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:PAR:C
G	:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:PAR:G
L	:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:PAR:L
R	:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:PAR:R

:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:USER:FIL

-

/

:CALC{1-36}:FSIM:BAL:DZC:STAT

Z0

:CALC{1-36}:FSIM:BAL:DZC:BPOR{1-2}:Z0

:CALC{1-36}:FSIM:BAL:DZC:BPOR{1-2}:REAL

:CALC{1-36}:FSIM:BAL:DZC:BPOR{1-2}:IMAG

-
-
-

-
-

/

:CALC{1-36}:TRAN:TIME:STAT

/

:CALC{1-36}:TRAN:TIME

/

:CALC{1-36}:TRAN:TIME:STIM

:CALC{1-36}:TRAN:TIME:LPFR

β	:CALC{1-36}:TRAN:TIME:KBES
	:CALC{1-36}:TRAN:TIME:IMP:WIDT
	:CALC{1-36}:TRAN:TIME:STEP:RTIM

// *β*

β	13	6	0

	:CALC{1-36}:TRAN:TIME:STAR
	:CALC{1-36}:TRAN:TIME:STOP
	:CALC{1-36}:TRAN:TIME:CENT
	:CALC{1-36}:TRAN:TIME:SPAN

I

:CALC{1-36}:FILT:TIME:STAT

:CALC{1-36}:FILT:TIME

:CALC{1-36}:FILT:TIME:SHAP

	:CALC{1-36}:FILT:TIME:STAR
	:CALC{1-36}:FILT:TIME:STOP
	:CALC{1-36}:FILT:TIME:CENT
	:CALC{1-36}:FILT:TIME:SPAN

S

-
-
-
-
- S1/S

:CALC{1-36}:CONV

:CALC{1-36}:CONV:FUNC

-
-

- .
- /__
- /__
- /“”_____

E5071C100//

:CALC{1-36}:LIM:DATA

CSV

:MMEM:LOAD:LIM

:MMEM:STOR:LIM

/

:CALC{1-36}:LIM:DISP

/

:CALC{1-36}:LIM

/“”

“”“”

:DISP:FSIG

-
-
-
-
-

	:CALC{1-36}:LIM:REP?
	:CALC{1-36}:LIM:REP:POIN?

:CALC{1-36}:LIM:FAIL?

{1-16}1~14140:1:111414{1-16}15~160:1:1
15216

01

11011

{1-16}	
	:STAT:QUES:LIM:CHAN{1-16}:COND?
	:STAT:QUES:LIM:CHAN{1-16}?
{1-16}	
	:STAT:QUES:LIM:CHAN{1-16}:ECH:COND?
	:STAT:QUES:LIM:CHAN{1-16}:ECH?

E5071C36/16

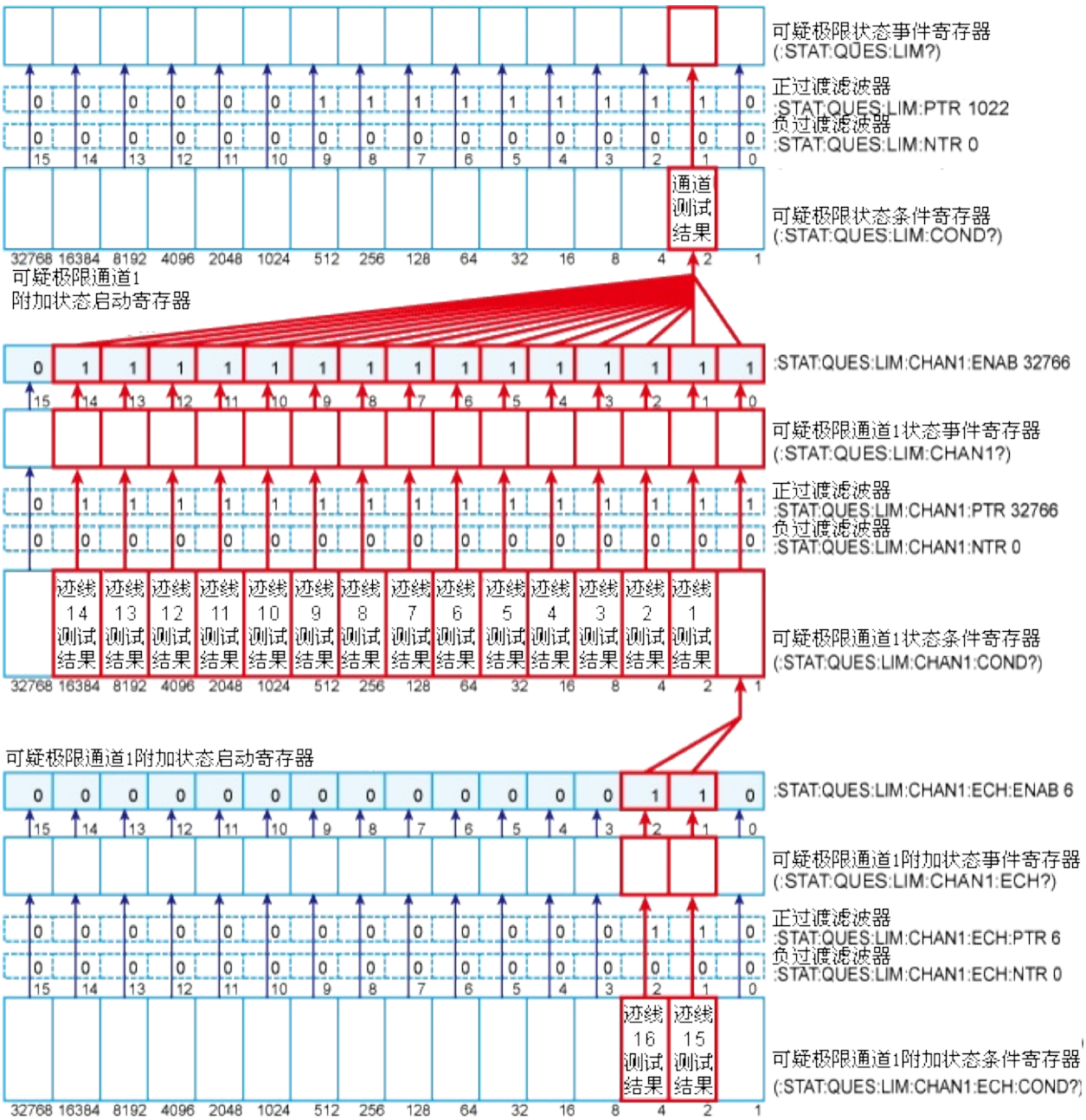
1~14140:1:1114141~290:1:115216

*CLS0“”1

11011

	:STAT:QUES:LIM:COND?
	:STAT:QUES:LIM?
	:STAT:QUES:LIM:ELIM:COND?
	:STAT:QUES:LIM:ELIM?

1


可疑极限状态事件寄存器
(:STAT:QUES:LIM?)
正过渡滤波器
:STAT:QUES:LIM:PTR 1022
负过渡滤波器
:STAT:QUES:LIM:NTR 0
通道测试结果
可疑极限状态条件寄存器
(:STAT:QUES:LIM:COND?)
可疑极限通道1
附加状态启动寄存器
:STAT:QUES:LIM:CHAN1:ENAB 32766
可疑极限通道1状态事件寄存器
(:STAT:QUES:LIM:CHAN1?)
正过渡滤波器
:STAT:QUES:LIM:CHAN1:PTR 32766
负过渡滤波器
:STAT:QUES:LIM:CHAN1:NTR 0
迹线14测试结果
迹线13测试结果
迹线12测试结果
迹线11测试结果
迹线10测试结果
迹线9测试结果
迹线8测试结果
迹线7测试结果
迹线6测试结果
迹线5测试结果
迹线4测试结果
迹线3测试结果
迹线2测试结果
迹线1测试结果
可疑极限通道1状态条件寄存器
(:STAT:QUES:LIM:CHAN1:COND?)
可疑极限通道1附加状态启动寄存器
:STAT:QUES:LIM:CHAN1:ECH:ENAB 6
可疑极限通道1附加状态事件寄存器
(:STAT:QUES:LIM:CHAN1:ECH?)
正过渡滤波器
:STAT:QUES:LIM:CHAN1:ECH:PTR 6
负过渡滤波器
:STAT:QUES:LIM:CHAN1:ECH:NTR 0
迹线16测试结果
迹线15测试结果
可疑极限通道1附加状态条件寄存器
(:STAT:QUES:LIM:CHAN1:ECH:COND?)

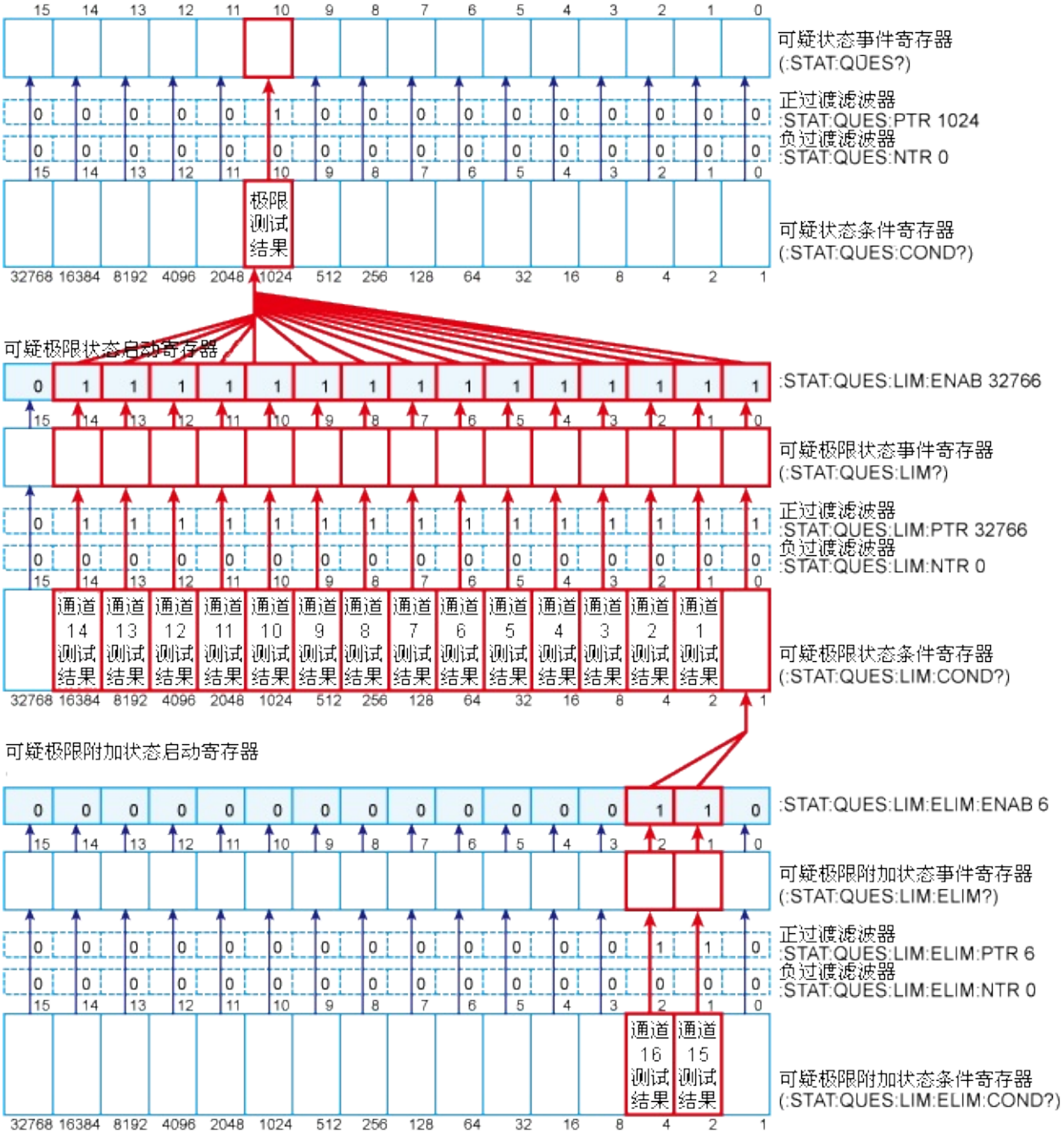
e5071c437

100:1:

*CLS100“”1

1011001101

	:STAT:QUES:COND?
	:STAT:QUES?


可疑状态事件寄存器
(:STAT:QUES?)
正过渡滤波器
:STAT:QUES:PTR 1024
负过渡滤波器
:STAT:QUES:NTR 0
极限测试结果
可疑状态条件寄存器
(:STAT:QUES:COND?)
可疑极限状态启动寄存器
:STAT:QUES:LIM:ENAB 32766
可疑极限状态事件寄存器
(:STAT:QUES:LIM?)
正过渡滤波器
:STAT:QUES:LIM:PTR 32766
负过渡滤波器
:STAT:QUES:LIM:NTR 0
通道14测试结果
通道13测试结果
通道12测试结果
通道11测试结果
通道10测试结果
通道9测试结果
通道8测试结果
通道7测试结果
通道6测试结果
通道5测试结果
通道4测试结果
通道3测试结果
通道2测试结果
通道1测试结果
可疑极限状态条件寄存器
(:STAT:QUES:LIM:COND?)
可疑极限附加状态启动寄存器
:STAT:QUES:LIM:ELIM:ENAB 6
可疑极限附加状态事件寄存器
(:STAT:QUES:LIM:ELIM?)
正过渡滤波器
:STAT:QUES:LIM:ELIM:PTR 6
负过渡滤波器
:STAT:QUES:LIM:ELIM:NTR 0
通道16测试结果
通道15测试结果
可疑极限附加状态条件寄存器
(:STAT:QUES:LIM:ELIM:COND?)

e5071c438

-
-

-
-
-
-
-
-
-
- /
- /
- /VBA

“D”“/”“\”

2

-
- AD

4

-
-
- /
- /

:MMEM:STOR:STYP

//
:MMEM:STOR:SALL

:MMEM:STOR

:MMEM:LOAD

E5071Cautorec.staA:autorec.sta

AD

:MMEM:STOR:CHAN

:MMEM:LOAD:CHAN

:MMEM:STOR:CHAN:CLE

CSVComma Separated Value

:MMEM:STOR:FDAT

- :MMEM:STOR:SNP:TYPE:S1P
- :MMEM:STOR:SNP:TYPE:S2P
- :MMEM:STOR:SNP:TYPE:S3P
- :MMEM:STOR:SNP:TYPE:S4P

:MMEM:STOR:SNP:FORM

:MMEM:STOR:SNP

LCD.bmp.png.

:MMEM:STOR:IMAG

[Capture]

CSVComma Separated Value

:MMEM:STOR:SEGM

:MMEM:LOAD:SEGM

CSVComma Separated Value

:MMEM:STOR:LIM

:MMEM:LOAD:LIM

/

CSVComma Separated ValueE5071C

- :MMEM:STOR:ASCF
- :MMEM:STOR:BSCF

- :MMEM:LOAD:ASCF
- :MMEM:LOAD:BSCF

/

CSVComma Separated Value

:MMEM:STOR:PLOS{1-4}

:MMEM:LOAD:PLOS{1-4}

/VBA

VBA

VBAVBA

:MMEM:STOR:PROG

VBAVBA/

:MMEM:LOAD:PROG

/

vba	VBA
bas	
frm	
cls	

-
-

:MMEM:MDIR

:MMEM:DEL

:MMEM:COPY

E5071CE5071C

:MMEM:TRAN

E5071CE5071C

E5071C

:MMEM:CAT?

I/O

- I/O
- I/O
- /
-
-
-

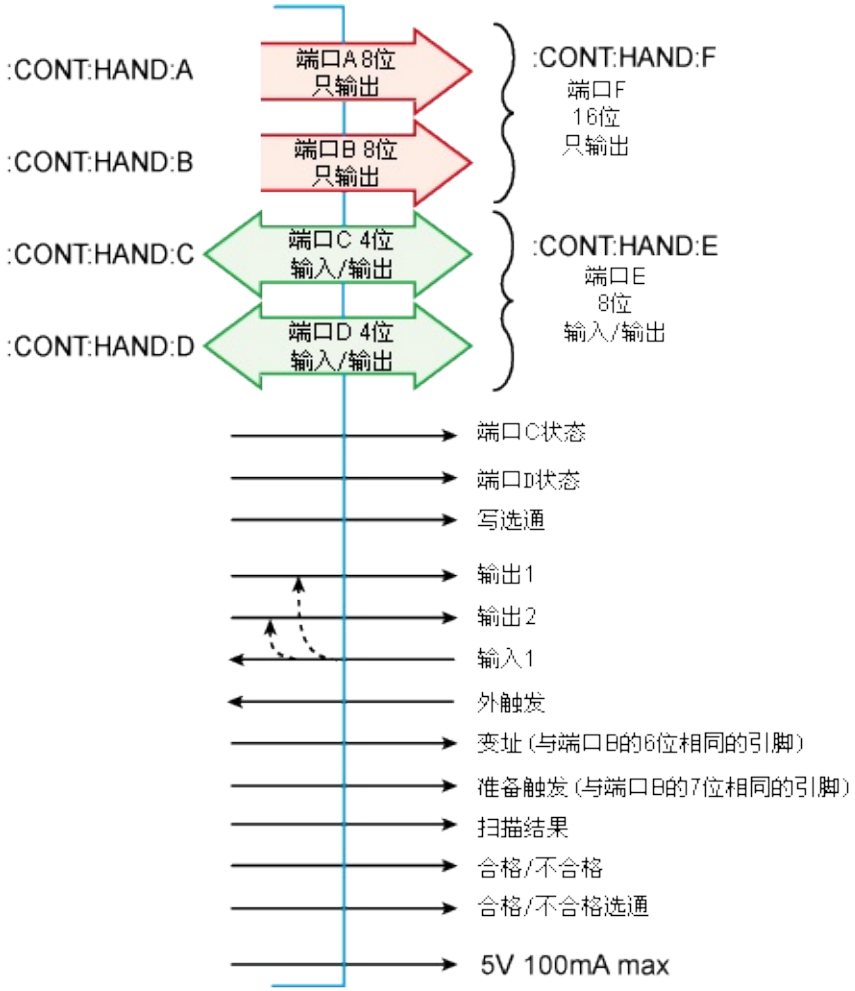
I/O

E5071CI/O/I/OTTL

I/O84168

I/O

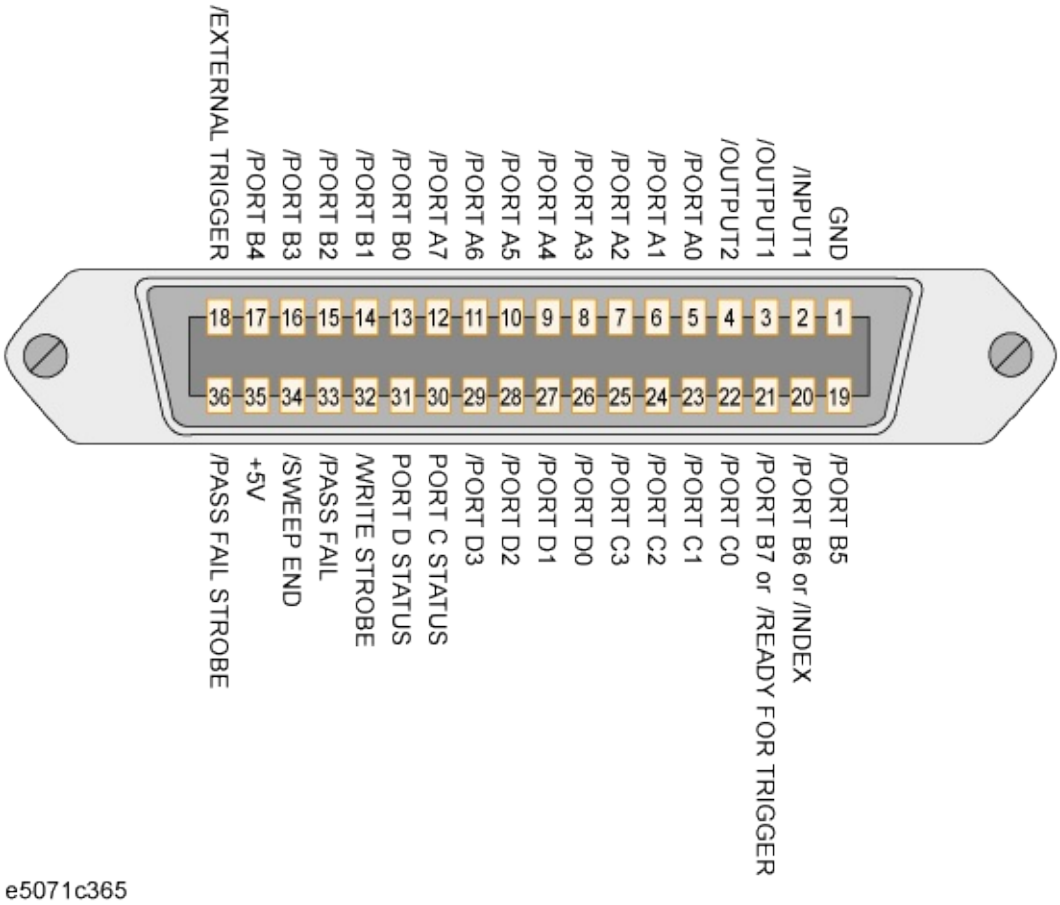
I/O



e5071c364

I/O

I/O



e5071c365

(/)

		/	
1			
2	/INPUT1		/OUTPUT1/OUTPUT2
3	/OUTPUT1		/INPUT1/
4	/OUTPUT2		/INPUT1/

5	/PORT A0		A08
6	/PORT A1		A1
7	/PORT A2		A2
8	/PORT A3		A3
9	/PORT A4		A4
10	/PORT A5		A5
11	/PORT A6		A6
12	/PORT A7		A7
13	/PORT B0		B08
14	/PORT B1		B1
15	/PORT B2		B2
16	/PORT B3		B3
17	/PORT B4		B4
18	/EXTERNAL TRIGGER		“External”
19	/PORT B5		B5
20	/PORT B6		B6
	/INDEX		/INDEX (DUT)
	/PORT B7		B7

21	/READY FOR TRIGGER		
22	/PORT C0	/	C04I/O
23	/PORT C1	/	C1
24	/PORT C2	/	C2
25	/PORT C3	/	C3
26	/PORT D0	/	D04I/O
27	/PORT D1	/	D1
28	/PORT D2	/	D2
29	/PORT D3	/	D3
30	PORT C STATUS		CC
31	PORT D STATUS		DD
32	/WRITE STROBE		
33	/PASS FAIL		FAIL PASS
34	/SWEEP END		

35	+5V		+5V
36	/PASS FAIL STROBE		/PASS FAIL

/

-
-
-
-
-

E5071CI/OI/O

A		A7 A6 A5 A4 A3 A2 A1 A0 8 bits
B		B7 B6 B5 B4 B3 B2 B1 B0 8 bits
C	/	C3 C2 C1 C0 4 bits
D	/	D3 D2 D1 D0 4 bits
E	/	D3 D2 D1 D0 C3 C2 C1 C0 8 bits
F		B7 B6 B5 B4 B3 B2 B1 B0 A7 A6 A5 A4 A3 A2 A1 A0 16 bits

I/OCD/

CDECD

C	:CONT:HAND:C:MODE
D	:CONT:HAND:D:MODE

CDE(0)/(1)

C	:CONT:HAND:C
D	:CONT:HAND:D
E	:CONT:HAND:E

AFCDE0/1

A	:CONT:HAND:A
B	:CONT:HAND:B
C	:CONT:HAND:C
D	:CONT:HAND:D
E	:CONT:HAND:E
F	:CONT:HAND:F

- /INDEX :CONT:HAND:B6 :CONT:HAND:F14
- /READY FOR TRIGGER :CONT:HAND:F7 :CONT:HAND:F15

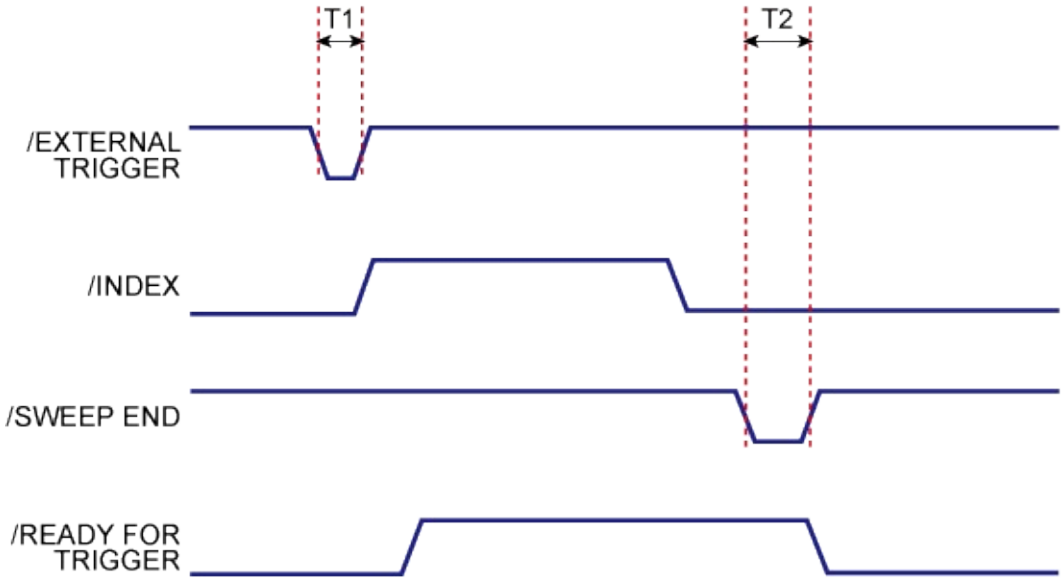
I/O

A	
B	
C	
D	
C	
D	
/1	
/2	
/	
/	

-
- I/O
-
-
- /INPUT11/OUTPUT11/OUTPUT22

I/O/

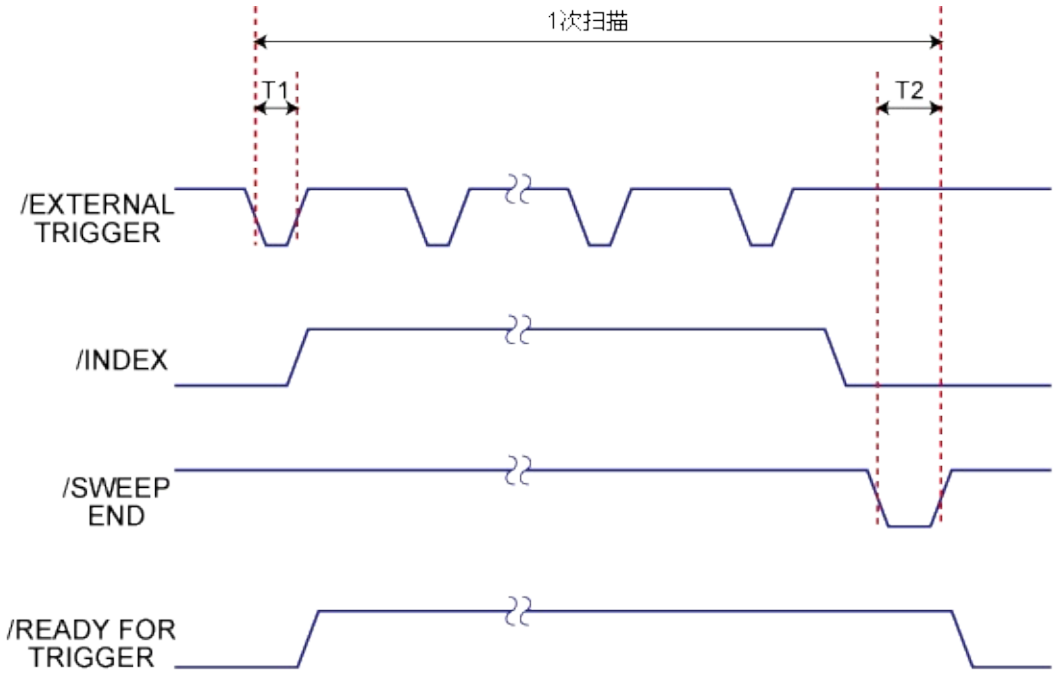
I/O



e5071c368

T1	/EXTERNAL TRIGGER	1 µs	-	-
T2	/SWEEP END	10 µs	12 µs	-

I/O



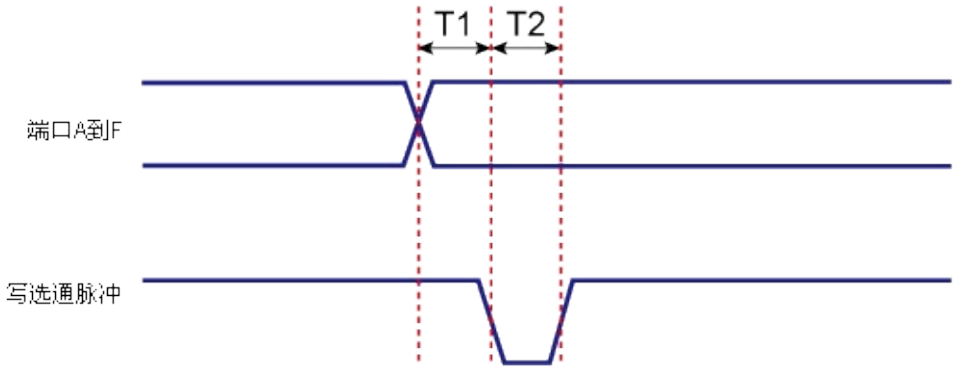
e5071c309

/EXTERNAL TRIGGER/INDEX

/READY FOR TRIGGER

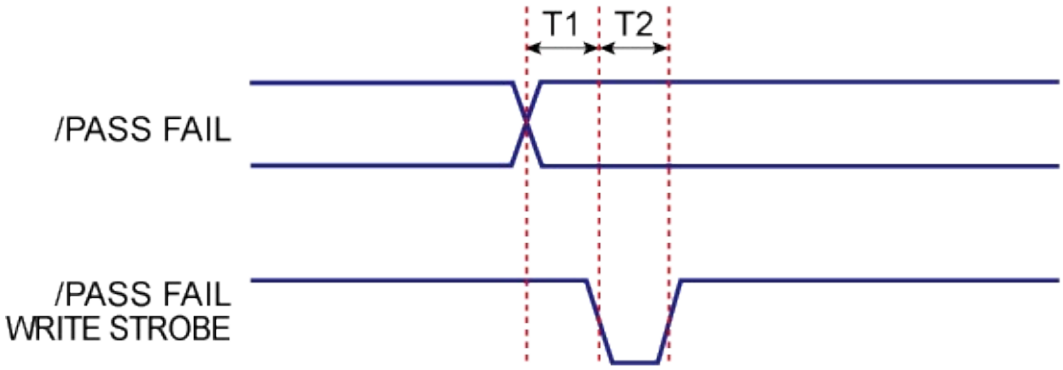
/READY FOR TRIGGER /READY

T1T2 I/O



e5071c366

T1		1 μs
T2		1 μs

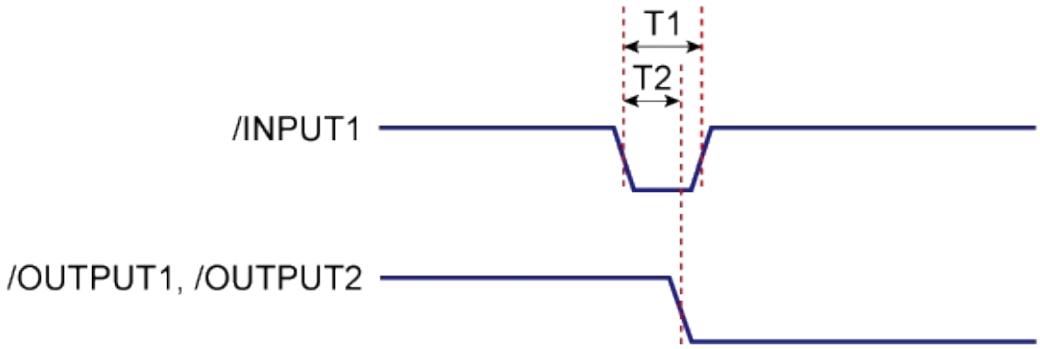


e5071c367

T1	/PASS FAIL	1 μs
T2	/PASS FAIL	1 μs

“”

/INPUT11/OUTPUT1 1 /OUTPUT22



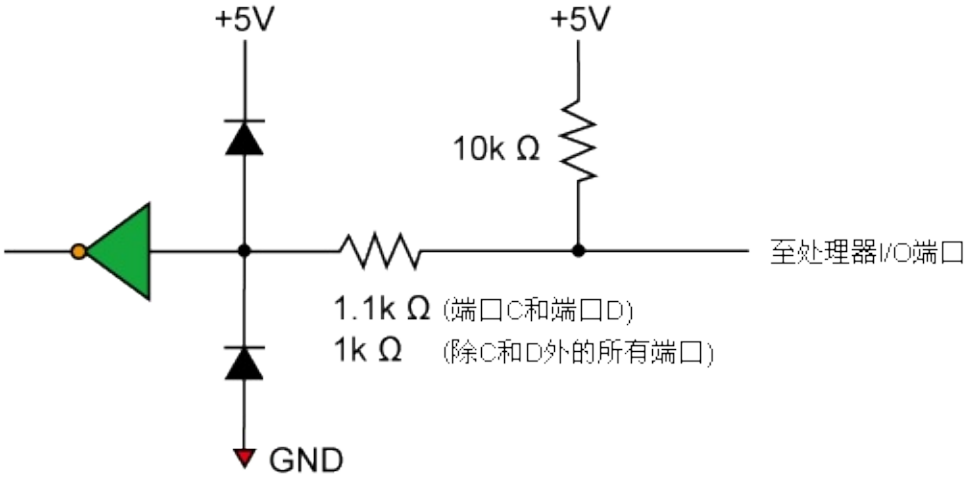
e5071c369

T1	/INPUT1	1 µs	-
T2	/OUTPUT1, /OUTPUT2	0.4 µs	0.6 µs

-
-
- (+5 V)

TTL

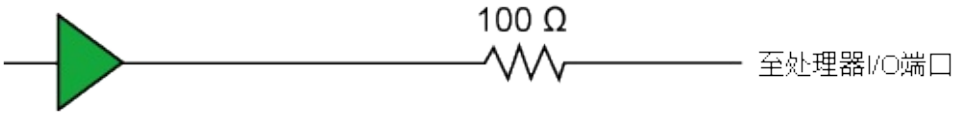
	-0.5 V5.5 V
	2.0 V5.0 V
	0 V0.5 V



e5071c370

TTL

		-10 mA10 mA
		-5 mA
		3 mA
		2.0 V3.3 V (-5 mA0 mA) 3.20 V (-1 mA) 2.75 V (-5 mA)
		0 V0.8 V (0 mA3 mA) 0.25 V (1 mA) 0.55 V (3 mA)



e5071c371

(+5 V)

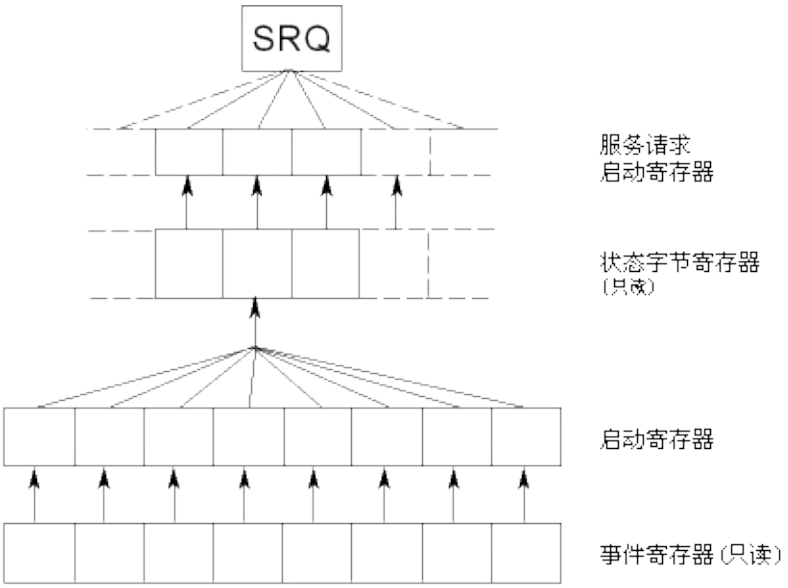
+5 V

	4.5 V5.5 V
	100 mA

-
-
-

-
-
-
-
-

Agilent E5071C



4294ape021

1

“1” “1”“1”

“1”(SRQ)SRQE5071CSRQSRQ “

E5071CE5071C“1”SCPI

“1”"1"

“1”“1”

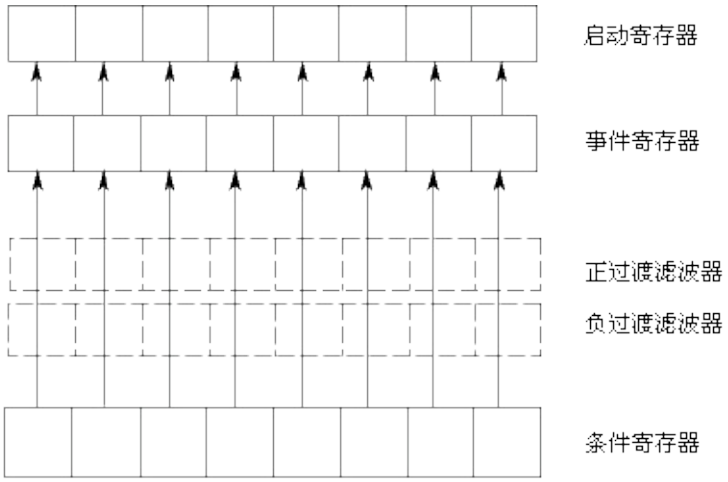
“1”“1”SRQ

*STB?HTBasicSPOLL

*STB?HTBasicSPOLLRQS

*SRE

3103“1”3"1"



4294ape022

E5071C

-
-
-
-
- {1-16}
- {1-16}
-
-
- {1-16}
- {1-16}

-
-
- {1-16}
- {1-16}

- *CLS
- *SRE
- *STB?
- *ESE
- *ESR?
- :STAT:PRES
- :STAT:OPER:ENAB
- :STAT:OPER:COND?
- :STAT:OPER?
- :STAT:OPER:PTR
- :STAT:OPER:NTR
- :STAT:QUES:ENAB
- :STAT:QUES:COND?
- :STAT:QUES?
- :STAT:QUES:PTR
- :STAT:QUES:NTR
- :STAT:QUES:LIM:ENAB
- :STAT:QUES:LIM:COND?
- :STAT:QUES:LIM?
- :STAT:QUES:LIM:PTR
- :STAT:QUES:LIM:NTR
- :STAT:QUES:LIM:ELIM:ENAB

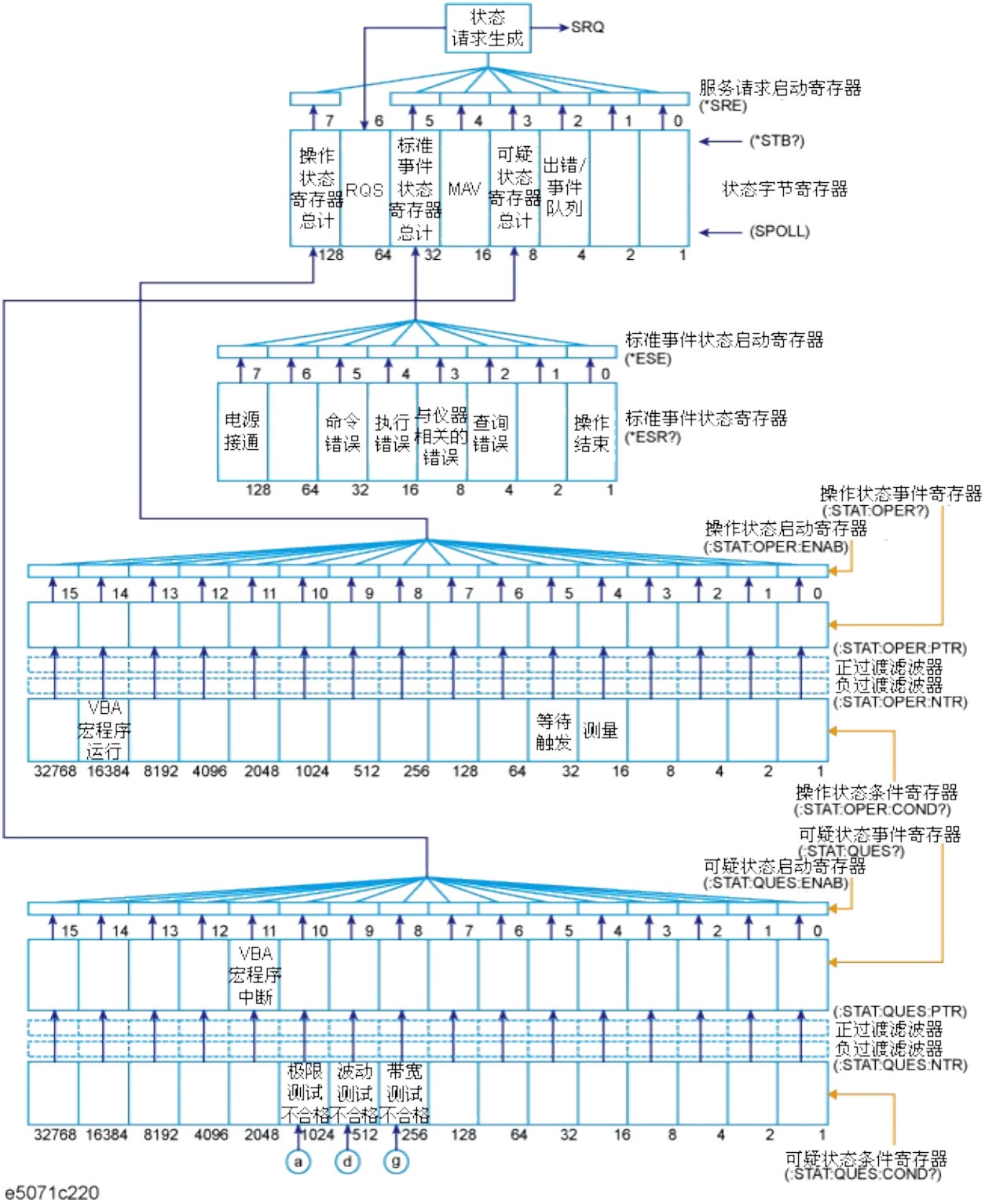
- :STAT:QUES:LIM:ELIM:COND?
- :STAT:QUES:LIM:ELIM?
- :STAT:QUES:LIM:ELIM:PTR
- :STAT:QUES:LIM:ELIM:NTR
- :STAT:QUES:LIM:CHAN{1-16}:ENAB
- :STAT:QUES:LIM:CHAN{1-16}:COND?
- :STAT:QUES:LIM:CHAN{1-16}?
- :STAT:QUES:LIM:CHAN{1-16}:PTR
- :STAT:QUES:LIM:CHAN{1-16}:NTR
- :STAT:QUES:LIM:CHAN{1-16}:ECH:ENAB
- :STAT:QUES:LIM:CHAN{1-16}:ECH:COND?
- :STAT:QUES:LIM:CHAN{1-16}:ECH?
- :STAT:QUES:LIM:CHAN{1-16}:ECH:PTR
- :STAT:QUES:LIM:CHAN{1-16}:ECH:NTR
- :STAT:QUES:BLIM:ENAB
- :STAT:QUES:BLIM:COND?
- :STAT:QUES:BLIM?
- :STAT:QUES:BLIM:PTR
- :STAT:QUES:BLIM:NTR
- :STAT:QUES:BLIM:ELIM:ENAB
- :STAT:QUES:BLIM:ELIM:COND?
- :STAT:QUES:BLIM:ELIM?
- :STAT:QUES:BLIM:ELIM:PTR
- :STAT:QUES:BLIM:ELIM:NTR
- :STAT:QUES:BLIM:CHAN{1-16}:ENAB

- :STAT:QUES:BLIM:CHAN{1-16}:COND?
- :STAT:QUES:BLIM:CHAN{1-16}?
- :STAT:QUES:BLIM:CHAN{1-16}:PTR
- :STAT:QUES:BLIM:CHAN{1-16}:NTR
- :STAT:QUES:BLIM:CHAN{1-16}:ECH:ENAB
- :STAT:QUES:BLIM:CHAN{1-16}:ECH:COND?
- :STAT:QUES:BLIM:CHAN{1-16}:ECH?
- :STAT:QUES:BLIM:CHAN{1-16}:ECH:PTR
- :STAT:QUES:BLIM:CHAN{1-16}:ECH:NTR
- :STAT:QUES:RLIM:ENAB
- :STAT:QUES:RLIM:COND?
- :STAT:QUES:RLIM?
- :STAT:QUES:RLIM:PTR
- :STAT:QUES:RLIM:NTR
- :STAT:QUES:RLIM:ELIM:ENAB
- :STAT:QUES:RLIM:ELIM:COND?
- :STAT:QUES:RLIM:ELIM?
- :STAT:QUES:RLIM:ELIM:PTR
- :STAT:QUES:RLIM:ELIM:NTR
- :STAT:QUES:RLIM:CHAN{1-16}:ENAB
- :STAT:QUES:RLIM:CHAN{1-16}:COND?
- :STAT:QUES:RLIM:CHAN{1-16}?
- :STAT:QUES:RLIM:CHAN{1-16}:PTR
- :STAT:QUES:RLIM:CHAN{1-16}:NTR
- :STAT:QUES:RLIM:CHAN{1-16}:ECH:ENAB

- :STAT:QUES:RLIM:CHAN{1-16}:ECH:COND?
- :STAT:QUES:RLIM:CHAN{1-16}:ECH?
- :STAT:QUES:RLIM:CHAN{1-16}:ECH:PTR
- :STAT:QUES:RLIM:CHAN{1-16}:ECH:NTR

- .
- _____
- (1) _____
- (2) _____
- _____
- 1 _____
- 2 _____
- _____
- (1) _____
- (2) _____

E5171C36/16


状态
请求生成
SRQ
服务请求启动寄存器
(*SRE)
7 6 5 4 3 2 1 0
(*STB?)
操作状态寄存器总计
RQS
标准事件状态寄存器总计
MAV
可疑状态寄存器总计
出错/事件队列
状态字节寄存器
(SPOLL)
128 64 32 16 8 4 2 1
标准事件状态启动寄存器
(*ESE)
7 6 5 4 3 2 1 0
电源接通
命令错误
执行错误
与仪器相关的错误
查询错误
操作结束
标准事件状态寄存器
(*ESR?)
128 64 32 16 8 4 2 1
操作状态事件寄存器
(:STAT:OPER?)
操作状态启动寄存器
(:STAT:OPER:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:OPER:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:OPER:NTR)
VBA
宏程序
运行
等待
触发
测量
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
操作状态条件寄存器
(:STAT:OPER:COND?)
可疑状态事件寄存器
(:STAT:QUES?)
可疑状态启动寄存器
(:STAT:QUES:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
VBA
宏程序
中断
(:STAT:QUES:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:NTR)
极限
测试
不合格
波动
测试
不合格
带宽
测试
不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
a
d
g
可疑状态条件寄存器
(:STAT:QUES:COND?)

e5071c220

0, 1		0
2	/	/“1”“0”
3		“1”“1”
4	MAV	1 “0”
5		“1”“1”
6	RQS	“1”“1”“0”
7		“1”“1”

*CLS

0		*OPC?“1”
1		0
2		1.E5071C“1” 2.E5071C“1”
3		“1”
4		1.SCPIE5071C“1” 2.SCPIE5071C“1”
		1.IEEE 488.2E5071CIEEE 488.2

5		“1”E5071C 2.“1”E5071CE5071CIEEE 3.GET“1”
6		0
7		E5071C“1”

*CLS

0 - 3		0
4		“1”
5		“1”
6 - 13		0
14	VBA	VBA“1”
15		0

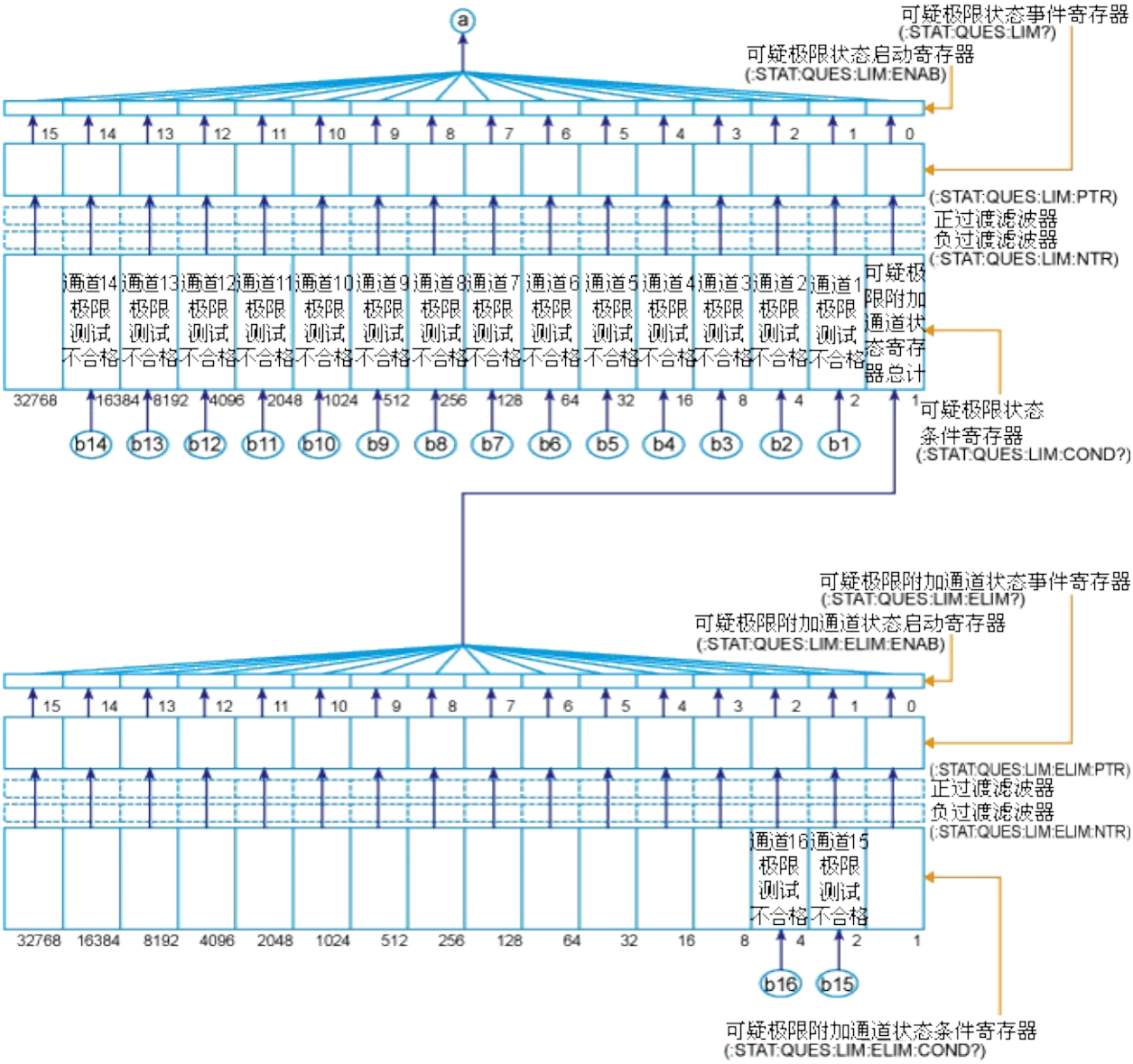
*CLS

0 - 7		0

8		“1”“1”
9		“1”“1”
10		“1”“1”
11 - 15		0

0 - 7		0
8		“1”
9		“1”
10		“1”
11	VBA	VBA“1” VBA“End” :PROG:STAT STOP **CTRL** + **Break** **Marcro BreakMacro Setup** > **Stop**
12 - 15		0

*CLS co



e5071c221

0	15, 16	“1”“1”

1	11	1“1”“1”
2	22	2“1”“1”
3	33	3“1”“1”
4	44	4“1”“1”
5	55	5“1”“1”
6	66	6“1”“1”
7	77	7“1”“1”
8	88	8“1”“1”
9	99	9“1”“1”
10	1010	10“1”“1”
11	1111	11“1”“1”
12	1212	12“1”“1”
13	1313	13“1”“1”
14	1414	14“1”“1”
15		0

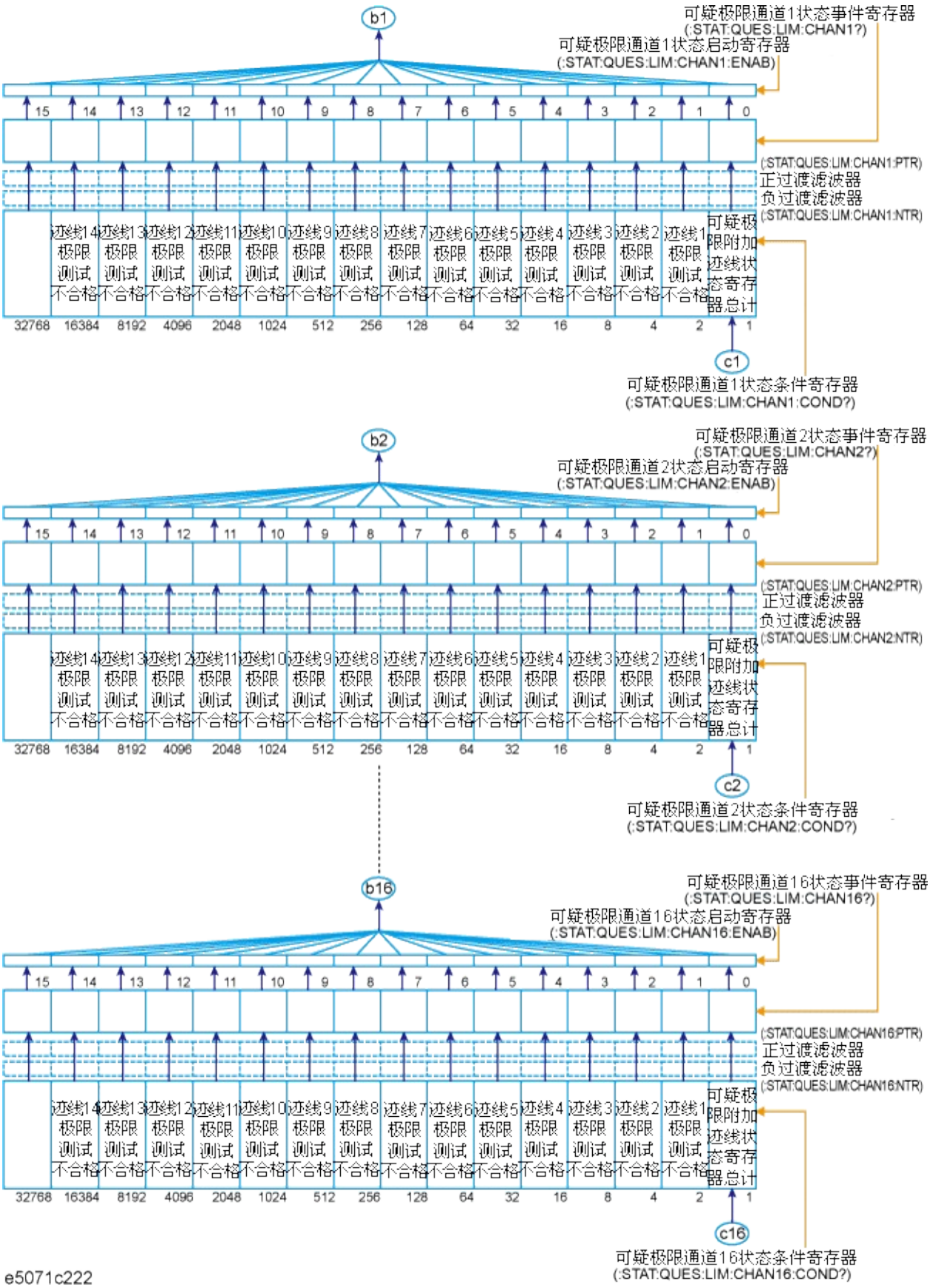
*CLS

0		0

1	1515	15“1”“1”
2	1616	15“1”“1”
3 - 15		0

*CLS

(1)


b1
可疑极限通道1状态事件寄存器
(:STAT:QUES:LIM:CHAN1?)
可疑极限通道1状态启动寄存器
(:STAT:QUES:LIM:CHAN1:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:LIM:CHAN1:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:LIM:CHAN1:NTR)
迹线14极限测试不合格
迹线13极限测试不合格
迹线12极限测试不合格
迹线11极限测试不合格
迹线10极限测试不合格
迹线9极限测试不合格
迹线8极限测试不合格
迹线7极限测试不合格
迹线6极限测试不合格
迹线5极限测试不合格
迹线4极限测试不合格
迹线3极限测试不合格
迹线2极限测试不合格
迹线1极限测试不合格
可疑极限附加迹线状态寄存器总计
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
c1
可疑极限通道1状态条件寄存器
(:STAT:QUES:LIM:CHAN1:COND?)
b2
可疑极限通道2状态事件寄存器
(:STAT:QUES:LIM:CHAN2?)
可疑极限通道2状态启动寄存器
(:STAT:QUES:LIM:CHAN2:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:LIM:CHAN2:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:LIM:CHAN2:NTR)
迹线14极限测试不合格
迹线13极限测试不合格
迹线12极限测试不合格
迹线11极限测试不合格
迹线10极限测试不合格
迹线9极限测试不合格
迹线8极限测试不合格
迹线7极限测试不合格
迹线6极限测试不合格
迹线5极限测试不合格
迹线4极限测试不合格
迹线3极限测试不合格
迹线2极限测试不合格
迹线1极限测试不合格
可疑极限附加迹线状态寄存器总计
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
c2
可疑极限通道2状态条件寄存器
(:STAT:QUES:LIM:CHAN2:COND?)
b16
可疑极限通道16状态事件寄存器
(:STAT:QUES:LIM:CHAN16?)
可疑极限通道16状态启动寄存器
(:STAT:QUES:LIM:CHAN16:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:LIM:CHAN16:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:LIM:CHAN16:NTR)
迹线14极限测试不合格
迹线13极限测试不合格
迹线12极限测试不合格
迹线11极限测试不合格
迹线10极限测试不合格
迹线9极限测试不合格
迹线8极限测试不合格
迹线7极限测试不合格
迹线6极限测试不合格
迹线5极限测试不合格
迹线4极限测试不合格
迹线3极限测试不合格
迹线2极限测试不合格
迹线1极限测试不合格
可疑极限附加迹线状态寄存器总计
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
c16
可疑极限通道16状态条件寄存器
(:STAT:QUES:LIM:CHAN16:COND?)

e5071c222

{1-16}

0	15, 16{1-16}	{1-16}“1” “1”
1	1	“0” “1”“fail”1
2	2	“0” “1”“fail”2
3	3	“0” “1”“fail”3
4	4	“0” “1”“fail”4
5	5	“0” “1”“fail”5
6	6	“0” “1”“fail”6
7	7	“0” “1”“fail”7
8	8	“0” “1”“fail”8
9	9	“0” “1”“fail”9
10	10	“0” “1”“fail”10
		“0”

11	11	“1”“fail”11
12	12	“0” “1”“fail”12
13	13	“0” “1”“fail”13
14	14	“0” “1”“fail”14
15		0

*CLS{1-16}

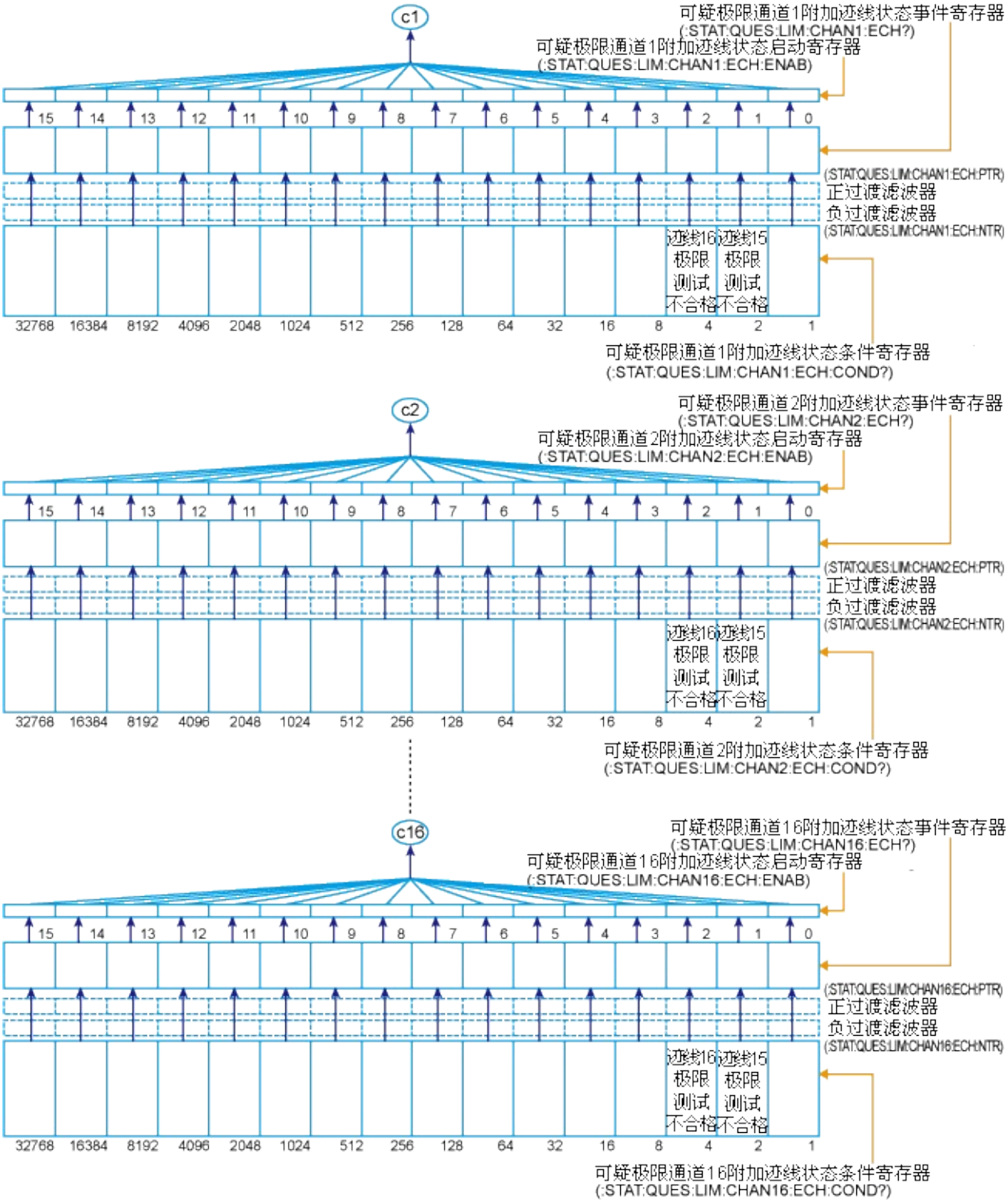
{1-16}

0		0
1	15	“0” “1”“fail”15
2	16	“0” “1”“fail”16
3 - 15		0

*CLS{1-16}

E5071C3616

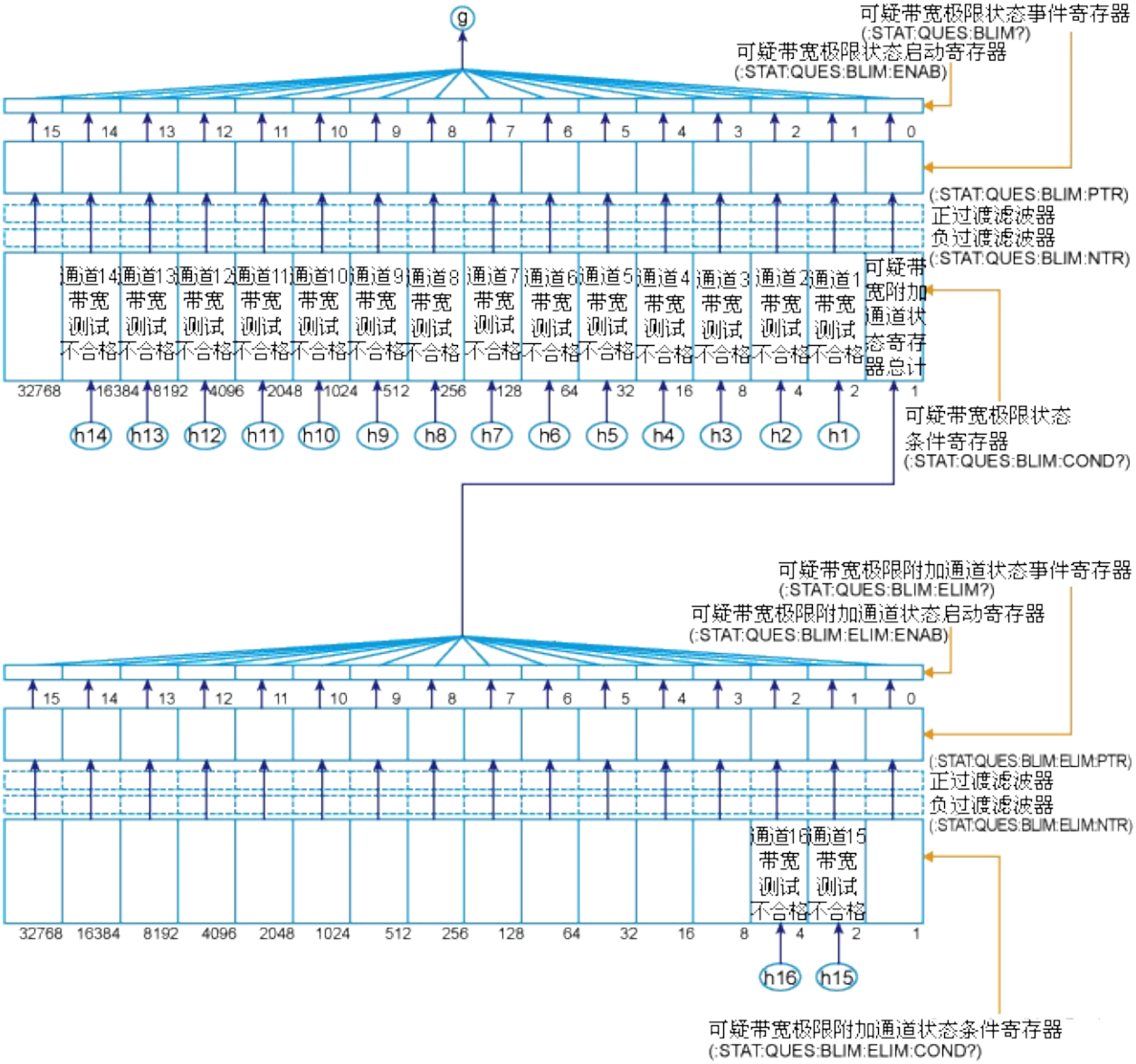
(2)


c1
可疑极限通道1附加迹线状态事件寄存器
(:STAT:QUES:LIM:CHAN1:ECH?)
可疑极限通道1附加迹线状态启动寄存器
(:STAT:QUES:LIM:CHAN1:ECH:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:LIM:CHAN1:ECH:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:LIM:CHAN1:ECH:NTR)
迹线16极限测试不合格
迹线15极限测试不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
可疑极限通道1附加迹线状态条件寄存器
(:STAT:QUES:LIM:CHAN1:ECH:COND?)
c2
可疑极限通道2附加迹线状态事件寄存器
(:STAT:QUES:LIM:CHAN2:ECH?)
可疑极限通道2附加迹线状态启动寄存器
(:STAT:QUES:LIM:CHAN2:ECH:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:LIM:CHAN2:ECH:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:LIM:CHAN2:ECH:NTR)
迹线16极限测试不合格
迹线15极限测试不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
可疑极限通道2附加迹线状态条件寄存器
(:STAT:QUES:LIM:CHAN2:ECH:COND?)
c16
可疑极限通道16附加迹线状态事件寄存器
(:STAT:QUES:LIM:CHAN16:ECH?)
可疑极限通道16附加迹线状态启动寄存器
(:STAT:QUES:LIM:CHAN16:ECH:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:LIM:CHAN16:ECH:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:LIM:CHAN16:ECH:NTR)
迹线16极限测试不合格
迹线15极限测试不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
可疑极限通道16附加迹线状态条件寄存器
(:STAT:QUES:LIM:CHAN16:ECH:COND?)

e5071c223

0		0
1	1515	15“1”“1”
2	1616	16“1”“1”
3 - 15		0

*CLS



e5071c226

{1-16}

0	15, 16{1-16}	{1-16}“1” “1”

1	1	“0” “1”“fail”1
2	2	“0” “1”“fail”2
3	3	“0” “1”“fail”3
4	4	“0” “1”“fail”4
5	5	“0” “1”“fail”5
6	6	“0” “1”“fail”6
7	7	“0” “1”“fail”7
8	8	“0” “1”“fail”8
9	9	“0” “1”“fail”9
10	10	“0” “1”“fail”10
11	11	“0” “1”“fail”11
12	12	“0” “1”“fail”12
13	13	“0” “1”“fail”13

14	14	“0” “1”“fail”14
15		0

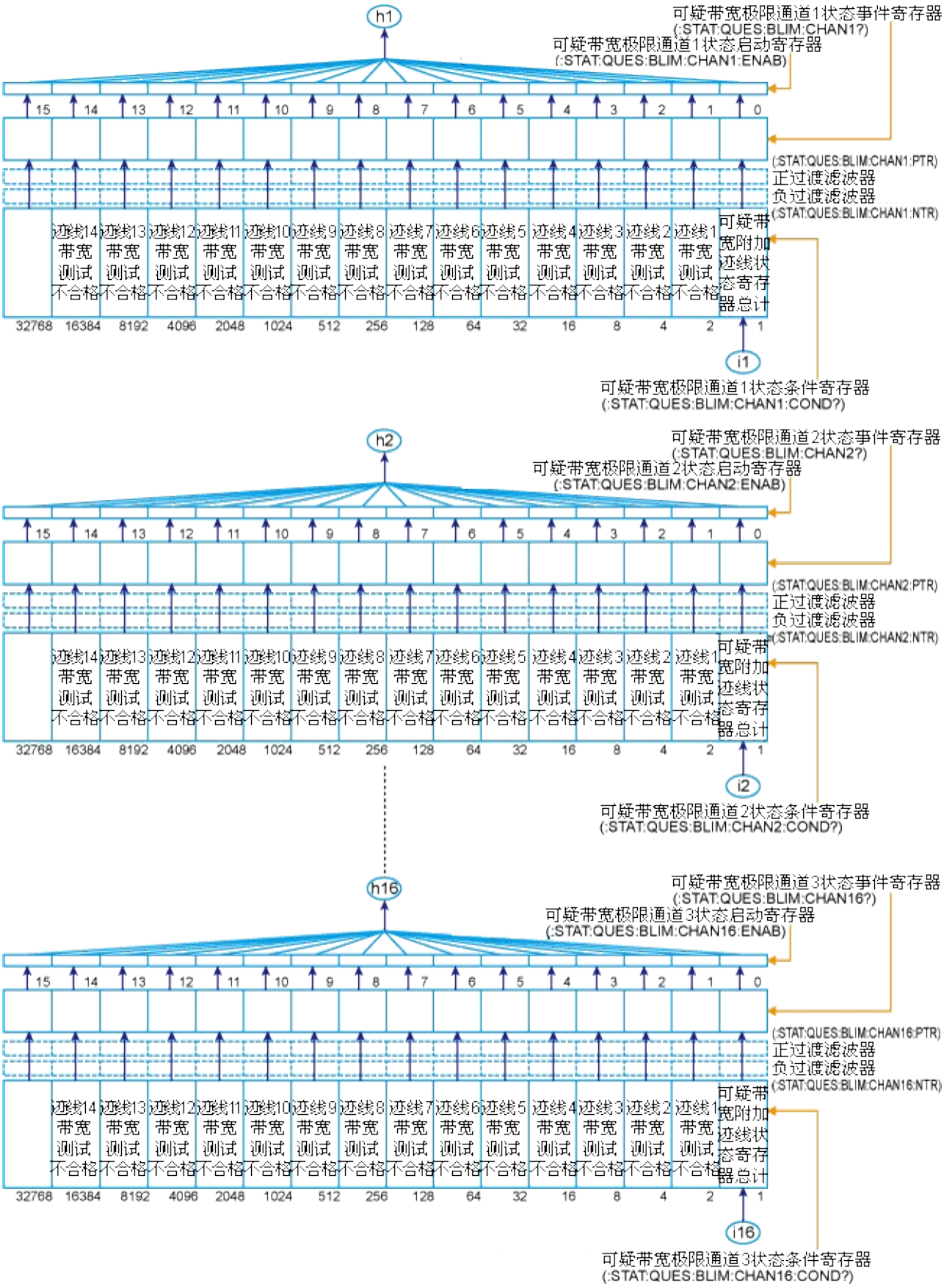
*CLS{1-16}

{1-16}

0		0
1	15	“0” “1”“fail”15
2	16	“0” “1”“fail”16
3 - 15		0

*CLS{1-16}

1

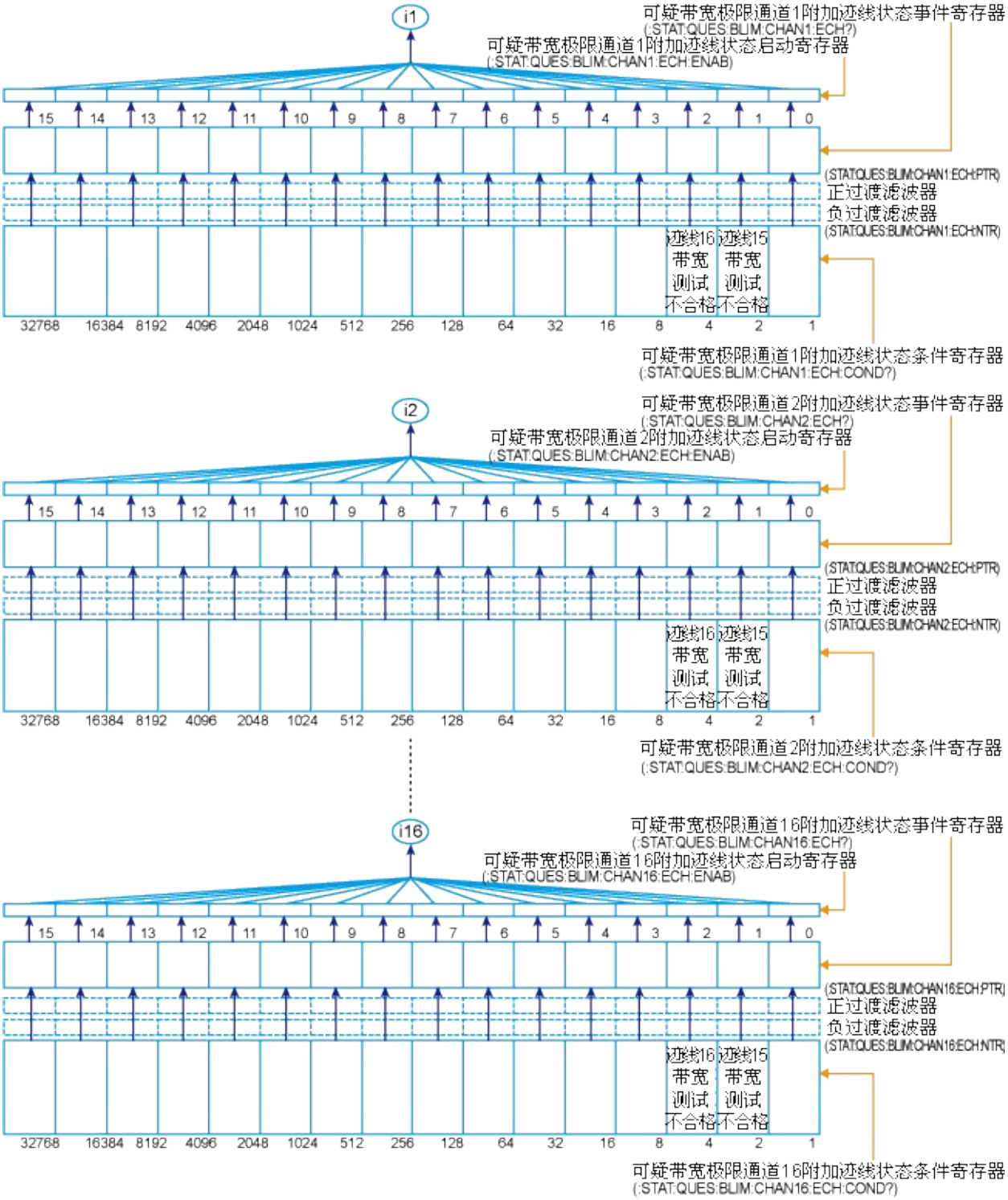

h1
可疑带宽极限通道1状态事件寄存器
(:STAT:QUES:BLIM:CHAN1?)
可疑带宽极限通道1状态启动寄存器
(:STAT:QUES:BLIM:CHAN1:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:BLIM:CHAN1:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:BLIM:CHAN1:NTR)
迹线14带宽测试不合格
迹线13带宽测试不合格
迹线12带宽测试不合格
迹线11带宽测试不合格
迹线10带宽测试不合格
迹线9带宽测试不合格
迹线8带宽测试不合格
迹线7带宽测试不合格
迹线6带宽测试不合格
迹线5带宽测试不合格
迹线4带宽测试不合格
迹线3带宽测试不合格
迹线2带宽测试不合格
迹线1带宽测试不合格
可疑带宽附加迹线状态寄存器总计
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
i1
可疑带宽极限通道1状态条件寄存器
(:STAT:QUES:BLIM:CHAN1:COND?)
h2
可疑带宽极限通道2状态事件寄存器
(:STAT:QUES:BLIM:CHAN2?)
可疑带宽极限通道2状态启动寄存器
(:STAT:QUES:BLIM:CHAN2:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:BLIM:CHAN2:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:BLIM:CHAN2:NTR)
迹线14带宽测试不合格
迹线13带宽测试不合格
迹线12带宽测试不合格
迹线11带宽测试不合格
迹线10带宽测试不合格
迹线9带宽测试不合格
迹线8带宽测试不合格
迹线7带宽测试不合格
迹线6带宽测试不合格
迹线5带宽测试不合格
迹线4带宽测试不合格
迹线3带宽测试不合格
迹线2带宽测试不合格
迹线1带宽测试不合格
可疑带宽附加迹线状态寄存器总计
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
i2
可疑带宽极限通道2状态条件寄存器
(:STAT:QUES:BLIM:CHAN2:COND?)
h16
可疑带宽极限通道3状态事件寄存器
(:STAT:QUES:BLIM:CHAN16?)
可疑带宽极限通道3状态启动寄存器
(:STAT:QUES:BLIM:CHAN16:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:BLIM:CHAN16:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:BLIM:CHAN16:NTR)
迹线14带宽测试不合格
迹线13带宽测试不合格
迹线12带宽测试不合格
迹线11带宽测试不合格
迹线10带宽测试不合格
迹线9带宽测试不合格
迹线8带宽测试不合格
迹线7带宽测试不合格
迹线6带宽测试不合格
迹线5带宽测试不合格
迹线4带宽测试不合格
迹线3带宽测试不合格
迹线2带宽测试不合格
迹线1带宽测试不合格
可疑带宽附加迹线状态寄存器总计
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
i16
可疑带宽极限通道3状态条件寄存器
(:STAT:QUES:BLIM:CHAN16:COND?)

e5071c227

0	15, 16	“1”“1”
1	11	1“1”“1”
2	22	2“1”“1”
3	33	3“1”“1”
4	44	4“1”“1”
5	55	5“1”“1”
6	66	6“1”“1”
7	77	7“1”“1”
8	88	8“1”“1”
9	99	9“1”“1”
10	1010	10“1”“1”
11	1111	11“1”“1”
12	1212	12“1”“1”
13	1313	13“1”“1”
14	1414	14“1”“1”
15		0

*CLS

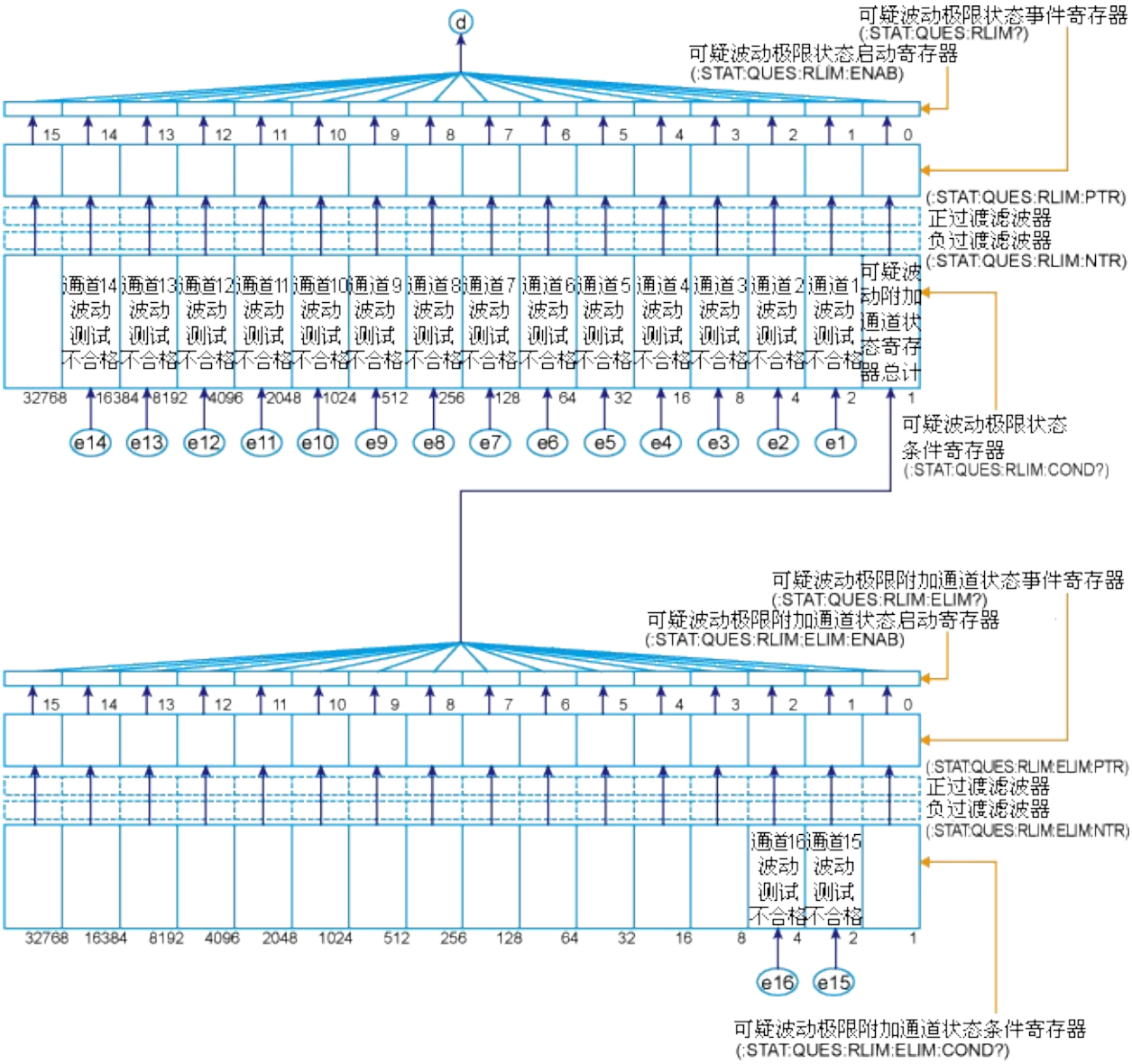
2


i1
可疑带宽极限通道1附加迹线状态事件寄存器
(:STAT:QUES:BLIM:CHAN1:ECH?)
可疑带宽极限通道1附加迹线状态启动寄存器
(:STAT:QUES:BLIM:CHAN1:ECH:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:BLIM:CHAN1:ECH:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:BLIM:CHAN1:ECH:NTR)
迹线16 带宽 测试 不合格
迹线15 带宽 测试 不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
可疑带宽极限通道1附加迹线状态条件寄存器
(:STAT:QUES:BLIM:CHAN1:ECH:COND?)
i2
可疑带宽极限通道2附加迹线状态事件寄存器
(:STAT:QUES:BLIM:CHAN2:ECH?)
可疑带宽极限通道2附加迹线状态启动寄存器
(:STAT:QUES:BLIM:CHAN2:ECH:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:BLIM:CHAN2:ECH:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:BLIM:CHAN2:ECH:NTR)
迹线16 带宽 测试 不合格
迹线15 带宽 测试 不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
可疑带宽极限通道2附加迹线状态条件寄存器
(:STAT:QUES:BLIM:CHAN2:ECH:COND?)
i16
可疑带宽极限通道16附加迹线状态事件寄存器
(:STAT:QUES:BLIM:CHAN16:ECH?)
可疑带宽极限通道16附加迹线状态启动寄存器
(:STAT:QUES:BLIM:CHAN16:ECH:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:BLIM:CHAN16:ECH:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:BLIM:CHAN16:ECH:NTR)
迹线16 带宽 测试 不合格
迹线15 带宽 测试 不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
可疑带宽极限通道16附加迹线状态条件寄存器
(:STAT:QUES:BLIM:CHAN16:ECH:COND?)

e5071c228

0		0
1	1515	15“1”“1”
2	1616	16“1”“1”
3 - 15		0

*CLS



e5071c224

0	15, 16	“1”“1”

1	11	1“1”“1”
2	22	2“1”“1”
3	33	3“1”“1”
4	44	4“1”“1”
5	55	5“1”“1”
6	66	6“1”“1”
7	77	7“1”“1”
8	88	8“1”“1”
9	99	9“1”“1”
10	1010	10“1”“1”
11	1111	11“1”“1”
12	1212	12“1”“1”
13	1313	13“1”“1”
14	1414	14“1”“1”
15		0

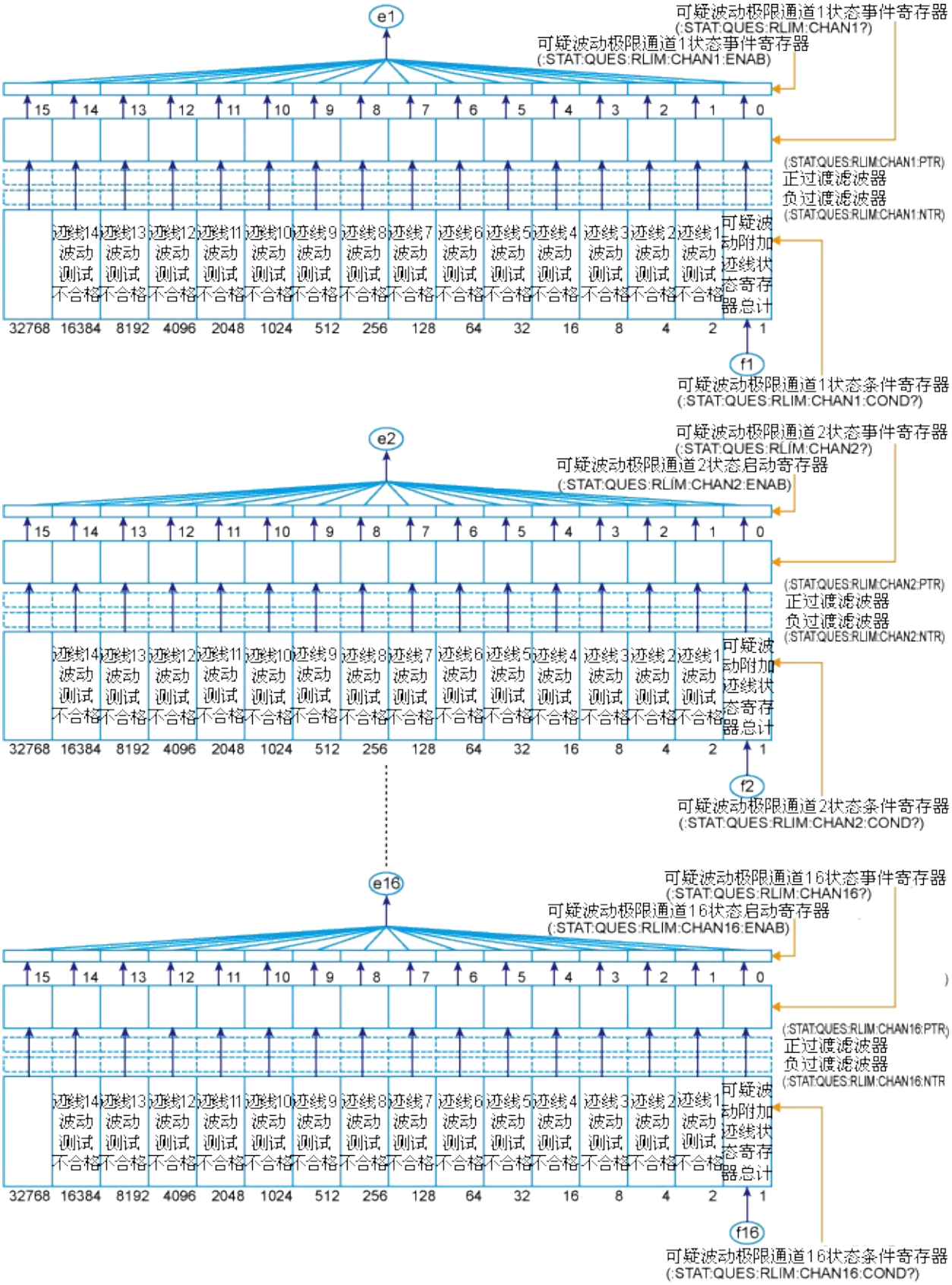
*CLS

0		0

1	1515	15“1”“1”
2	1616	16“1”“1”
3-15		0

*CLS

(1)


e1
可疑波动极限通道1状态事件寄存器
(:STAT:QUES:RLIM:CHAN1?)
可疑波动极限通道1状态启动寄存器
(:STAT:QUES:RLIM:CHAN1:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:RLIM:CHAN1:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:RLIM:CHAN1:NTR)
迹线14波动测试不合格
迹线13波动测试不合格
迹线12波动测试不合格
迹线11波动测试不合格
迹线10波动测试不合格
迹线9波动测试不合格
迹线8波动测试不合格
迹线7波动测试不合格
迹线6波动测试不合格
迹线5波动测试不合格
迹线4波动测试不合格
迹线3波动测试不合格
迹线2波动测试不合格
迹线1波动测试不合格
可疑波动附加迹线状态寄存器总计
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
f1
可疑波动极限通道1状态条件寄存器
(:STAT:QUES:RLIM:CHAN1:COND?)
e2
可疑波动极限通道2状态事件寄存器
(:STAT:QUES:RLIM:CHAN2?)
可疑波动极限通道2状态启动寄存器
(:STAT:QUES:RLIM:CHAN2:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:RLIM:CHAN2:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:RLIM:CHAN2:NTR)
迹线14波动测试不合格
迹线13波动测试不合格
迹线12波动测试不合格
迹线11波动测试不合格
迹线10波动测试不合格
迹线9波动测试不合格
迹线8波动测试不合格
迹线7波动测试不合格
迹线6波动测试不合格
迹线5波动测试不合格
迹线4波动测试不合格
迹线3波动测试不合格
迹线2波动测试不合格
迹线1波动测试不合格
可疑波动附加迹线状态寄存器总计
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
f2
可疑波动极限通道2状态条件寄存器
(:STAT:QUES:RLIM:CHAN2:COND?)
e16
可疑波动极限通道16状态事件寄存器
(:STAT:QUES:RLIM:CHAN16?)
可疑波动极限通道16状态启动寄存器
(:STAT:QUES:RLIM:CHAN16:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:RLIM:CHAN16:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:RLIM:CHAN16:NTR
迹线14波动测试不合格
迹线13波动测试不合格
迹线12波动测试不合格
迹线11波动测试不合格
迹线10波动测试不合格
迹线9波动测试不合格
迹线8波动测试不合格
迹线7波动测试不合格
迹线6波动测试不合格
迹线5波动测试不合格
迹线4波动测试不合格
迹线3波动测试不合格
迹线2波动测试不合格
迹线1波动测试不合格
可疑波动附加迹线状态寄存器总计
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
f16
可疑波动极限通道16状态条件寄存器
(:STAT:QUES:RLIM:CHAN16:COND?)

e5071c324

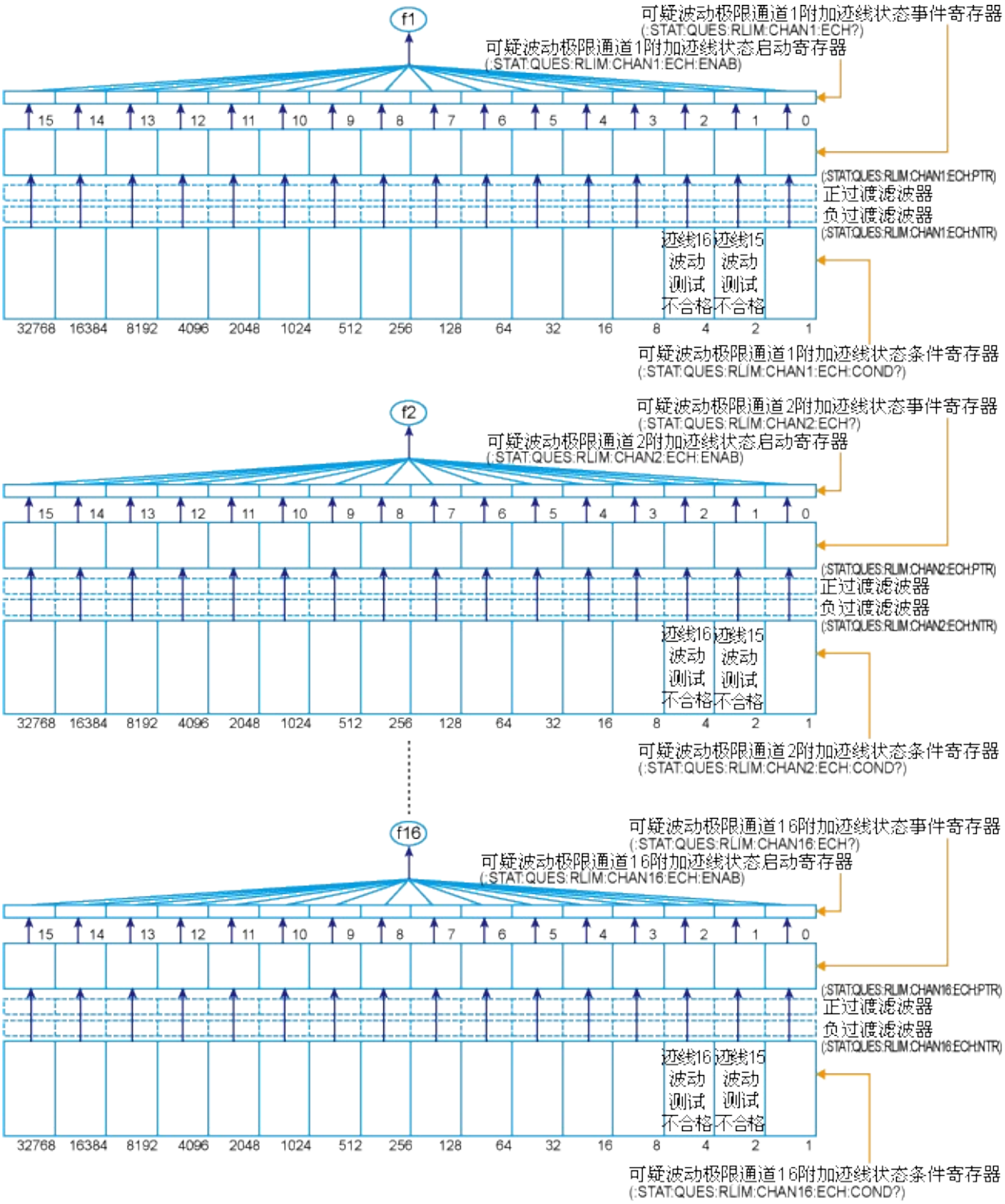
{1-16}

0	15, 16{1-16}	{1-16}“1” “1”
1	1	“0” “1”“fail”1
2	2	“0” “1”“fail”2
3	3	“0” “1”“fail”3
4	4	“0” “1”“fail”4
5	5	“0” “1”“fail”5
6	6	“0” “1”“fail”6
7	7	“0” “1”“fail”7
8	8	“0” “1”“fail”8
9	9	“0” “1”“fail”9
10	10	“0” “1”“fail”10

11	11	“0” “1”“fail”11
12	12	“0” “1”“fail”12
13	13	“0” “1”“fail”13
14	14	“0” “1”“fail”14
15		0

*CLS{1-16}

(2)


f1
可疑波动极限通道1附加迹线状态事件寄存器
(:STAT:QUES:RLIM:CHAN1:ECH?)
可疑波动极限通道1附加迹线状态启动寄存器
(:STAT:QUES:RLIM:CHAN1:ECH:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:RLIM:CHAN1:ECH:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:RLIM:CHAN1:ECH:NTR)
迹线16波动测试不合格
迹线15波动测试不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
可疑波动极限通道1附加迹线状态条件寄存器
(:STAT:QUES:RLIM:CHAN1:ECH:COND?)
f2
可疑波动极限通道2附加迹线状态事件寄存器
(:STAT:QUES:RLIM:CHAN2:ECH?)
可疑波动极限通道2附加迹线状态启动寄存器
(:STAT:QUES:RLIM:CHAN2:ECH:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:RLIM:CHAN2:ECH:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:RLIM:CHAN2:ECH:NTR)
迹线16波动测试不合格
迹线15波动测试不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
可疑波动极限通道2附加迹线状态条件寄存器
(:STAT:QUES:RLIM:CHAN2:ECH:COND?)
f16
可疑波动极限通道16附加迹线状态事件寄存器
(:STAT:QUES:RLIM:CHAN16:ECH?)
可疑波动极限通道16附加迹线状态启动寄存器
(:STAT:QUES:RLIM:CHAN16:ECH:ENAB)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
(:STAT:QUES:RLIM:CHAN16:ECH:PTR)
正过渡滤波器
负过渡滤波器
(:STAT:QUES:RLIM:CHAN16:ECH:NTR)
迹线16波动测试不合格
迹线15波动测试不合格
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
可疑波动极限通道16附加迹线状态条件寄存器
(:STAT:QUES:RLIM:CHAN16:ECH:COND?)

e5071c225

{1-16}

0		0
1	15	"0" "1""fail”15
2	16	"0" "1""fail”16
3-15		0

*CLS{1-16}

- [illegible]
- [illegible]
- [illegible]

:SYST:KLOC:KBD	
:SYST:KLOC:MOUS	

SCPI

E5071CLCDSCPI

LCDSCPI

LCD

:DISP:ENAB

LCDSCPI

1. SCPI
2. LCD
3.
4.
5. LCD:DISP:UPD
6. 3

LCD

-
-
-

E5071C

SRQ

SRQ

*SRE

*ESE

1. E5071C1SRQ
2. SRQ

SRQ

:SYST:ERR?

1. 0“No
2. SRQ

Error Detection(SRQ) .

SCPIVBA VBA

-
-
-
-
- /
- OPC?
- SQR
- SRQ
- ASCII
-
- Ascii
-
-
-
-
-
-
-
-
- SICL-LAN
- Telnet
- LCD

-
- E5091A

http://www.agilent.com/find/ena_support/

-
- Excel VBA
- HT Basic

sample.sta

		12/321/3
		4
		4

1	1		S11
			Smith(Lin)
			2
	2		S21
			9
			2
			10 dBm
	3		S12
			9
			2
			10 dBm
	4		S22
			Smith(Lin)
			2
			1.9 GHz

			500 MHz
			101
			70 kHz
			0 dBm
			4
2	1		S21
			9
			2
			10 dBm
	2		S22
			SmithLin)
			2

1

1	1.7 GHz	1.9 GHz	21	50 kHz	0 dBm
2	1.9 GHz	2 GHz	101	10 kHz	-10 dBm
3	2 GHz	2.2 GHz	21	50 kHz	0 dBm

Excel VBA

```
Sub Setup()
    '
    Dim defrm As Long
    Dim vi As Long
    Const TimeOutTime = 20000

    Dim Allocate1 As String, Allocate2 As String, File As String
    Dim Para1(4) As String, Para2(2) As String
    Dim Fmt1(4) As String, Fmt2(2) As String
    Dim Star1(3) As String, Stop1(3) As String
    Dim IfBw1(3) As Double, IfBw2 As Double
    Dim Power1(3) As Double, Power2 As Double
    Dim Cent2 As Double, Span2 As Double
    Dim RefLev1(4) As Double, RefLev2(2) As Double, Scale1(4) As
Double, Scale2(2) As Double
    Dim Segm As Integer, Nop1(3) As Integer, Nop2 As Integer
    Dim NumOfTr1 As Integer, NumOfTr2 As Integer
    Dim RefPos1(4) As Integer, RefPos2(2) As Integer
    Dim SendData As String
    '
    Segm = 3 ' Number of Segment Ch.1 : 3
    Star1(1) = "1.7E9"      ' Start Frequency Ch.1 Segm.1: 1.7 GHz
    Star1(2) = "1.9E9"      ' Segm.2: 1.9 GHz
```

```
Star1(3) = "2E9"        ' Segm.3: 2 GHz
Stop1(1) = "1.9E9"      ' Stop Frequency Ch.1 Segm.1: 1.9 GHz
Stop1(2) = "2E9"        ' Segm.2: 2 GHz
Stop1(3) = "2.2E9"      ' Segm.3: 2.2 GHz
Cent2 = 1900000000#     ' Center Frequency Ch.2 : 1.9 GHz
Span2 = 500000000#      ' Span Ch.2 : 500 MHz
Nop1(1) = 21            ' Number of points Segm.1: 21
Nop1(2) = 101           '                  Segm.2: 101
Nop1(3) = 21            '                  Segm.3: 21
Nop2 = 101              '       Ch.2 : 101
IfBw1(1) = 50000#       ' IFBW  Ch.1    Segm.1: 50 kHz
IfBw1(2) = 10000#       '               Segm.2: 10 kHz
IfBw1(3) = 50000#       '               Segm.3: 50 kHz
IfBw2 = 70000#          '       Ch.2 : 70 kHz
Power1(1) = 0           ' Power Ch.1    Segm.1: 0 dBm
Power1(2) = -10         '               Segm.2: -10 dBm
Power1(3) = 0           '               Segm.3: 0 dBm
Power2 = 0              '       Ch.2 : 0 dBm
NumOfTr1 = 4            ' Number of Trace Ch.1 : 4
NumOfTr2 = 2            ' Ch.2 : 2
Allocate1 = "D12_34"    ' Allocate Traces Ch.1 : D12_34
Allocate2 = "D12"       ' Ch.2 : D12
Para1(1) = "S11"        ' Measurement Ch.1 Trace1: S11
Para1(2) = "S21"        ' Parameter Trace2: S21
```

```
Para1(3) = "S12"        ' Trace3: S12
Para1(4) = "S22"        ' Trace4: S22
Para2(1) = "S31"        ' Ch.2 Trace1: S31
Para2(2) = "S33"        ' Trace2: S33
Fmt1(1) = "SLIN"        ' Data Format Ch.1 Trace1: Smith(Lin/Phase)
Fmt1(2) = "MLOG"        ' Trace2: Log Mag
Fmt1(3) = "MLOG"        ' Trace3: Log Mag
Fmt1(4) = "SLIN"        ' Trace4: Smith(Lin/Phase)
Fmt2(1) = "MLOG"        ' Ch.2 Trace1: Log Mag
Fmt2(2) = "SLIN"        ' Trace2: Smith(Lin/Phase)
RefPos1(1) = 9          ' Reference Ch.1 Trace2: 9
RefPos1(2) = 9          ' Position Trace3: 9
RefPos2(1) = 9          ' Ch.2 Trace1: 9
RefLev1(1) = 0          ' Reference Level Ch.1 Trace2: 0 dBm
RefLev1(2) = 0          ' Trace3: 0 dBm
RefLev2(1) = 0          ' Ch.2 Trace1: 0 dBm
Scale1(1) = 2           ' Scale Ch.1 Trace1: 2
Scale1(2) = 10          ' Trace2: 10 dBm
Scale1(3) = 10          ' Trace3: 10 dBm
Scale1(4) = 2           ' Trace4: 2
Scale2(1) = 10          ' Ch.2 Trace1: 10 dBm
Scale2(2) = 2           ' Trace2: 2
StaFileName = "sample.sta" ' Save File Name : sample.sta
'
```

```
    ' Assigns a GPIB address to the I/O pass.
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime) ' Set time
out
    '
    Call viVPrintf(vi, ":SYST:PRES" + vbLf, 0)
    Call viVPrintf(vi, ":DISP:SPL D1_1_2" + vbLf, 0) 'Allocate Channel
    Call viVPrintf(vi, ":INIT1:CONT ON" + vbLf, 0) 'Turn on Continuous
Activation mode for channel 1
    Call viVPrintf(vi, ":INIT2:CONT ON" + vbLf, 0) 'Turn on Continuous
Activation mode for channel 2
    ' ===============
    ' Setup Channel 1
    ' ===============
    Call viVPrintf(vi, ":SENS1:SWE:TYPE SEGM" + vbLf, 0) 'Sets channel
1 sweep type to segment
    '
    ' Create the data string for Segment Table
    SendData = "5,0,1,1,0,0," & Str(Segm)
    For i = 1 To Segm
        SendData = SendData + "," & Star1(i) & "," + Stop1(i) & "," &
CStr(Nop1(i)) & "," & CStr(IfBw1(i)) & "," & CStr(Power1(i))
    Next i
    Call viVPrintf(vi, ":SENS1:SEGM:DATA " + SendData + vbLf, 0)
    '
```

```
    Call viVPrintf(vi, ":CALC1:PAR:COUN " & CStr(NumOfTr1) & vbLf, 0) 'Set number of traces

    Call viVPrintf(vi, ":DISP:WIND1:SPL " & Allocate1 & vbLf, 0) 'Set graph layout


    For i = 1 To NumOfTr1

        Call viVPrintf(vi, ":CALC1:PAR" & CStr(i) & ":DEF " & Para1(i) & vbLf, 0) 'Set measurement parameter

        Call viVPrintf(vi, ":CALC1:PAR" & CStr(i) & ":SEL" & vbLf, 0) ' Make trace active

        Call viVPrintf(vi, ":CALC1:FORM " & Fmt1(i) & vbLf, 0) ' Set data format

        '

        Select Case Fmt1(i)

        Case "SLIN", "SLOG", "SCOM", "SMIT", "SADM", "PLIN", "PLOG", "POL"

        ' If data format is neither Smith chart nor polar, sets reference division line number and scale per division

            Call viVPrintf(vi, ":DISP:WIND1:TRAC" & CStr(i) & ":Y:PDIV " + CStr(Scale1(i)) + vbLf, 0)

        Case Else

        ' If data format is Smith chart or polar, set full-scale value

            Call viVPrintf(vi, ":DISP:WIND1:TRAC" & CStr(i) & ":Y:RPOS " & CStr(RefPos1(i)) & vbLf, 0)

            Call viVPrintf(vi, ":DISP:WIND1:TRAC" & CStr(i) & ":Y:RLEV " & CStr(RefLev1(i)) & vbLf, 0)

            Call viVPrintf(vi, ":DISP:WIND1:TRAC" & CStr(i) & ":Y:PDIV " & CStr(Scale1(i)) & vbLf, 0)
```

```
        End Select
    Next i
    ' ===============
    ' Setup Channel 2
    ' ===============
    Call viVPrintf(vi, ":SENS2:SWE:TYPE LIN " + vbLf, 0) ' Set sweep type to linear
    Call viVPrintf(vi, ":SENS2:FREQ:CENT " + CStr(Cent2) + vbLf, 0) ' Set center frequency
    Call viVPrintf(vi, ":SENS2:FREQ:SPAN " + CStr(Span2) + vbLf, 0) ' Set span frequency
    Call viVPrintf(vi, ":SENS2:SWE:POIN " + CStr(Nop2) + vbLf, 0) ' Set number of points
    Call viVPrintf(vi, ":SENS2:BAND " + CStr(IfBw2) + vbLf, 0) ' Set IFBW
    Call viVPrintf(vi, ":SOUR2:POW " + CStr(Power2) + vbLf, 0) ' Set power level
    Call viVPrintf(vi, ":CALC2:PAR:COUN " & CStr(NumOfTr2) & vbLf, 0) ' Set number of traces
    Call viVPrintf(vi, ":DISP:WIND2:SPL " & Allocate2 & vbLf, 0) 'Set graph layout

    For i = 1 To NumOfTr2
        Call viVPrintf(vi, ":CALC2:PAR" & CStr(i) & ":DEF " & Para2(i) & vbLf, 0) 'Set measurement parameter
        Call viVPrintf(vi, ":CALC2:PAR" & CStr(i) & ":SEL" & vbLf, 0) ' Make trace active
        Call viVPrintf(vi, ":CALC2:FORM " & Fmt2(i) & vbLf, 0) ' Set data format
```

```
        '
        Select Case Fmt2(i)
        Case "SLIN", "SLOG", "SCOM", "SMIT", "SADM", "PLIN", "PLOG",
"POL"
        ' If data format is neither Smith chart nor polar, sets reference
division line number and scale per division
            Call viVPrintf(vi, ":DISP:WIND2:TRAC" & CStr(i) & ":Y:PDIV " +
CStr(Scale2(i)) + vbLf, 0)
        Case Else
        ' If data format is Smith chart or polar, set full-scale value
            Call viVPrintf(vi, ":DISP:WIND2:TRAC" & CStr(i) & ":Y:RPOS " &
CStr(RefPos2(i)) & vbLf, 0)
            Call viVPrintf(vi, ":DISP:WIND2:TRAC" & CStr(i) & ":Y:RLEV " &
CStr(RefLev2(i)) & vbLf, 0)
            Call viVPrintf(vi, ":DISP:WIND2:TRAC" & CStr(i) & ":Y:PDIV " &
CStr(Scale2(i)) & vbLf, 0)
        End Select
    Next i
    '
    Call viVPrintf(vi, ":MMEM:STOR """ & StaFileName & """" & vbLf, 0) '
Save ENA settings to file
    ' Close IO
    Call viClose(vi)
    Call viClose(defrm)

End Sub
```

HT Basic.htb

```
10 DIM Allocate1$[9],Allocate2$[9],File$[20]
20 DIM Para1$(1:4)[9],Para2$(1:2)[9],Fmt1$(1:4)[9],Fmt2$(1:2)[9]
30 REAL Star1(1:3),Stop1(1:3),Pow1(1:3),Cent2,Span2,Pow2
40 REAL Ref_rev1(1:4),Ref_rev2(1:2),Scale1(1:4),Scale2(1:2)
50 INTEGER Segm,Nop1(1:3),Nop2,Num_of_tr1,Num_of_tr2
60 INTEGER Ref_pos1(1:4),Ref_pos2(1:2),I
70 ASSIGN @Agte507x TO 717
80 !
90 Segm=3 ! Number of Segment Ch.1 : 3
100 Star1(1)=1.7E+9 ! Start Frequency Ch.1 Segm.1: 1.7 GHz
110 Star1(2)=1.9E+9 ! Segm.2: 1.9 GHz
120 Star1(3)=2.E+9 ! Segm.3: 2 GHz
130 Stop1(1)=1.9E+9 ! Stop Frequency Ch.1 Segm.1: 1.9 GHz
140 Stop1(2)=2.E+9 ! Segm.2: 2 GHz
150 Stop1(3)=2.2E+9 ! Segm.3: 2.2 GHz
160 Cent2=1.9E+9 ! Center Frequency Ch.2 : 1.9 GHz
170 Span2=5.00E+8 ! Span Ch.2 : 500 MHz
180 Nop1(1)=21 ! Number Ch.1 Segm.1: 21
190 Nop1(2)=101 ! of Points Segm.2: 101
200 Nop1(3)=21 ! Segm.3: 21
210 Nop2=101 ! Ch.2 : 101
220 If_bw1(1)=5.0E+4 ! IF Bandwidth    Ch.1 Segm.1: 50 kHz
230 If_bw1(2)=1.0E+4 ! Segm.2: 10 kHz
```

```
240 If_bw1(3)=5.0E+4 ! Segm.3: 50 kHz
250 If_bw2=7.0E+4 ! Ch.2 : 70 kHz
260 Pow1(1)=0 ! Power Ch.1 Segm.1: 0 dBm
270 Pow1(2)=-10 ! Segm.2: -10 dBm
280 Pow1(3)=0 ! Segm.3: 0 dBm
290 Pow2=0 ! Ch.2 : 0 dBm
300 Num_of_tr1=4 ! Number Ch.1 : 4
310 Num_of_tr2=2 ! of Traces Ch.2 : 2
320 Allocate1$="D12_34" ! Allocate Traces Ch.1 : D12_34
330 Allocate2$="D12" ! Ch.2 : D12
340 Para1$(1)="S11" ! Measurement Ch.1 Trace1: S11
350 Para1$(2)="S21" ! Parameter Trace2: S21
360 Para1$(3)="S12" ! Trace3: S12
370 Para1$(4)="S22" ! Trace4: S22
380 Para2$(1)="S31" ! Ch.2 Trace1: S31
390 Para2$(2)="S33" ! Trace2: S33
400 Fmt1$(1)="SLIN" ! Data Format Ch.1 Trace1: Smith(Lin/Phase)
410 Fmt1$(2)="MLOG" ! Trace2: Log Mag
420 Fmt1$(3)="MLOG" ! Trace3: Log Mag
430 Fmt1$(4)="SLIN" ! Trace4: Smith(Lin/Phase)
440 Fmt2$(1)="MLOG" ! Ch.2 Trace1: Log Mag
450 Fmt2$(2)="SLIN" ! Trace2: Smith(Lin/Phase)
460 Ref_pos1(2)=9 ! Reference Ch.1 Trace2: 9
470 Ref_pos1(3)=9 ! Position Trace3: 9
```

```
480 Ref_pos2(1)=9 ! Ch.2 Trace1: 9
490 Ref_lev1(2)=0 ! Reference Level Ch.1 Trace2: 0 dBm
500 Ref_lev1(3)=0 ! Trace3: 0 dBm
510 Ref_lev2(1)=0 ! Ch.2 Trace1: 0 dBm
520 Scale1(1)=2 ! Scale Ch.1 Trace1: 2
530 Scale1(2)=10 ! Trace2: 10 dBm
540 Scale1(3)=10 ! Trace3: 10 dBm
550 Scale1(4)=2 ! Trace4: 2
560 Scale2(1)=10 ! Ch.2 Trace1: 10 dBm
570 Scale2(2)=2 ! Trace2: 2
580 File$="sample.sta" ! Save File Name : sample.sta
590 !
600 OUTPUT @Agte507x;":SYST:PRES"
610 !
620 OUTPUT @Agte507x;":DISP:SPL D1_1_2"
630 OUTPUT @Agte507x;":INIT1:CONT ON"
640 OUTPUT @Agte507x;":INIT2:CONT ON"
650 !
660 ! Channel 1
670 !
680 OUTPUT @Agte507x;":SENS1:SWE:TYPE SEGM"
690 OUTPUT @Agte507x;":SENS1:SEGM:DATA 5,0,1,1,0,0,";Segm;",";
700 FOR I=1 TO Segm-1
710 OUTPUT @Agte507x;Star1(I);",";Stop1(I);",";Nop1(I);",";If_bw1
(I);",";Pow1(I);",";
```

```
720 NEXT I

730 OUTPUT
@Agte507x;Star1(Segm);",";Stop1(Segm);",";Nop1(Segm);","
;If_bw1(Segm);",";Pow(Segm)

740 !

750 OUTPUT @Agte507x;":CALC1:PAR:COUN ";Num_of_tr1

760 OUTPUT @Agte507x;":DISP:WIND1:SPL "&Allocate1$

770 FOR I=1 TO Num_of_tr1

780 OUTPUT @Agte507x;":CALC1:PAR"&VAL$(I)&":DEF "&Para1$(I)

790 OUTPUT @Agte507x;":CALC1:PAR"&VAL$(I)&":SEL"

800 OUTPUT @Agte507x;":CALC1:FORM "&Fmt1$(I)

810 SELECT Fmt1$(I)

820 CASE
"SLIN","SLOG","SCOM","SMIT","SADM","PLIN","PLOG","POL"

830 OUTPUT @Agte507x;":DISP:WIND1:TRAC"&VAL$(I)&":Y:PDIV ";
Scale1(I)

840 CASE ELSE

850 OUTPUT @Agte507x;":DISP:WIND1:TRAC"&VAL$(I)&":Y:RPOS ";
Ref_pos1(I)

860 OUTPUT @Agte507x;":DISP:WIND1:TRAC"&VAL$(I)&":Y:RLEV ";
Ref_rev1(I)

870 OUTPUT @Agte507x;":DISP:WIND1:TRAC"&VAL$(I)&":Y:PDIV ";
Scale1(I)

880 END SELECT

890 NEXT I

900 !

910 ! Channel 2
```

```
920 !
930 OUTPUT @Agte507x;":SENS2:SWE:TYPE LIN"
940 OUTPUT @Agte507x;":SENS2:FREQ:CENT ";Cent2
950 OUTPUT @Agte507x;":SENS2:FREQ:SPAN ";Span2
960 OUTPUT @Agte507x;":SENS2:SWE:POIN ";Nop2
970 OUTPUT @Agte507x;":SENS2:BAND ";If_bw2
980 OUTPUT @Agte507x;":SOUR2:POW ";Pow2
990 !
1000 OUTPUT @Agte507x;":CALC2:PAR:COUN ";Num_of_tr2
1010 OUTPUT @Agte507x;":DISP:WIND2:SPL "&Allocate2$
1020 FOR I=1 TO Num_of_tr2
1030 OUTPUT @Agte507x;":CALC2:PAR"&VAL$(I)&":DEF "&Para2$(I)
1040 OUTPUT @Agte507x;":CALC2:PAR"&VAL$(I)&":SEL"
1050 OUTPUT @Agte507x;":CALC2:FORM "&Fmt2$(I)
1060 SELECT Fmt2$(I)
1070 CASE
"SLIN","SLOG","SCOM","SMIT","SADM","PLIN","PLOG","POL"
1080 OUTPUT @Agte507x;":DISP:WIND2:TRAC"&VAL$(I)&":Y:PDIV ";
Scale2(I)
1090 CASE ELSE
1100 OUTPUT @Agte507x;":DISP:WIND2:TRAC"&VAL$(I)&":Y:RPOS ";
Ref_pos2(I)
1110 OUTPUT @Agte507x;":DISP:WIND2:TRAC"&VAL$(I)&":Y:RLEV ";
Ref_rev2(I)
1120 OUTPUT @Agte507x;":DISP:WIND2:TRAC"&VAL$(I)&":Y:PDIV ";
Scale2(I)
```

```
1130 END SELECT
1140 NEXT I
1150 !
1160 OUTPUT @Agte507x;":MMEM:STOR """&File$&""""
1170 END
```

-
- Excel VBA
- HT Basic

Excel VBA

```
Sub Cal_Click()
    Dim defrm As Long          'Session to Default Resource Manager
    Dim vi As Long             'Session to instrument
    Dim Ch As String
    Dim CalKit As Integer
    Dim Port(4) As String

    Const TimeOutTime = 40000   'timeout time.
    Const Cal85032F = 4         'cal kit number.

    Ch = Cells(5, 5)            'Select channel
    Port(1) = Cells(3, 6)       'Sets the select port 1.
    Port(2) = Cells(3, 7)       'Sets the select port 2.
    Port(3) = Cells(3, 8)       'Sets the select port 3.
    Port(4) = Cells(3, 9)       'Sets the select port 4.
    CalKit = Cal85032F          'Sets cal kit (85032F)

    Call viOpenDefaultRM(defrm)    'Initializes the VISA system.
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)    'Opens the session
to the specified instrument.
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)   'The
state of an attribute for the specified session.
```

```
    Call viVPrintf(vi, "*RST" & vbLf, 0)   'Presets the setting state of the
ENA.
    Call viVPrintf(vi, "*CLS" & vbLf, 0)   'Clears the all status register.

    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:CKIT " & CalKit &
vbLf, 0)  'Select the calibration kit

    Select Case Cells(3, 5)
        Case "Response (Open)"   'Perform response calibration (OPEN).
            Call Cal_Resp(vi, Ch, "OPEN", Port(1))
        Case "Response (Short)"   'Perform response calibration (SHORT).
            Call Cal_Resp(vi, Ch, "Short", Port(1))
        Case "Response (Thru)"    'Perform response calibration (Thru).
            Call Cal_RespThru(vi, Ch, "Thru", Port(1), Port(2))
        Case "Full 1 Port"    'Perform 1-port calibration.
            Call Cal_Slot(vi, Ch, 1, Port)
        Case "Full 2 Port"     'Perform full 2-port calibration.
            Call Cal_Slot(vi, Ch, 2, Port)
        Case "Full 3 Port"     'Perform full 3-port calibration.
            Call Cal_Slot(vi, Ch, 3, Port)
        Case "Full 4 Port"    'Perform full 4-port calibration.
            Call Cal_Slot(vi, Ch, 4, Port)
    End Select

    Call viClose(vi)  'Closes the resource manager session.
```

```
    Call viClose(defrm)  'Breaks the communication and terminates the
VISA system.


    End                                                    'End
End Sub


Sub Cal_Resp(vi As Long, Ch As String, CalType As String, Port As
String)
    Dim Dummy As Variant   'Variant to receive the result


    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:" & CalType &
" " & Port & vbLf, 0)  'Sets the calibration type.


    MsgBox ("Set " & CalType & " to Port " & Port & ". then click [OK]
button")  'Display the message box.


    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:" & CalType & " " &
Port & vbLf, 0) 'Measurement the calibration data.
    Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the *OPC?
result.
    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SAVE" & vbLf, 0)
     'Calculating the calibration coefficients.


    Call ErrorCheck(vi)      'Checking the error.


End Sub
```

```
Sub Cal_RespThru(vi As Long, Ch As String, CalType As String, Port1
As String, Port2 As String)

   Dim Dummy As Variant   'Variant to receive the result.


   If Port1 <> Port2 Then

       Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:" & CalType
& " " & Port1 & "," & Port2 & vbLf, 0)   'Sets the calibration type


       MsgBox ("Set " & CalType & " to Port " & Port1 & "&" & Port2 & ".
then click [OK] button")   'Display the message box.


       Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:" & CalType & " " &
Port1 & "," & Port2 & vbLf, 0)  'Measurement the calibration data.

       Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)      'Reads the
*OPC? result.

       Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SAVE" & vbLf, 0)
   'Calculating the calibration coefficients.


       Call ErrorCheck(vi)     'Checking the error.

   Else

       MsgBox ("Thru calibration select port error!")   'Displaying the error
message when selected same ports.

       Exit Sub

   End If


End Sub

Sub Cal_Slot(vi As Long, Ch As String, NumPort As String, Port() As
```

```
String)
    Dim Dummy
    Dim i As Integer, j As Integer


    Select Case NumPort
        Case 1
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" & NumPort & " " & Port(1) & vbLf, 0)   'Set the 1-port calibration type.
        Case 2
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" & NumPort & " " & Port(1) & "," & Port(2) & vbLf, 0)   'Set the full 2-port calibration type.
        Case 3
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" & NumPort & " " & Port(1) & "," & Port(2) & "," & Port(3) & vbLf, 0)   'Set the full 3-port calibration type.
        Case 4
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT4 1,2,3,4" & vbLf, 0)   'Set the full 4-port calibration type.
    End Select
    'Reflection
    For i = 1 To NumPort
        MsgBox ("Set Open to Port " & Port(i) & ". then click [OK] button")  'Display the message box.
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:OPEN " & Port(i) & vbLf, 0)    'Measurement the OPEN calibration.
        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the *OPC? result.
```

```
        MsgBox ("Set Short to Port " & Port(i) & ". then click [OK] button")
   'Display the message box.

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SHORT " & Port(i)
& vbLf, 0)  'Measurement the SHORT calibration.

        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the
*OPC? result.


        MsgBox ("Set Load to Port " & Port(i) & ". then click [OK] button")
  'Display the message box.

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:LOAD " & Port(i) &
vbLf, 0)    'Measurement the LOAD calibration.

        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)     'Reads the
*OPC? result.

    Next i

    'Transmission

    For i = 1 To NumPort - 1

        For j = i + 1 To NumPort

            MsgBox ("Set Thru to Port " & Port(i) & "&" & Port(j) & ". then click
[OK] button")   'Display the message box.

            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:THRU " & Port(i)
& "," & Port(j) & vbLf, 0)    'Measurement the THRU calibration.

            Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the
*OPC result.

            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:THRU " & Port(j)
& "," & Port(i) & vbLf, 0)    'Measurement the THRU calibration.

            Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the
*OPC result.
```

```
        Next j
    Next i
    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SAVE" & vbLf, 0)
 'Calculating the calibration coefficients.


    Call ErrorCheck(vi)    'Checking the error.


End Sub
Sub ErrorCheck(vi As Long)
    Dim err As String * 50, ErrNo As Variant, Response


    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)    'Reads error
message.
    ErrNo = Split(err, ",")    'Gets the error code.


    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)    'Display the
message box.
    End If
End Sub
```

HT Basic.htb

```
10 DIM File$[20],Ch$[9],Inp_char$[9]
20 INTEGER Cal_kit,Cal_type,Port(1:4)
30 !
40 ASSIGN @Agte507x TO 717
50 File$="Ex_4_1.sta"
60 Ch$="1"
70 !
80 Select_cal_kit(@Agte507x,Ch$)
90 !
100 CLEAR SCREEN
110 ON ERROR GOTO Type_select
120 Type_select: !
130 PRINT "## Calibration Type Selection ##"
140 PRINT " 1: Response (Open)"
150 PRINT " 2: Response (Short)"
160 PRINT " 3: Response (Thru)"
170 PRINT " 4: Full 1 Port"
180 PRINT " 5: Full 2 Port"
190 PRINT " 6: Full 3 Port"
200 PRINT " 7: Full 4 Port"
210 PRINT ""
220 PRINT "Input 1 to 7"
230 INPUT "Input number? (1 to 7)",Inp_char$
```

```
240 Cal_type=IVAL(Inp_char$,10)
250 IF Cal_type<1 OR Cal_type>7 THEN Type_select
260 OFF ERROR
270 !
280 SELECT Cal_type
290 CASE 1
300 Select_port(1,Port(*))
310 Cal_resp(@Agte507x,Ch$,"OPEN",Port(1))
320 CASE 2
330 Select_port(1,Port(*))
340 Cal_resp(@Agte507x,Ch$,"SHOR",Port(1))
350 CASE 3
360 Select_port(2,Port(*))
370 Cal_resp_thru(@Agte507x,Ch$,Port(1),Port(2))
380 CASE 4
390 Select_port(1,Port(*))
400 Cal_solt(@Agte507x,Ch$,1,Port(*))
410 CASE 5
420 Select_port(2,Port(*))
430 Cal_solt(@Agte507x,Ch$,2,Port(*))
440 CASE 6
450 Select_port(3,Port(*))
460 Cal_solt(@Agte507x,Ch$,3,Port(*))
470 CASE 7
```

```
480 Select_port(4,Port(*))
490 Cal_solt(@Agte507x,Ch$,4,Port(*))
500 END SELECT
510 !
520 OUTPUT @Agte507x;":MMEM:STOR:STYP CST"
530 OUTPUT @Agte507x;":MMEM:STOR """&File$&"""
540 END
550 !==========================================
560 ! Calibration Kit Selection Function
570 !==========================================
580 SUB Select_cal_kit(@Agte507x,Ch$)
590 DIM Cal_kit_lbl$(1:10)[20],Inp_char$[9]
600 INTEGER Cal_kit,I
610 CLEAR SCREEN
620 !
630 FOR I=1 TO 10
640 OUTPUT @Agte507x;":SENS1:CORR:COLL:CKIT ";I
650 OUTPUT @Agte507x;":SENS1:CORR:COLL:CKIT:LAB?"
660 ENTER @Agte507x;Cal_kit_lbl$(I)
670 NEXT I
680 ON ERROR GOTO Kit_select
690 Kit_select: !
700 PRINT "## Calibration Kit Selection ##"
710 FOR I=1 TO 10
```

```
720 PRINT USING "X,2D,A,X,20A";I,":",Cal_kit_lbl$(I)
730 NEXT I
740 PRINT ""
750 PRINT "Input 1 to 10"
760 INPUT "Input number? (1 to 10)",Inp_char$
770 Cal_kit=IVAL(Inp_char$,10)
780 IF Cal_kit<1 OR Cal_kit>10 THEN Kit_select
790 OFF ERROR
800 !
810 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:CKIT ";Cal_kit
820 SUBEND
830 !==========================================
840 ! Port Selection Function
850 !==========================================
860 SUB Select_port(INTEGER Num_of_ports,INTEGER Port(*))
870 DIM Inp_char$[9]
880 !
890 CLEAR SCREEN
900 IF Num_of_ports=4 THEN
910 Port(1)=1
920 Port(2)=2
930 Port(3)=3
940 Port(4)=4
950 ELSE
```

```
960 PRINT "## Test Ports Selection ##"
970 ON ERROR GOTO Port_select
980 FOR I=1 TO Num_of_ports
990 PRINT "Port("&VAL$(I)&"):";
1000 Port_select:!
1010 INPUT "Number?",Inp_char$
1020 Port(I)=IVAL(Inp_char$,10)
1030 IF Port(I)<1 OR Port(I)>4 THEN Port_select
1040 FOR J=1 TO I-1
1050 IF Port(I)=Port(J) THEN Port_select
1060 NEXT J
1070 PRINT Port(I)
1080 NEXT I
1090 OFF ERROR
1100 END IF
1110 SUBEND
1120 !==========================================
1130 ! Response (Open/Short) Calibration Function
1140 !==========================================
1150 SUB Cal_resp(@Agte507x,Ch$,Type$,INTEGER Port)
1160 DIM Buff$[9]
1170 !
1180 PRINT "## Response ("&Type$&") Calibration ##"
1190 OUTPUT
@Agte507x;":SENS"&Ch$&":CORR:COLL:METH:"&Type$&" ";Port
```

```
1200 PRINT "Set "&Type$&" to Port "&VAL$(Port)&". Then push [Enter] key."

1210 INPUT "",Buff$

1220 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:"&Type$&" ";Port

1230 OUTPUT @Agte507x;"*OPC?"

1240 ENTER @Agte507x;Buff$

1250 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:SAVE"

1260 PRINT "Done"

1270 SUBEND

1280 !==============================================

1290 ! Response (Thru) Calibration Function

1300 !==============================================

1310 SUB Cal_resp_thru(@Agte507x,Ch$,INTEGER Port1,Port2)

1320 DIM Buff$[9]

1330 !

1340 PRINT "## Response (Thru) Calibration ##"

1350 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:METH:THRU ";Port1;","; Port2

1360 PRINT "Set THRU between Port "&VAL$(Port1)&" and Port "&VAL$(Port2 )&". Then push [Enter] key."

1370 INPUT "",Buff$

1380 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:THRU ";Port1;",";Port2

1390 OUTPUT @Agte507x;"*OPC?"

1400 ENTER @Agte507x;Buff$
```

```
1410 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:SAVE"
1420 PRINT "Done"
1430 SUBEND
1440 !==============================================
1450 ! Full n Port Calibration Function
1460 !==============================================
1470 SUB Cal_solt(@Agte507x,Ch$,INTEGER Num_of_ports,INTEGER
Port(*))
1480 DIM Buff$[9]
1490 INTEGER I,J
1500 !
1510 PRINT "## Full "&VAL$(Num_of_ports)&" Port Calibration ##"
1520 !
1530 ! Calibration Type Selection
1540 !
1550 OUTPUT
@Agte507x;":SENS"&Ch$&":CORR:COLL:METH:SOLT"&VAL$(Num_of_
ports)&" ";
1560 FOR I=1 TO Num_of_ports-1
1570 OUTPUT @Agte507x;Port(I);",";
1580 NEXT I
1590 OUTPUT @Agte507x;Port(Num_of_ports)
1600 !
1610 ! Reflection Measurement
1620 !
```

```
1630 FOR I=1 TO Num_of_ports
1640 PRINT "Set OPEN to Port "&VAL$(Port(I))&". Then push [Enter] key."
1650 INPUT "",Buff$
1660 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:OPEN ";Port(I)
1670 OUTPUT @Agte507x;"*OPC?"
1680 ENTER @Agte507x;Buff$
1690 PRINT "Set SHORT to Port "&VAL$(Port(I))&". Then push [Enter] key."
1700 INPUT "",Buff$
1710 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:SHOR ";Port(I)
1720 OUTPUT @Agte507x;"*OPC?"
1730 ENTER @Agte507x;Buff$
1740 PRINT "Set LOAD to Port "&VAL$(Port(I))&". Then push [Enter] key."
1750 INPUT "",Buff$
1760 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:LOAD ";Port(I)
1770 OUTPUT @Agte507x;"*OPC?"
1780 ENTER @Agte507x;Buff$
1790 NEXT I
1800 !
1810 ! Transmission Measurement
1820 !
1830 FOR I=1 TO Num_of_ports-1
```

```
1840 FOR J=I+1 TO Num_of_ports

1850 PRINT "Set THRU between Port "&VAL$(Port(I))&" and Port "&
VAL$(Port(J))&". Then push [Enter] key."

1860 INPUT "",Buff$

1870 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:THRU
";Port(I);"," ;Port(J)

1880 OUTPUT @Agte507x;"*OPC?"

1890 ENTER @Agte507x;Buff$

1900 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:THRU
";Port(J);"," ;Port(I)

1910 OUTPUT @Agte507x;"*OPC?"

1920 ENTER @Agte507x;Buff$

1930 NEXT J

1940 NEXT I

1950 !

1960 ! Done

1970 !

1980 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:SAVE"

1990 PRINT "Done"

2000 SUBEND
```

-
- Excel VBA
- HT Basic

Excel VBA

```
Sub ECal_Click()
    Dim defrm As Long          'Session to Default Resource Manager
    Dim vi As Long             'Session to instrument
    Dim Ch As String
    Dim CalKit As Integer
    Dim Port(4) As String
    Const TimeOutTime = 40000   'timeout time.

    Ch = Cells(5, 5)           'Select channel
    Port(1) = Cells(3, 6)       'Sets the select port 1.
    Port(2) = Cells(3, 7)       'Sets the select port 2.
    Port(3) = Cells(3, 8)       'Sets the select port 3.
    Port(4) = Cells(3, 9)       'Sets the select port 4.

    Call viOpenDefaultRM(defrm)     'Initializes the VISA system.
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)      'Opens the
session to the specified instrument.
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)   'The
state of an attribute for the specified session.

    Call viVPrintf(vi, "*RST" & vbLf, 0)   'Presets the setting state of the
ENA.
    Call viVPrintf(vi, "*CLS" & vbLf, 0)   'Clears the all status register.
```

```
    Select Case Cells(3, 5)
        Case "1 Port"
            Call ECal(vi, Ch, 1, Port)   'Perform 1-port calibration.
        Case "2 Port"
            Call ECal(vi, Ch, 2, Port)   'Perform full 2-port calibration.
        Case "3 Port"
            Call ECal(vi, Ch, 3, Port)   'Perform full 3-port calibration.
        Case "4 Port"
            Call ECal(vi, Ch, 4, Port)   'Perform full 4-port calibration.
    End Select

    Call viClose(vi)    'Closes the resource manager session.
    Call viClose(defrm)    'Breaks the communication and terminates the
VISA system.

    End
End Sub
Sub ECal(vi As Long, Ch As String, NumPort As String, Port() As String)
    Dim Dummy As Variant
    Dim i As Integer, j As Integer

    Select Case NumPort
        Case 1
            MsgBox ("Connect Port " & Port(1) & ". then click [OK] button")
 'Display the message box.
```

```
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:ECAL:SOLT" &
NumPort & " " & Port(1) & vbLf, 0) 'Execute the 1-port calibration.

    Case 2

        MsgBox ("Connect Port " & Port(1) & " and Port " & Port(2) & ".
then click [OK] button")   'Display the message box.

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:ECAL:SOLT" &
NumPort & " " & Port(1) & "," & Port(2) & vbLf, 0) 'Execute the full 2-port
calibration.

    Case 3

        MsgBox ("Connect Port " & Port(1) & "," & Port(2) & " and Port " &
Port(3) & ". then click [OK] button")  'Display the message box.

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:ECAL:SOLT" &
NumPort & " " & Port(1) & "," & Port(2) & "," & Port(3) & vbLf, 0)
  'Execute the full 3-port calibration.

    Case 4

        MsgBox ("Connect Port 1, 2, 3 and 4. then click [OK] button")
   'Display the message box.

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:ECAL:SOLT4
1,2,3,4" & vbLf, 0) 'Execute the full 4-port calibration.

   End Select


   Call ErrorCheck(vi)      'Checking the error.


End Sub

Sub ErrorCheck(vi As Long)

   Dim err As String * 50, ErrNo As Variant, Response


   Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)    'Reads error
```

```
message.
    ErrNo = Split(err, ",")   'Gets the error code.


    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)    'Display the
message box.
    End If
End Sub
```

HT Basicecal.htb

```
10 DIM File$[20],Ch$[9],Inp_char$[9]
20 INTEGER Cal_kit,Cal_type,Port(1:4)
30 !
40 ASSIGN @Agte507x TO 717
50 File$="Ex_4_2.sta"
60 Ch$="1"
70 !
80 CLEAR SCREEN
90 ON ERROR GOTO Type_select
100 Type_select: !
110 PRINT "## Calibration Type Selection ##"
120 PRINT " 1: Full 1 Port"
130 PRINT " 2: Full 2 Port"
140 PRINT " 3: Full 3 Port"
150 PRINT " 4: Full 4 Port"
160 PRINT ""
170 PRINT "Input 1 to 4"
180 INPUT "Input number? (1 to 4)",Inp_char$
190 Cal_type=IVAL(Inp_char$,10)
200 IF Cal_type<1 OR Cal_type>4 THEN Type_select
210 OFF ERROR
220 !
230 Select_port(Cal_type,Port(*))
```

```
240 Ecal(@Agte507x,Ch$,Cal_type,Port(*))
250 !
260 OUTPUT @Agte507x;":MMEM:STOR:STYP CST"
270 OUTPUT @Agte507x;":MMEM:STOR """&File$&"""
280 END
290 !==============================================
300 ! Port Selection Function
310 !==============================================
320 SUB Select_port(INTEGER Num_of_ports,INTEGER Port(*))
330 DIM Inp_char$[9]
340 !
350 CLEAR SCREEN
360 IF Num_of_ports=4 THEN
370 Port(1)=1
380 Port(2)=2
390 Port(3)=3
400 Port(4)=4
410 ELSE
420 PRINT "## Test Ports Selection ##"
430 ON ERROR GOTO Port_select
440 FOR I=1 TO Num_of_ports
450 PRINT "Port("&VAL$(I)&"):";
460 Port_select: !
470 INPUT "Number?",Inp_char$
```

```
480 Port(I)=IVAL(Inp_char$,10)
490 IF Port(I)<1 OR Port(I)>4 THEN Port_select
500 FOR J=1 TO I-1
510 IF Port(I)=Port(J) THEN Port_select
520 NEXT J
530 PRINT Port(I)
540 NEXT I
550 OFF ERROR
560 END IF
570 SUBEND
580 !==========================================
590 ! Electronic Calibration Function
600 !==========================================
610 SUB Ecal(@Agte507x,Ch$,INTEGER Num_of_ports,INTEGER
Port(*))
620 DIM Buff$[9],Err_msg$[100]
630 INTEGER Err_no,Port1
640 !
650 PRINT "## Full "&VAL$(Num_of_ports)&" Port ECal ##"
660 !
670 OUTPUT @Agte507x;"*CLS"
680 SELECT Num_of_ports
690 CASE 1
700 PRINT "Connect Port "&VAL$(Port(1))&" to ECal Module."
710 PRINT "Then push [Enter] key."
```

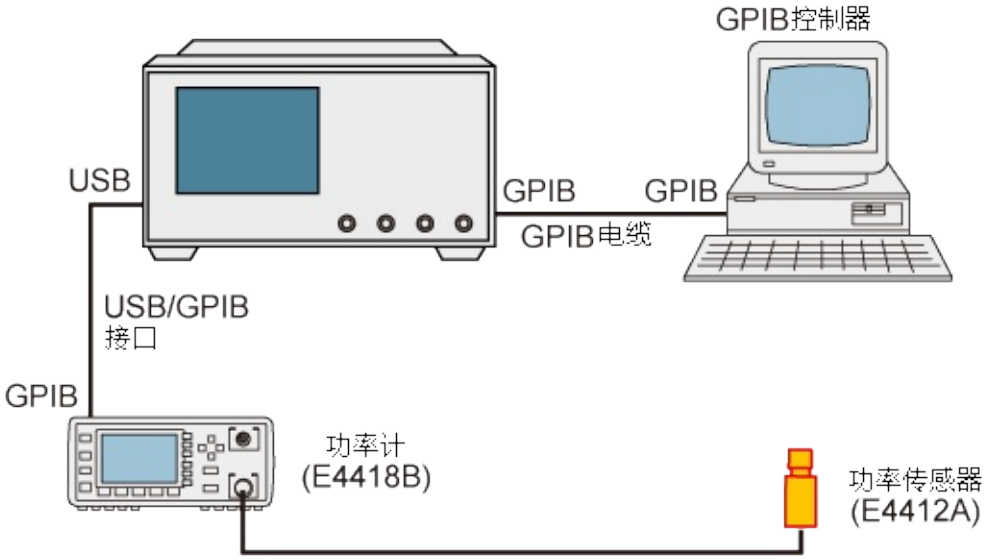
```
720 INPUT "",Buff$

730 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:ECAL:SOLT1
";Port(1)

740 CASE 2

750 PRINT "Connect Port "&VAL$(Port(1));

760 PRINT " and Port "&VAL$(Port(2))&" to ECal Module."

770 PRINT "Then push [Enter] key."

780 INPUT "",Buff$

790 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:ECAL:SOLT2
";Port(1); ",";Port(2)

800 CASE 3

810 PRINT "Connect Port "&VAL$(Port(1));

820 PRINT ", Port "&VAL$(Port(2));

830 PRINT " and Port "&VAL$(Port(3))&" to ECal Module."

840 PRINT "Then push [Enter] key."

850 INPUT "",Buff$

860 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:ECAL:SOLT3
";Port(1); ",";Port(2);",";Port(3)

870 CASE 4

880 PRINT "Connect Port 1, Port 2, Port 3 and Port 4 to to ECal Mod
ule."

890 PRINT "Then push [Enter] key."

900 INPUT "",Buff$

910 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:ECAL:SOLT4
1,2,3,4"

920 END SELECT
```

```
930 PRINT "Executing ..."
940 OUTPUT @Agte507x;":SYST:ERR?"
950 ENTER @Agte507x;Err_no,Err_msg$
960 IF Err_no<>0 THEN
970 PRINT "Error occurred!!"
980 PRINT " No:";Err_no,"Description: "&Err_msg$
990 PRINT "ECAL INTERRUPT!!"
1000 ELSE
1010 PRINT "Done"
1020 END IF
1030 SUBEND
```

-
- Excel VBA
- HT Basic

USB/GPIBE5071C(E4418B)GPIBE5071C(E4412A)
E5071C

E5071C



e5071c334

Excel VBA

```
Sub pow_cal_Click()
    '
    Dim defrm As Long
    Dim vi As Long
    Dim SwpType As String, StartPower As String, StopPower As String, CwFreq As String
    Dim Nop As Long, NumOfAve As String, Limit As Double, CorrData() As Double
    Dim Result As String * 10000, OpcRes As String * 2, Res As Variant
    Dim i As Long, Stat As VbMsgBoxResult
    Dim err As String * 50, ErrNo As Variant
    Const TimeOutTime = 50000 ' TimeOut time should be greater than the measurement time.
    ' Assign a GPIB address to the I/O pass.
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    '
    SwpType = "POW"         ' Sweep type              : POWER
    Nop = 21                ' Number of points        : 41
    StartPower = "-20"      ' Start Power             : -20 dBm
    StopPower = "-10"       ' Stop Power              : -10 dBm
    CwFreq = "1E9"          ' CW frequency            : 1 GHz
    NumOfAve = "4"          ' Number of averaging     : 4
```

```
    Limit = 10              ' Limit for corrected data : 10 dBm
    '

    Call viVPrintf(vi, ":SYST:PRES" + vbLf, 0) ' Presetting the analyzer

    Call viVPrintf(vi, ":SYST:COMM:GPIB:PMET:ADDR 13" + vbLf, 0) '
Setting GPIB address of the power meter to ENA
    '

    ' Setting measurement conditions

    Call viVPrintf(vi, ":SENS1:SWE:TYPE " & SwpType & vbLf, 0)

    Call viVPrintf(vi, ":SENS1:SWE:POIN " & CStr(Nop) & vbLf, 0)

    Call viVPrintf(vi, ":SOUR1:POW:STAR " & StartPower & vbLf, 0)

    Call viVPrintf(vi, ":SOUR1:POW:STOP " & StopPower & vbLf, 0)

    Call viVPrintf(vi, ":SENS1:FREQ " & CwFreq & vbLf, 0)


    Stat = MsgBox("Do you perform zeroing and calibrating the power
sensor?", vbYesNo)


    If Stat = vbYes Then

        MsgBox "Zero and calibrate the power sensor by using the power
meter, then press [OK] key.", vbOKOnly

    End If


MeasStart:
    '

    ' Connecting the power sensor A to the port-1 of ENA

    Call viVPrintf(vi, "*CLS" + vbLf, 0)
```

```
    Stat = MsgBox("Set the power sensor connected to the port 1 in the
ENA, then press [OK] key.", vbOKOnly)
    '
    ' Performing power calibration measurement
    Call viVPrintf(vi, ":SOUR1:POW:PORT1:CORR:COLL:AVER " &
NumOfAve & vbLf, 0)
    Call viVPrintf(vi, ":SOUR1:POW:PORT1:CORR:COLL ASEN" + vbLf,
0)
    Call viVPrintf(vi, "*OPC?" + vbLf, 0)
    Call viVScanf(vi, "%t", OpcRes)
    '
    ' Error hnadling at power meter measurement
    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)
    ErrNo = Split(err, ",")
    '
    If Val(ErrNo(0)) = 0 Then
        ReDim CorrData(Nop)

        Call viVPrintf(vi, ":FORM:DATA ASC" + vbLf, 0)
        Call viVPrintf(vi, ":SOUR1:POW:PORT1:CORR:DATA?" + vbLf, 0)
        Call viVScanf(vi, "%t", Result)
        Res = Split(Result, ",")

        If fnLim(vi, Nop, Limit, Res) Then
            MsgBox "Power meter calibration measurement is complete.",
vbOKOnly
```

```
            For i = 1 To Nop
                Cells(i + 5, 2) = i
                Cells(i + 5, 3) = Res(i - 1)
            Next i
        Else
            GoTo ReCalibration
        End If
    Else
        MsgBox "Error", vbOKOnly
        GoTo ReCalibration
    End If
ProgEnd:
    Call viClose(vi)
    Call viClose(defrm)
    Exit Sub
ReCalibration:
        Stat = MsgBox("Do you perform the power meter calibration
measurement again?", vbYesNo)
        If Stat = vbYes Then
            GoTo MeasStart
        Else
            GoTo ProgEnd
        End If
End Sub
Function fnLim(vi As Long, Nop As Long, Limit As Double, Res As
```

```
Variant) As Boolean
    For i = 1 To Nop
        If Abs(Res(i - 1)) > Limit Then
            Call viVPrintf(vi, "SOUR1:POW:PORT1:CORR OFF" + vbLf, 0)
            MsgBox "The corrected data is out of limit!", vbOKOnly
            fnLim = False
            Exit Function
        End If
    Next i
    fnLim = True
End Function
```

HT Basic (pow_cal.htb)

```
10 DIM Swp_type$[11],Inp_char$[9],Buff$[9],Err_mes$[50],File$[20]
20 DIM Corr_data(1:1601)
30 REAL Start_p,Stop_p,Cw_freq,Limit
40 INTEGER Nop,Pow_rang,Num_avg,Err_no,Verifier,Data_size,I
50 CLEAR SCREEN
60 !
70 ASSIGN @Agte507x TO 717
80 !
90 Swp_type$="POW" !Sweep type : POWER
100 Nop=41 !Number of points : 41
110 Pow_rang=0 !Power Range :-20 to +12 dBm
120 Start_p=-2.0E+1 !Start Power :-20 dBm
130 Stop_p=-1.0E+1 !Stop Power :-10 dBm
140 Cw_freq=1.0E+9 !CW frequency : 1 GHz
150 Num_avg=4 !Number of averaging : 4
160 Limit=10 !limit for corrected data : 10 dBm
170 !
180 ! Presetting the analyzer
190 !
200 OUTPUT @Agte507x;":SYST:PRES"
210 !
220 ! Setting GPIB address of the power meter to E5071C
230 !
```

```
240 OUTPUT @Agte507x;":SYST:COMM:GPIB:PMET:ADDR 13"
250 !
260 ! Setting measurement conditions
270 !
280 OUTPUT @Agte507x;":SENS1:SWE:TYPE "&Swp_type$
290 OUTPUT @Agte507x;":SENS1:SWE:POIN ";Nop
300 OUTPUT @Agte507x;":SOUR1:POW:ATT ";Pow_rang
310 OUTPUT @Agte507x;":SOUR1:POW:STAR ";Start_p
320 OUTPUT @Agte507x;":SOUR1:POW:STOP ";Stop_p
330 OUTPUT @Agte507x;":SENS1:FREQ ";Cw_freq
340 !
350 PRINT "Do you perform zeroing and calibrating the power sensor?"
360 PRINT
370 INPUT "[Y/N]",Inp_char$
380 IF UPC$(Inp_char$)="Y" THEN
390 PRINT "Zero and calibrate the power sensor by using the power
meter, then press [Enter] key."
400 PRINT
410 INPUT "",Inp_char$
420 END IF
430 !
440 Meas_start: !
450 !
460 ! Connecting the power sensor to the port 1 in the ENA
470 !
```

```
480 OUTPUT @Agte507x;"*CLS"

490 PRINT "Set the power sensor connected to the port 1 in the ENA,
then press [Enter] key."

500 PRINT

510 INPUT "",Inp_char$

520 !

530 ! Performing power calibration measurement

540 !

550 OUTPUT @Agte507x;":SOUR1:POW:PORT1:CORR:COLL:AVER
";Num_avg

560 OUTPUT @Agte507x;":SOUR1:POW:PORT1:CORR:COLL ASEN"

570 OUTPUT @Agte507x;"*OPC?"

580 ENTER @Agte507x;Buff$

590 !

600 ! Error hnadling at power meter measurement

610 !

620 OUTPUT @Agte507x;":SYST:ERR?"

630 ENTER @Agte507x;Err_no,Err_mes$

640 !

650 IF Err_no=0 THEN

660 REDIM Corr_data(1:Nop)

670 OUTPUT @Agte507x;":FORM:DATA ASC"

680 OUTPUT @Agte507x;":SOUR1:POW:PORT1:CORR:DATA?"

690 ENTER @Agte507x;Corr_data(*)

700 Verifier=FNLim(@Agte507x,Nop,Limit,Corr_data(*))
```

```
710 IF Verifier=-1 THEN
720 PRINT "Do you perform the power meter calibration measurement again?"
730 PRINT
740 INPUT "[Y/N]",Inp_char$
750 IF UPC$(Inp_char$)="Y" THEN GOTO Meas_start
760 IF UPC$(Inp_char$)<>"Y" THEN GOTO Prog_stop
770 END IF
780 PRINT "Power meter calibration measurement is complete."
790 PRINT
800 ELSE
810 PRINT "Error: "&Err_mes$
820 PRINT
830 PRINT "Do you perform the power meter calibration measurement again?"
840 PRINT
850 INPUT "[Y/N]",Inp_char$
860 IF UPC$(Inp_char$)="Y" THEN GOTO Meas_start
870 IF UPC$(Inp_char$)<>"Y" THEN GOTO Prog_stop
880 END IF
890 !
900 ! Installing the corrected data to a file
910 !
920 File$="CORR_DATA"
930 Data_size=Nop*8
```

```
940 ON ERROR GOTO Skip_purge
950 PURGE File$
960 Skip_purge: OFF ERROR
970 PRINT "The file installing power correction data: "&File$
980 PRINT
990 CREATE File$,Data_size
1000 ASSIGN @File TO File$;FORMAT ON
1010 FOR I=1 TO Nop
1020 OUTPUT @File USING "3D,3X,MD.4DE";I,Corr_data(I)
1030 NEXT I
1040 ASSIGN @File TO *
1050 !
1060 PRINT "Installing the corrected data to the file is DONE."
1070 !
1080 GOTO Prog_end
1090 !
1100 Prog_stop: !
1110 PRINT "Program Interruption"
1120 !
1130 Prog_end: !
1140 !
1150 END
1160 !
1170 ! Limit Test Function for the Corrected Data
```

```
1180 !
1190 DEF FNLim(@Agte507x,INTEGER Nop,REAL Limit,REAL
Corr_data(*))
1200 INTEGER I
1210 !
1220 FOR I=1 TO Nop
1230 IF ABS(Corr_data(I))>Limit THEN
1240 OUTPUT @Agte507x;"SOUR1:POW:PORT1:CORR OFF"
1250 PRINT "The corrected data is out of limit!"
1260 PRINT
1270 RETURN -1
1280 GOTO Fn_exit
1290 END IF
1300 NEXT I
1310 !
1320 RETURN 0
1330 !
1340 Fn_exit: !
1350 !
1360 FNEND
```

/

-
- Excel VBA
- HT Basic

/

2E5071C

E5071C

Excel VBA

```
Sub Err_Term_Click()
    Dim defrm As Long          'Session to Default Resource Manager
    Dim vi As Long             'Session to instrument
    Dim Ch As String
    Dim CalKit As Integer
    Dim Port(2) As String
    Dim Result As String * 10
    Dim tNop As Long
    Dim Respons As String
    Dim Stimulus As String
    Dim ErrTerm As String

    Const TimeOutTime = 40000   'timeout time.
    Const Cal85032F = 4         'cal kit number

    Ch = Cells(2, 6)            'Select channel
    Port(1) = Cells(4, 6)       'Sets the select port 1.
    Port(2) = Cells(5, 6)       'Sets the select port 2.
    Respons = Cells(6, 6)       'Sets the respons port.
    Stimulus = Cells(7, 6)      'Sets the stimulus port.
    ErrTerm = Cells(8, 6)       'Sets the error term.

    CalKit = Cal85032F          'Set cal kit (85032F)
```

```
    Call viOpenDefaultRM(defrm)    'Initializes the VISA system.

    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)    'Opens the session
to the specified instrument.

    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)      'The
state of an attribute for the specified session.


    Call viVPrintf(vi, "*RST" & vbLf, 0)   'Presets the setting state of the
ENA.

    Call viVPrintf(vi, "*CLS" & vbLf, 0)   'Clears the all status register.


    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:CKIT " & CalKit &
vbLf, 0)  'Select the calibration kit.


    Call Set_sgm_tbl(vi, Ch)     'Configures the segment table.


    Select Case Cells(3, 6)    'Sets the read/write.

        Case "Read"

            Call Cal_Slot(vi, Ch, 2, Port)  'Full 2-Port Calibration.

        Case "Write"

            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COEF:METH:SOLT2
1,2" & vbLf, 0)   'Sets the calibration type to the full 2-port calibration.

    End Select


    Call viVPrintf(vi, ":SENS" & Ch & ":SEGM:SWE:POIN?" & vbLf, 0)
 'Reads out the total number of the measurement points of all segments.
```

```
    Call viVScanf(vi, "%t", Result)

    Call Exec_Error_Term(vi, Ch, Val(Result), ErrTerm, Respons,
Stimulus)    'Reads the error coefficient.

    Call viClose(vi)      'Closes the resource manager session.
    Call viClose(defrm)      'Breaks the communication and terminates the
VISA system.

    End
End Sub
Sub Exec_Error_Term(vi As Long, Ch As String, Nop As Long, ErrTerm
As String, Respons As String, Stimulus As String)

    Dim Error_Term_Data As Variant
    Dim Freq_Data As Variant
    Dim i As Integer, j As Integer
    Dim SelMode As String
    Dim Result As String * 10000
    Dim RealData As Double
    Dim ImagData As Double
    Dim FreqData As Double

    ReDim Error_Term_Data(Nop * 2) As String        'Defines the stock
variables for the error coefficient as needed for NOP.
    ReDim Freq_Data(Nop) As String                  'Defines the stock
```

variables for the frequency values.

```
    SelMode = Cells(3, 6)   'Reads the read/write mode.

    Select Case SelMode
        Case "Read"                             'Reads the error coefficient from the ena.
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COEF? " & ErrTerm & "," & Respons & "," & Stimulus & vbLf, 0)  'Read the calibration coefficient data.
            Call viVScanf(vi, "%t", Result)
            Error_Term_Data = Split(Result, ",")    'Splits the read data by comma.

            Freq_Data = Make_Freq(vi, Nop)       'Calculates the frequency values.

            For i = 0 To Nop - 1
                RealData = CDbl(Error_Term_Data(i * 2))       'Reads the real data from error coefficient items.
                ImagData = CDbl(Error_Term_Data(i * 2 + 1))  'Reads the imag data from error coefficient items.
                FreqData = CDbl(Freq_Data(i + 1))                'Reads the frequency values.
                Cells(10 + i, 2) = RealData                         'Displays the real data to the excel sheet.
                Cells(10 + i, 3) = ImagData                         'Displays the imag data to the excel sheet.
```

```
            Cells(10 + i, 1) = FreqData                         'Displays the
frequency values to the excel sheet.
        Next i

        Call Data_Plot(vi, Nop, ErrTerm)                        'Displays the
graph to the excel sheet.

      Case "Write"     'Write the error coefficient to the ena.
        Error_Term_Data = ErrTerm & "," & Respons & "," & Stimulus
   'Sets the command parameter.
        For i = 0 To Nop - 1
            RealData = Cells(10 + i, 2)                         'Retrieves the
real data from the excel sheet.
            ImagData = Cells(10 + i, 3)                         'Retrieves the
imag data from the excel sheet.
            Error_Term_Data = Error_Term_Data & "," & RealData & "," &
ImagData        'Sets the command parameter.
        Next i

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COEF " &
Error_Term_Data & vbLf, 0)    'Write the calibration coefficient data.
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COEF:SAVE" & vbLf, 0)
      'Calculates the calibration coefficients.

   End Select
End Sub
Function Make_Freq(vi As Long, tPoint As Long) As Variant
```

```
    Dim start_freq As Double
    Dim stop_freq As Double
    Dim Nop As Integer
    Dim fStep As Double
    Dim fPoint As Double
    Dim freq_arry() As Variant
    Dim MeasPoint As Integer

    Const SegmentCnt = 2        'number of segment table.

    ReDim freq_arry(tPoint) As Variant

    MeasPoint = 1

    For j = 1 To SegmentCnt
        start_freq = Cells(3 + j - 1, 9)                    'Sets the start
frequency of segment table.
        stop_freq = Cells(3 + j - 1, 10)                    'Sets the
stop frequency of segment table.
        Nop = Cells(3 + j - 1, 13)                          'Sets the nop
of segment table.
        fStep = (stop_freq - start_freq) / (Nop - 1)
                        'Calculate the frequency step.

        fPoint = start_freq                                 'Sets the
frequency start point.
```

```
        For i = 1 To Nop
            freq_arry(MeasPoint) = fPoint                                   'Sets the
frequency value.
            fPoint = fPoint + fStep                                         'Calculate the
frequency points.
            MeasPoint = MeasPoint + 1                                       'Add to
measurement points.
        Next i
    Next j


    Make_Freq = freq_arry                                                   'Sets the
frequency data array.


End Function
Sub Data_Plot(vi As Long, Nop As Long, ErrTerm As String)
    Range("B10:C" & Nop + 9 & "").Select    'Select the error coefficient.
    Charts.Add
    ActiveChart.ChartType = xlLineStacked     'Sets the chart type.
    ActiveChart.SetSourceData Source:=Sheets("Error
Term").Range("A9:C" & Nop + 9 & "") 'Sets the error coefficient and
displays the graph.
    ActiveChart.Location Where:=xlLocationAsObject, Name:="Error Term"


    ActiveChart.Axes(xlCategory).Select    'Select the formatting of X-axis.
    With Selection
        .TickLabelPosition = xlLow     'Displays the frequency values to low
area.
```

```
    End With

    With ActiveChart
        .HasTitle = True
        .ChartTitle.Characters.Text = "Error Term " & ErrTerm     'Display the
title.
    End With
End Sub
Sub Set_sgm_tbl(vi As Long, Ch As String)
    Dim Star1(2) As Double, Stop1(2) As Double, Pow1(2) As Double,
If_bw1(2) As Double
    Dim Segm As Integer, Nop1(2) As Integer, Num_of_tr1 As Integer
    Dim i As Integer

    Segm = 2
    Star1(1) = Cells(3, 9)                 'Sets the start frequency of segment 1
table.
    Stop1(1) = Cells(3, 10)                 'Sets the stop frequency of segment
1 table.
    Pow1(1) = Cells(3, 11)                 'Sets the power of segment 1 table.
    If_bw1(1) = Cells(3, 12)               'Sets the ifbw of segment 1 table.
    Nop1(1) = Cells(3, 13)                 'Sets the nop of segment 1 table.
    Star1(2) = Cells(4, 9)                 'Sets the start frequency of segment 2
table.
    Stop1(2) = Cells(4, 10)                 'Sets the stop frequency of segment
2 table.
```

```
    Pow1(2) = Cells(4, 11)                'Sets the power of segment 2 table.

    If_bw1(2) = Cells(4, 12)              'Sets the ifbw of segment 2 table.

    Nop1(2) = Cells(4, 13)                'Sets the nop of segment 2 table.


    Call viVPrintf(vi, ":SENS" & Ch & ":SWE:TYPE SEGM" & vbLf, 0)
  'Sets sweep type to segment.

    Call viVPrintf(vi, ":SENS" & Ch & ":SEGM:DATA 5,0,1,1,0,0," & Segm
& ",", 0)       'Sets the header of segment table.

    Call viVPrintf(vi, Star1(1) & "," & Stop1(1) & "," & Nop1(1) & "," &
If_bw1(1) & "," & Pow1(1) & ",", 0)  'Sets the 1st parameter.

    Call viVPrintf(vi, Star1(2) & "," & Stop1(2) & "," & Nop1(2) & "," &
If_bw1(2) & "," & Pow1(2) & vbLf, 0)  'Sets the 2nd parameter.


    Call ErrorCheck(vi)   'Checking the error.


End Sub

Sub Cal_Slot(vi As Long, Ch As String, NumPort As String, Port() As
String)

    Dim Dummy

    Dim i As Integer, j As Integer


    Select Case NumPort

        Case 1

            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" &
NumPort & " " & Port(1) & vbLf, 0)   'Set the 1-port calibration type.

        Case 2
```

```
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" & NumPort & " " & Port(1) & "," & Port(2) & vbLf, 0)  'Set the full 2-port calibration type.

    Case 3

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" & NumPort & " " & Port(1) & "," & Port(2) & "," & Port(3) & vbLf, 0)   'Set the full 3-port calibration type.

    Case 4

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT4 1,2,3,4" & vbLf, 0)    'Set the full 4-port calibration type.

  End Select

  'Reflection

  For i = 1 To NumPort

    MsgBox ("Set Open to Port " & Port(i) & ". then click [OK] button")  'Display the message box.

    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:OPEN " & Port(i) & vbLf, 0)      'Measurement the OPEN calibration.

    Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)     'Reads the *OPC? result.


    MsgBox ("Set Short to Port " & Port(i) & ". then click [OK] button")  'Display the message box.

    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SHORT " & Port(i) & vbLf, 0)  'Measurement the SHORT calibration.

    Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)  'Reads the *OPC? result.


    MsgBox ("Set Load to Port " & Port(i) & ". then click [OK] button")  'Display the message box.
```

```
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:LOAD " & Port(i) &
vbLf, 0)   'Measurement the LOAD calibration.

        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)   'Reads the
*OPC? result.

    Next i

    'Transmission

    For i = 1 To NumPort - 1

        For j = i + 1 To NumPort

            MsgBox ("Set Thru to Port " & Port(i) & "&" & Port(j) & ". then click
[OK] button")  'Display the message box.

            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:THRU " & Port(i)
& "," & Port(j) & vbLf, 0)   'Measurement the THRU calibration.

            Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)   'Reads the
*OPC? result.

            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:THRU " & Port(j)
& "," & Port(i) & vbLf, 0)   'Measurement the THRU calibration.

            Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)      'Reads the
*OPC? result.

        Next j

    Next i


    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SAVE" & vbLf, 0)
   'Calculating the calibration coefficients.


    Call ErrorCheck(vi)   'Checking the error.


End Sub
```

```
Sub ErrorCheck(vi As Long)
    Dim err As String * 50, ErrNo As Variant, Response

    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)   'Reads error
message.
    ErrNo = Split(err, ",")      'Gets the error code.

    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)    'Display the
message box.
    End If
End Sub
```

HT BasicErrTerm.htb

```
2000 Main:!
2010 INTEGER Agte507x,Ii,Nop
2020 INTEGER Respons,Stimulas
2030 INTEGER Port(1:2)
2040 REAL Stok(12,1:5000)
2050 REAL Stok2(12,1:5000)
2060 REAL Stok3(12,1:5000)
2070 DIM Ch$[10],Wk$[128]
2080 !
2090 ! PC's Monitor Clear
2100 CLEAR SCREEN
2110 GINIT
2130 !
2140 ! Set ENA++'s Addr
2150 Agte507x=717
2160 !
2170 Ch$="1"
2180 !
2190 ! Set ENA++'s I/O Path
2200 ASSIGN @Agte507x TO Agte507x
2210 !
2220 ON TIMEOUT SC(@Agte507x),15 RECOVER Tout
2230 !
```

```
2240 ! Set Start Port and End Port
2250 Port(1)=1
2260 Port(2)=2
2270 !
2280 ! Setup Segment Table
2290 CALL Set_sgm_tbl(@Agte507x)
2300 !
2310 ! Select Cal Kit
2320 CALL Select_cal_kit(@Agte507x,Ch$)
2330 !
2340 ! Execute Full-2Port Calibration
2350 CALL Cal_solt(@Agte507x,Ch$,2,Port(*))
2360 !
2370 ! Get All Segment's Points
2380 CALL Get_nop(@Agte507x,Nop,Ch$)
2390 !
2400 REDIM Stok(12,1:Nop*2)
2410 REDIM Stok2(12,1:Nop*2)
2420 REDIM Stok3(12,1:Nop*2)
2430 !
2440 CALL
Exec_error_term(@Agte507x,"READ","ES",Ch$,1,Nop,1,1,Stok(*))
2450 CALL
Exec_error_term(@Agte507x,"READ","ES",Ch$,2,Nop,2,2,Stok(*))
2460 CALL
```

```
Exec_error_term(@Agte507x,"READ","ER",Ch$,3,Nop,1,1,Stok(*))

2470 CALL
Exec_error_term(@Agte507x,"READ","ER",Ch$,4,Nop,2,2,Stok(*))

2480 CALL
Exec_error_term(@Agte507x,"READ","ED",Ch$,5,Nop,1,1,Stok(*))

2490 CALL
Exec_error_term(@Agte507x,"READ","ED",Ch$,6,Nop,2,2,Stok(*))

2500 !

2510 CALL
Exec_error_term(@Agte507x,"READ","EL",Ch$,7,Nop,1,2,Stok(*))

2520 CALL
Exec_error_term(@Agte507x,"READ","EL",Ch$,8,Nop,2,1,Stok(*))

2530 CALL
Exec_error_term(@Agte507x,"READ","ET",Ch$,9,Nop,1,2,Stok(*))

2540 CALL
Exec_error_term(@Agte507x,"READ","ET",Ch$,10,Nop,2,1,Stok(*))

2550 !

2560 CLEAR SCREEN

2570 PRINT "Push [Preset] - OK of ENA. Then push [Enter] key."

2580 INPUT "",Wk$

2590 !

2600 CALL Set_sgm_tbl(@Agte507x)

2610 !

2620 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COEF:METH:SOLT2
";Port(1);",";Port(2)

2630 !

2640 CALL
Exec_error_term(@Agte507x,"WRITE","ES",Ch$,1,Nop,1,1,Stok(*))
```

```
2650 CALL
Exec_error_term(@Agte507x,"WRITE","ES",Ch$,2,Nop,2,2,Stok(*))

2660 CALL
Exec_error_term(@Agte507x,"WRITE","ER",Ch$,3,Nop,1,1,Stok(*))

2670 CALL
Exec_error_term(@Agte507x,"WRITE","ER",Ch$,4,Nop,2,2,Stok(*))

2680 CALL
Exec_error_term(@Agte507x,"WRITE","ED",Ch$,5,Nop,1,1,Stok(*))

2690 CALL
Exec_error_term(@Agte507x,"WRITE","ED",Ch$,6,Nop,2,2,Stok(*))

2700 !

2710 CALL
Exec_error_term(@Agte507x,"WRITE","EL",Ch$,7,Nop,1,2,Stok(*))

2720 CALL
Exec_error_term(@Agte507x,"WRITE","EL",Ch$,8,Nop,2,1,Stok(*))

2730 CALL
Exec_error_term(@Agte507x,"WRITE","ET",Ch$,9,Nop,1,2,Stok(*))

2740 CALL
Exec_error_term(@Agte507x,"WRITE","ET",Ch$,10,Nop,2,1,Stok(*))

2750 !

2760 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COEF:SAVE"

2770 !

2780 CALL
Exec_error_term(@Agte507x,"READ","ES",Ch$,1,Nop,1,1,Stok2(*))

2790 CALL
Exec_error_term(@Agte507x,"READ","ES",Ch$,2,Nop,2,2,Stok2(*))

2800 CALL
Exec_error_term(@Agte507x,"READ","ER",Ch$,3,Nop,1,1,Stok2(*))

2810 CALL
```

```
Exec_error_term(@Agte507x,"READ","ER",Ch$,4,Nop,2,2,Stok2(*))
2820 CALL
Exec_error_term(@Agte507x,"READ","ED",Ch$,5,Nop,1,1,Stok2(*))
2830 CALL
Exec_error_term(@Agte507x,"READ","ED",Ch$,6,Nop,2,2,Stok2(*))
2840 !
2850 CALL
Exec_error_term(@Agte507x,"READ","EL",Ch$,7,Nop,1,2,Stok2(*))
2860 CALL
Exec_error_term(@Agte507x,"READ","EL",Ch$,8,Nop,2,1,Stok2(*))
2870 CALL
Exec_error_term(@Agte507x,"READ","ET",Ch$,9,Nop,1,2,Stok2(*))
2880 CALL
Exec_error_term(@Agte507x,"READ","ET",Ch$,10,Nop,2,1,Stok2(*))
2890 !
2900 ASSIGN @Agte507x TO *
2910 !
2920 DISP CHR$(139)&" Done ..."&CHR$(136)
2930 STOP
2940 !
2950 Tout: OFF TIMEOUT SC(@Agte507x)
2960 !
2970 ASSIGN @Agte507x TO *
2980 !
2990 PRINT CHR$(137)&" ENA Timeout ..."&CHR$(136)
3000 END
```

```
3010!
3020 Set_sgm_tbl: SUB Set_sgm_tbl(@Agte507x)
3030 REAL Star1(1:2),Stop1(1:2),Pow1(1:2)
3040 INTEGER Segm,Nop1(1:2),Num_of_tr1
3050 INTEGER I
3060 !
3070 CLEAR SCREEN
3080 DISP CHR$(138)&" Wait ..."&CHR$(136)
3090 !
3100 Segm=2 ! Number of Segment Ch.1 : 2
3110 Star1(1)=3.E+6 ! Start Frequency Ch.1 Segm.1: 3.0 MHz
3120 Star1(2)=5.0E+7 ! Segm.2: 50.0 MHz
3130 Stop1(1)=1.0E+7 ! Stop Frequency Ch.1 Segm.1: 10.0 MHz
3140 Stop1(2)=8.E+9 ! Segm.2: 8.0 GHz
3150 Nop1(1)=2 ! Number Ch.1 Segm.1: 2
3160 Nop1(2)=170 ! of Points Segm.2: 170
3170 If_bw1(1)=7.0E+4 ! IF Bandwidth Ch.1 Segm.1: 70 kHz
3180 If_bw1(2)=7.0E+4 ! Segm.2: 70 kHz
3190 Pow1(1)=0 ! Power Ch.1 Segm.1: 0 dBm
3200 Pow1(2)=0 ! Segm.2: 0 dBm
3210 !
3220 OUTPUT @Agte507x;":SYST:PRES"
3230 !
3240 WAIT 5
```

```
3250 !
3260 ! Channel 1
3270 !
3280 OUTPUT @Agte507x;":SENS1:SWE:TYPE SEGM"
3290 OUTPUT @Agte507x;":SENS1:SEGM:DATA 5,0,1,1,0,0,";Segm;",";
3300 FOR I=1 TO Segm-1
3310 OUTPUT
@Agte507x;Star1(I);",";Stop1(I);",";Nop1(I);",";If_bw1(I);",";Pow1(I);",";
3320 NEXT I
3330 OUTPUT
@Agte507x;Star1(Segm);",";Stop1(Segm);",";Nop1(Segm);",";If_bw1(Seg
3340 !
3350 OUTPUT @Agte507x;":CALC1:PAR:COUN ";Num_of_tr1
3360 FOR I=1 TO Num_of_tr1
3370 OUTPUT @Agte507x;":CALC1:PAR"&VAL$(I)&":SEL"
3380 NEXT I
3390 SUBEND
3400!
3410 Select_cal_kit: SUB Select_cal_kit(@Agte507x,Ch$)
3420 !==========================================
3430 ! Calibration Kit Selection Function
3440 !==========================================
3450 !
3460 DIM Cal_kit_lbl$(1:10)[20],Inp_char$[9]
3470 DIM Msg$[80],Wk$[10]
```

```
3480 INTEGER Cal_kit,I,Noc
3490 !
3500 ! PC's Monitor Clear
3510 CLEAR SCREEN
3520 !
3530 ! Number of Cal Kid
3540 Noc=10
3550 !
3560 FOR I=1 TO Noc
3570 OUTPUT @Agte507x;":SENS1:CORR:COLL:CKIT ";I
3580 OUTPUT @Agte507x;":SENS1:CORR:COLL:CKIT:LAB?"
3590 ENTER @Agte507x;Cal_kit_lbl$(I)
3600 NEXT I
3610 ON ERROR GOTO Kit_select
3620 !
3630 PRINT "## Calibration Kit Selection ##"
3640 FOR I=1 TO Noc
3650 PRINT USING "X,2D,A,X,20A";I,":",Cal_kit_lbl$(I)
3660 NEXT I
3670 PRINT ""
3680 PRINT "Input 1 to "&VAL$(Noc)
3690 !
3700 Msg$="Input number? (1 to "&VAL$(Noc)&") "
3710 LOOP
```

```
3720 LOOP
3730 DISP Msg$;
3740 INPUT Inp_char$
3750 Cal_kit=IVAL(Inp_char$,10)
3760 EXIT IF 1<=Cal_kit AND Cal_kit<=Noc
3770 Kit_select:!
3780 BEEP
3790 END LOOP
3800 !
3810 Wk$=""
3820 PRINT TABXY(1,Cal_kit+1);
3830 PRINT USING
"X,B,2D,A,X,20A,B";139,Cal_kit,":",Cal_kit_lbl$(Cal_kit),136
3840 INPUT "Sure ? [Y/N]",Wk$
3850 EXIT IF (UPC$(Wk$)="Y")
3860 PRINT TABXY(1,Cal_kit+1);
3870 PRINT USING "X,2D,A,X,20A";Cal_kit,":",Cal_kit_lbl$(Cal_kit)
3880 BEEP
3890 BEEP
3900 END LOOP
3910 OFF ERROR
3920 !
3930 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:CKIT ";Cal_kit
3940 SUBEND
3950!
```

```
3960 Cal_solt: SUB Cal_solt(@Agte507x,Ch$,INTEGER
Num_of_ports,INTEGER Port(*))
3970 !==========================================
3980 ! Full n Port Calibration Function
3990 !==========================================
4000 !
4010 DIM Buff$[9]
4020 INTEGER I,J
4030 !
4040 ! PC's Monitor Clear
4050 CLEAR SCREEN
4060 !
4070 PRINT "## Full "&VAL$(Num_of_ports)&" Port Calibration ##"
4080 !
4090 ! Calibration Type Selection
4100 !
4110 OUTPUT
@Agte507x;":SENS"&Ch$&":CORR:COLL:METH:SOLT"&VAL$(Num_of_
";
4120 FOR I=1 TO Num_of_ports-1
4130 OUTPUT @Agte507x;Port(I);",";
4140 NEXT I
4150 OUTPUT @Agte507x;Port(Num_of_ports)
4160 !
4170 ! Reflection Measurement
```

```
4180 !
4190 FOR I=1 TO Num_of_ports
4200 PRINT "Set OPEN to Port "&VAL$(Port(I))&". Then push [Enter] key."
4210 INPUT "",Buff$
4220 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:OPEN ";Port(I)
4230 OUTPUT @Agte507x;"*OPC?"
4240 ENTER @Agte507x;Buff$
4250 PRINT "Set SHORT to Port "&VAL$(Port(I))&". Then push [Enter] key."
4260 INPUT "",Buff$
4270 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:SHOR ";Port(I)
4280 OUTPUT @Agte507x;"*OPC?"
4290 ENTER @Agte507x;Buff$
4300 PRINT "Set LOAD to Port "&VAL$(Port(I))&". Then push [Enter] key."
4310 INPUT "",Buff$
4320 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:LOAD ";Port(I)
4330 OUTPUT @Agte507x;"*OPC?"
4340 ENTER @Agte507x;Buff$
4350 NEXT I
4360 !
4370 ! Transmission Measurement
4380 !
```

```
4390 FOR I=1 TO Num_of_ports-1

4400 FOR J=I+1 TO Num_of_ports

4410 PRINT "Set THRU between Port "&VAL$(Port(I))&" and Port
"&VAL$(Port(J))&". Then push [Enter] key."

4420 INPUT "",Buff$

4430 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:THRU
";Port(I);",";Port(J)

4440 OUTPUT @Agte507x;"*OPC?"

4450 ENTER @Agte507x;Buff$

4460 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:THRU
";Port(J);",";Port(I)

4470 OUTPUT @Agte507x;"*OPC?"

4480 ENTER @Agte507x;Buff$

4490 NEXT J

4500 NEXT I

4510 !

4520 ! Done

4530 !

4540 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COLL:SAVE"

4550 PRINT "Done"

4560 SUBEND

4570!

4580 Get_nop: SUB Get_nop(@Agte507x,INTEGER Nop,Ch$)

4590 ! Get All Segment's Points

4600 OUTPUT @Agte507x;":SENS"&Ch$&":SEGM:SWE:POIN?"
```

```
4610 ENTER @Agte507x;Nop

4620 SUBEND

4630 Exec_error_term: SUB
Exec_error_term(@Agte507x,Rw$,Id$,Ch$,INTEGER
Idx,Nop,Respons,Stimulas,REAL Stok(*))

4640 INTEGER Ii

4650 REAL Error_term_data(1:5000)

4660 !

4670 DISP CHR$(138)&" Wait ..."&CHR$(136)

4680 !

4690 REDIM Error_term_data(1:Nop*2)

4700 !

4710 SELECT Rw$

4720 CASE "WRITE"

4730 FOR Ii=1 TO Nop

4740 Error_term_data(2*Ii-1)=Stok(Idx,2*Ii-1)

4750 Error_term_data(2*Ii)=Stok(Idx,2*Ii)

4760 NEXT Ii

4770 !

4780 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COEF
"&Id$&",";Respons;",";Stimulas;",";Error_term_data(*)

4790 !

4800 CASE "READ"

4810 FOR Ii=1 TO Nop

4820 Error_term_data(2*Ii-1)=-999
```

```
4830 Error_term_data(2*Ii)=-999
4840 NEXT Ii
4850 !
4860 OUTPUT @Agte507x;":SENS"&Ch$&":CORR:COEF?
"&Id$&",";Respons;",";Stimulas
4870 ENTER @Agte507x;Error_term_data(*)
4880 !
4890 CALL Data_plot(Id$,Respons,Stimulas,Nop,Error_term_data(*))
4900 !
4910 FOR Ii=1 TO Nop
4920 Stok(Idx,2*Ii-1)=Error_term_data(2*Ii-1)
4930 Stok(Idx,2*Ii)=Error_term_data(2*Ii)
4940 NEXT Ii
4950 !
4960 END SELECT
4970 SUBEND
4980!
4990 Data_plot: SUB Data_plot(Error_term$,INTEGER
Respons,Stimulas,Nop,REAL Error_term_data(*))
5000 INTEGER Ii,Pen(1:2)
5010 REAL Y_minmax(1:2)
5020 DIM Wk$[20]
5030 !
5040 CLEAR SCREEN
5050 GINIT
```

```
5060 GCLEAR
5070 !
5080 Pen(1)=3
5090 Pen(2)=4
5100 !
5110 ! Get Min Value and Max Value from all data
5120 Y_minmax(1)=MIN(Error_term_data(*))
5130 Y_minmax(2)=MAX(Error_term_data(*))
5150 !
5160 IF (Y_minmax(1)=Y_minmax(2)) AND (Y_minmax(1)=0) THEN
5170 Y_minmax(1)=1
5180 Y_minmax(2)=-1
5190 ELSE
5200 IF (Y_minmax(1)=Y_minmax(2)) THEN
5210 Y_minmax(1)=Y_minmax(1)*.5
5220 Y_minmax(2)=Y_minmax(2)*1.5
5230 END IF
5240 END IF
5250 !
5260 VIEWPORT 25*RATIO,80*RATIO,40,90
5270 WINDOW 1,Nop,Y_minmax(1),Y_minmax(2)
5280 FRAME
5290 !
5300 VIEWPORT 80*RATIO,100*RATIO,40,90
```

```
5310 WINDOW 0,2,0,2
5320 PEN Pen(1)
5330 CSIZE 2.5
5340 LORG 2
5350 MOVE .2,1.5
5360 DRAW .4,1.5
5370 MOVE .5,1.5
5380 PEN 1
5390 LABEL ":Real Value"
5400 !
5410 PEN Pen(2)
5420 MOVE .2,1
5430 DRAW .4,1
5440 MOVE .5,1
5450 PEN 1
5460 LABEL ":Image Value"
5470 !
5480 VIEWPORT 25*RATIO,80*RATIO,90,100
5490 WINDOW 0,2,0,2
5500 CSIZE 3
5510 LORG 5
5520 MOVE 1,1.2
5530 LABEL "Error Term:"&Error_term$
5540 !
```

```
5550 MOVE 1,.5
5560 LABEL "Respons Port:"&VAL$(Respons)&" Stimulas
Port:"&VAL$(Stimulas)
5570 !
5580 VIEWPORT 0,25*RATIO,40,90
5590 WINDOW 0,2,0,2
5600 CLIP -10,10,-10,10
5610 LORG 8
5620 CSIZE 3
5630 !
5640 MOVE 1.9,0
5650 LABEL VAL$(Y_minmax(1))
5660 MOVE 1.9,2
5670 LABEL VAL$(Y_minmax(2))
5680 !
5690 VIEWPORT 25*RATIO,80*RATIO,30,40
5700 WINDOW 0,2,0,2
5710 CLIP -10,10,-10,10
5720 LORG 5
5730 MOVE 0,1.5
5740 LABEL VAL$(1)
5750 MOVE 2,1.5
5760 LABEL VAL$(Nop)
5770 !
5780 VIEWPORT 25*RATIO,80*RATIO,40,90
```

```
5790 WINDOW 1,Nop,Y_minmax(1),Y_minmax(2)
5800 FOR Ii=2 TO Nop
5820 PEN Pen(1)
5830 MOVE Ii-1,Error_term_data(2*(Ii-1)-1)
5840 DRAW Ii,Error_term_data(2*Ii-1)
5860 !
5870 PEN Pen(2)
5880 MOVE Ii-1,Error_term_data(2*(Ii-1))
5890 DRAW Ii,Error_term_data(2*Ii)
5900 NEXT Ii
5910 !
5920 PEN 1
5930 BEEP
5940 INPUT "Cont:push [Enter] key",Wk$
5950 SUBEND
5960!
```

OPC?

-
- Excel VBA
- HT Basic

:TRIG:SING

/9

:TRIG:SING *OPC

Excel VBA

```
Sub trg_sing_Click()
    Dim defrm As Long
    Dim vi As Long
    Dim ContMode(9) As String
    Dim Result As String * 10
    Dim i As Integer
    Const TimeOutTime = 100000 ' TimeOut time should be greater than the measurement time.
    '
    ' Assign a GPIB address to the I/O pass.
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    '
    ' Store the settings of continuous initiation mode for eachchannel
    ' (on for channels 1 and 2; off for channels 3 through 9)
    ' into the array variable ContMode().
    ContMode(1) = "ON"
    ContMode(2) = "ON"
    For i = 3 To 9
        ContMode(i) = "OFF"
    Next i
```

```
    '
    ' Turn on or off continuous initiation mode for each channel
    ' depending on the value of ContMode(*).
    For i = 1 To 9
        Call viVPrintf(vi, ":INIT" & CStr(i) & ":CONT " & ContMode(i) & vbLf,
0)
    Next i
    ' Set the trigger source to Bus Trigger.
    Call viVPrintf(vi, ":TRIG:SOUR BUS" & vbLf, 0)
    '
    ' Trigger the instrument to start a sweep cycle.
    Call viVPrintf(vi, ":TRIG:SING" & vbLf, 0)
    '
    ' Execute the *OPC? command and wait until the command
    ' returns 1 (i.e., the measurement cycle is completed).
    Call viVPrintf(vi, "*OPC?" & vbLf, 0)
    Call viVScanf(vi, "%t", Result)
    '
    ' Display a measurement completion message.
    Stat = MsgBox("Measurement complete", vbOKOnly)
    Call viClose(vi)
    Call viClose(defrm)
End Sub
```

HT Basictrg_sing.htb

```
10 DIM Cont_mode$(1:9)[9],Buff$[9]
20 INTEGER I
30 !
40 ASSIGN @Agte507x TO 717
50 !
60 Cont_mode$(1)="ON"
70 Cont_mode$(2)="ON"
80 Cont_mode$(3)="OFF"
90 Cont_mode$(4)="OFF"
100 Cont_mode$(5)="OFF"
110 Cont_mode$(6)="OFF"
120 Cont_mode$(7)="OFF"
130 Cont_mode$(8)="OFF"
140 Cont_mode$(9)="OFF"
150 !
160 FOR I=1 TO 9
170 OUTPUT @Agte507x;":INIT"&VAL$(I)&":CONT "&Cont_mode$(I)
180 NEXT I
190 OUTPUT @Agte507x;":TRIG:SOUR BUS"
200 !
210 OUTPUT @Agte507x;":TRIG:SING"
220 OUTPUT @Agte507x;"*OPC?"
230 ENTER @Agte507x;Buff$
```

```
240 !
250 PRINT "Measurement complete"
260 END
```

SQR

-
- Excel VBA
- HT Basic

SRQ

/9/9

SRQSQR

Excel VBA

```
Sub srq_meas_Click()
    Dim defrm As Long
    Dim vi As Long
    Dim ContMode(9) As String
    Dim Result As String * 10
    Dim i As Integer, StbStatus As Integer
    Const TimeOutTime = 100000 ' TimeOut time should be greater than the measurement time.
    '
    ' Assign a GPIB address to the I/O pass.
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    '
    ' Store the settings of continuous initiation mode for eachchannel
    ' (on for channels 1 and 2; off for channels 3 through 9)
    ' into the array variable ContMode().
    ContMode(1) = "ON"
    ContMode(2) = "ON"
    For i = 3 To 9
        ContMode(i) = "OFF"
    Next i
    '
```

```
    ' Turn on or off continuous initiation mode for each channel
    ' depending on the value of ContMode(*).
    For i = 1 To 9
        Call viVPrintf(vi, ":INIT" & CStr(i) & ":CONT " & ContMode(i) & vbLf,
0)
    Next i
    '
    ' Set the trigger source to Bus Trigger.
    Call viVPrintf(vi, ":TRIG:SOUR BUS" & vbLf, 0)
    '
    ' .
    Call viVPrintf(vi, ":STAT:OPER:PTR 0" & vbLf, 0) 'Set 0 at all bits of
Position Transition Filter
    Call viVPrintf(vi, ":STAT:OPER:NTR 16" & vbLf, 0) 'Set 1 at bit 4 of
Negative Transition Filter
    Call viVPrintf(vi, ":STAT:OPER:ENAB 16" & vbLf, 0) 'Set 1 at bit 4 of
Operation status enable
    Call viVPrintf(vi, "*SRE 128" & vbLf, 0) 'Set 1 at bit 7 of Service
Request Enable Register
    Call viVPrintf(vi, "*CLS" & vbLf, 0) ' Clear Register.
    '
    Call viVPrintf(vi, "*TRG" & vbLf, 0) 'Make a trigger
    ' Wait until Status Byte Register became 192
    Do
        Call viReadSTB(vi, StbStatus) ' Read Status Byte Register
        Range("B5").Value = StbStatus
```

```
    Loop Until StbStatus = 192
    '
    ' Display a measurement completion message.
    Stat = MsgBox("Measurement complete", vbOKOnly)
    ' Close IO
    Call viClose(vi)
    Call viClose(defrm)
End Sub
```

HT Basicsrq_meas.htb

```
10 DIM Cont_mode$(1:9)[9],Buff$[9]
20 INTEGER I
30 !
40 ASSIGN @Agte507x TO 717
50 !
60 Cont_mode$(1)="ON"
70 Cont_mode$(2)="ON"
80 Cont_mode$(3)="OFF"
90 Cont_mode$(4)="OFF"
100 Cont_mode$(5)="OFF"
110 Cont_mode$(6)="OFF"
120 Cont_mode$(7)="OFF"
130 Cont_mode$(8)="OFF"
140 Cont_mode$(9)="OFF"
150 !
160 FOR I=1 TO 9
170 OUTPUT @Agte507x;":INIT"&VAL$(I)&":CONT "&Cont_mode$(I)
180 NEXT I
190 OUTPUT @Agte507x;":TRIG:SOUR BUS"
200 !
210 OUTPUT @Agte507x;":STAT:OPER:PTR 0"
220 OUTPUT @Agte507x;":STAT:OPER:NTR 16"
230 OUTPUT @Agte507x;":STAT:OPER:ENAB 16"
```

```
240 OUTPUT @Agte507x;"*SRE 128"
250 OUTPUT @Agte507x;"*CLS"
260 OUTPUT @Agte507x;"*OPC?"
270 ENTER @Agte507x;Buff$
280 !
290 ON INTR 7 GOTO Meas_end
300 ENABLE INTR 7;2
310 OUTPUT @Agte507x;"*TRG"
320 PRINT "Waiting..."
330 Meas_wait: GOTO Meas_wait
340 Meas_end: OFF INTR 7
350 PRINT "Measurement Complete"
360 END
```

40

I/OGPIB

60140

1239Cont_mode$(*)

160180

Cont_mode$(*)

190

“”

210220

44101

230240

47

250270

290300

SRQSRQ

310320

350

SRQ

-
- HT Basic

SRQ

SRQ

Excel VBA

HT Basicsrq_err.bas

```
10 DIM Buff$[9],Err_mes$[50]
20 INTEGER Err_no
30 !
40 ASSIGN @Agte507x TO 717
50 !
60 OUTPUT @Agte507x;"*ESE 60"
70 OUTPUT @Agte507x;"*SRE 32"
80 OUTPUT @Agte507x;"*CLS"
90 OUTPUT @Agte507x;"*OPC?"
100 ENTER @Agte507x;Buff$
110 !
120 ON INTR 7 GOTO Err_proc
130 ENABLE INTR 7;2
140 OUTPUT @Agte507x;":CALC1:PAR:COUN 2"
150 PRINT "Trace 1 Meas.Para: S21"
160 PRINT "Trace 1 Format : Log Mag"
170 OUTPUT @Agte507x;":CALC1:PAR1:DEF S21"
180 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
190 OUTPUT @Agte507x;":CALC1:FORM MLOG"
200 PRINT "Trace 2 Meas.Para: S11"
210 PRINT "Trace 2 Format : Log Mag"
220 OUTPUT @Agte507x;":CALC1:PAR2:DEF S11"
230 OUTPUT @Agte507x;":CALC1:PAR2:SEL"
```

```
240 OUTPUT @Agte507x;":CALC1:FORM LOG"
250 OUTPUT @Agte507x;"*OPC?"
260 ENTER @Agte507x;Buff$
270 GOTO Skip_err_proc
280 Err_proc: OFF INTR 7
290 OUTPUT @Agte507x;";:SYST:ERR?"
300 ENTER @Agte507x;Err_no,Err_mes$
310 PRINT "Error occurred!!"
320 PRINT " No:";Err_no,"Description: "&Err_mes$
330 PRINT "PROGRAM INTERRUPT!!"
340 GOTO Prog_end
350 Skip_err_proc: PRINT "PROGRAM DONE."
360 Prog_end: END
```

40

I/OGPIB

6070

234551

80100

120130

SRQSRQ

14060

122

280330

290300:

310330

350

2

ASCII

-
- Excel VBA
- HT Basic

ASCII

111

Excel VBA

```
Sub read_asc_Click()
    '
    Dim defrm As Long
    Dim vi As Long
    Dim Result As String * 10000
    Dim Res As Variant
    Dim Res2 As Variant
    Dim Nop As Long
    Const TimeOutTime = 10000
    '
    ' Open the Analyzer
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    '
    ' Select Parameter 1
    Call viVPrintf(vi, ":CALC1:PAR1:SEL" + vbLf, 0)
    Call viVPrintf(vi, ":INIT1:CONT OFF" + vbLf, 0)
    Call viVPrintf(vi, ":ABOR" + vbLf, 0)
    '
    ' Read out NOP Data in ASCII transfer format
    Call viVPrintf(vi, ":SENS1:SWE:POIN?" + vbLf, 0)
    Call viVScanf(vi, "%t", Result)
```

```
Nop = Val(Result)
ReDim FMTData(Nop, 2)
ReDim Freq(Nop)
'
' Read out Measurement Data in ASCII transfer format
Call viVPrintf(vi, ":FORM:DATA ASC" + vbLf, 0)
Result = ""
Call viVPrintf(vi, ":CALC1:DATA:FDAT?" + vbLf, 0)
Call viVScanf(vi, "%t", Result)
Res = Split(Result, ",")
'
Range("A6:D1607").Clear 'Clear cells of Excel
'
' Write data in cells of Excel
j = 0
For i = 1 To Nop
    Cells(i + 5, 1) = i
    Cells(i + 5, 3) = Val(Res(j))
    Cells(i + 5, 4) = Val(Res(j + 1))
    j = j + 2
Next i
'
' Read out Measurement Frequency Data in ASCII transfer format
Result = ""
```

```
    Call viVPrintf(vi, ":SENS1:FREQ:DATA?" + vbLf, 0)
    Call viVScanf(vi, "%t", Result)
    Res2 = Split(Result, ",")
    '
    '  Write data in cells of Excel
    For i = 1 To Nop
        Cells(i + 5, 2) = Val(Res2(i - 1))
    Next i
    '
    ' Close the Analyzer
    Call viClose(vi)
    Call viClose(defrm)
End Sub
```

HT Basicread_asc.htb

```
10 REAL Fdata(1:1601,1:2),Freq(1:1601)
20 DIM Img$[30]
30 INTEGER Nop,I
40 !
50 ASSIGN @Agte507x TO 717
60 !
70 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
80 OUTPUT @Agte507x;":INIT1:CONT OFF"
90 OUTPUT @Agte507x;":ABOR"
100 OUTPUT @Agte507x;":SENS1:SWE:POIN?"
110 ENTER @Agte507x;Nop
120 REDIM Fdata(1:Nop,1:2),Freq(1:Nop)
130 !
140 ! Reading out in ASCII transfer format
150 !
160 OUTPUT @Agte507x;":FORM:DATA ASC"
170 !
180 OUTPUT @Agte507x;":CALC1:DATA:FDAT?"
190 ENTER @Agte507x;Fdata(*)
200 OUTPUT @Agte507x;":SENS1:FREQ:DATA?"
210 ENTER @Agte507x;Freq(*)
220 !
230 ! Displaying
```

```
240 !
250 OUTPUT @Agte507x;":CALC1:FORM?"
260 ENTER @Agte507x;Fmt$
270 SELECT Fmt$
280 CASE
"MLOG","PHAS","GDEL","MLIN","SWR","REAL","IMAG","UPH"
290 Img$="MD.4DE,2X,MD.6DE"
300 PRINT " Frequency Data"
310 FOR I=1 TO Nop
320 PRINT USING Img$;Freq(I),Fdata(I,1)
330 NEXT I
340 CASE ELSE
350 Img$="MD.4DE,2X,MD.6DE,2X,MD.6DE"
360 PRINT " Frequency Data1 Data2"
370 FOR I=1 TO Nop
380 PRINT USING Img$;Freq(I),Fdata(I,1),Fdata(I,2)
390 NEXT I
400 END SELECT
410 !
420 END
```

-
- Excel VBA
- HT Basic

1

Excel VBA

```
Sub read_bin_Click()
    Dim defrm As Long
    Dim vi As Long
    Dim Result As String * 10000
    Dim Res() As Double
    Dim Nop As Long
    Const TimeOutTime = 10000
    ' Open Analyzer
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    '
    Call viVPrintf(vi, ":CALC1:PAR1:SEL" + vbLf, 0)
    Call viVPrintf(vi, ":INIT1:CONT OFF" + vbLf, 0)
    Call viVPrintf(vi, ":ABOR" + vbLf, 0)
    '
    ' Reading out Measurement Frequency Data in Binary transfer format
    Call viVPrintf(vi, ":FORM:DATA REAL" + vbLf, 0)
    Call viVPrintf(vi, ":CALC1:DATA:FDAT?" + vbLf, 0)
    Call Scpi_read_binary_double_array(vi, Res, Nop)
    '
    ' Write data in cells of Excel
    Range("A6:D1607").Clear
```

```
    For i = 0 To Nop - 1
        j = i Mod 2
        k = i \ 2
        Cells(k + 6, j + 3).Value = Res(i)
    Next i
    '
    ' Read out Measurement Frequency Data in Binary transfer format
    Call viVPrintf(vi, ":SENS1:FREQ:DATA?" + vbLf, 0)
    Call Scpi_read_binary_double_array(vi, Res, Nop)
    '
    ' Write data in cells of Excel
    For i = 0 To Nop - 1
        Cells(i + 6, 1) = i + 1
        Cells(i + 6, 2).Value = Res(i)
    Next i
    ' Close
    Call viClose(vi)
    Call viClose(defrm)
End Sub
    '===================================
    ' BinaryAry Read Subroutine
    '===================================
Sub Scpi_read_binary_double_array(vi As Long, data() As Double, Nop
As Long)
    Dim dblArray(10000) As Double
```

```
    Dim paramsArray(3) As Long
    Dim err As Long
    Dim i As Long
    Dim lf_eoi As String * 1

    Nop = UBound(dblArray) - LBound(dblArray) + 1
    paramsArray(0) = VarPtr(Nop)
    paramsArray(1) = VarPtr(dblArray(0))
    err = viVScanf(vi, "%#Zb%1t", paramsArray(0))
    If err <> 0 Then MsgBox "Binary Error"

    ReDim data(Nop - 1)
    For i = 0 To Nop - 1
        data(i) = dblArray(i)
    Next
End Sub
```

HT Basicread_bin.htb

```
10 REAL Fdata(1:1601,1:2),Freq(1:1601)
20 DIM Buff$[9],Img$[30]
30 INTEGER Nop,I
40 !
50 ASSIGN @Agte507x TO 717
60 ASSIGN @Binary TO 717;FORMAT OFF
70 !
80 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
90 OUTPUT @Agte507x;":INIT1:CONT OFF"
100 OUTPUT @Agte507x;":ABOR"
110 OUTPUT @Agte507x;":SENS1:SWE:POIN?"
120 ENTER @Agte507x;Nop
130 REDIM Fdata(1:Nop,1:2),Freq(1:Nop)
140 !
150 ! Reading out in binary transfer format
160 !
170 OUTPUT @Agte507x;":FORM:DATA REAL"
180 !
190 OUTPUT @Agte507x;":CALC1:DATA:FDAT?"
200 ENTER @Agte507x USING "#,8A";Buff$
210 ENTER @Binary;Fdata(*)
220 ENTER @Agte507x USING "#,1A";Buff$
230 OUTPUT @Agte507x;":SENS1:FREQ:DATA?"
```

```
240 ENTER @Agte507x USING "#,8A";Buff$
250 ENTER @Binary;Freq(*)
260 ENTER @Agte507x USING "#,1A";Buff$
270 !
280 ! Displaying
290 !
300 OUTPUT @Agte507x;":CALC1:FORM?"
310 ENTER @Agte507x;Fmt$
320 SELECT Fmt$
330 CASE
"MLOG","PHAS","GDEL","MLIN","SWR","REAL","IMAG","UPH"
340 Img$="MD.4DE,2X,MD.6DE"
350 PRINT " Frequency Data"
360 FOR I=1 TO Nop
370 PRINT USING Img$;Freq(I),Fdata(I,1)
380 NEXT I
390 CASE ELSE
400 Img$="MD.4DE,2X,MD.6DE,2X,MD.6DE"
410 PRINT " Frequency Data1 Data2"
420 FOR I=1 TO Nop
430 PRINT USING Img$;Freq(I),Fdata(I,1),Fdata(I,2)
440 NEXT I
450 END SELECT
460 !
470 END
```

Ascii

-
- HT Basic

ASCII

HT Basicwrite_a.htb

```
10 REAL Freq,Fdata(1:1601,1:2)
20 DIM File$[300]
30 INTEGER Nop
40 !
50 ASSIGN @Agte507x TO 717
60 !
70 CALL Inp_file_name(File$)
80 !
90 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
100 OUTPUT @Agte507x;":INIT1:CONT OFF"
110 OUTPUT @Agte507x;":ABOR"
120 !
130 OUTPUT @Agte507x;":SENS1:SWE:POIN?"
140 ENTER @Agte507x;Nop
150 REDIM Fdata(1:Nop,1:2)
160 !
170 ON ERROR GOTO File_error
180 ASSIGN @File TO File$
190 ENTER @File USING "K";Buff$
200 ENTER @File USING "K";Buff$
210 ENTER @File USING "K";Buff$
220 FOR I=1 TO Nop
230 ENTER @File USING "19D,2X,19D,2X,19D";Freq,Fdata(I,1),Fdata
```

```
(I,2)
240 NEXT I
250 ASSIGN @File TO *
260 OFF ERROR
270 !
280 OUTPUT @Agte507x;":FORM:DATA ASC"
290 !
300 OUTPUT @Agte507x;":CALC1:DATA:FDAT ";Fdata(*)
310 !
320 GOTO Prog_end
330 !
340 File_error: OFF ERROR
350 PRINT "############# ERROR #############"
360 PRINT File$&" is NOT exist."
370 PRINT " or"
380 PRINT File$&" has UNSUITABLE data."
390 !
400 Prog_end: END
410 !==========================================
420 ! File Name Input Function
430 !==========================================
440 SUB Inp_file_name(Inp_name$)
450 DIM Inp_char$[9]
460 ON ERROR GOTO Inp_start
```

```
470 Inp_start: !
480 PRINT "Input File Name!"
490 INPUT "Name?",Inp_name$
500 PRINT "Input Name: "&Inp_name$
510 INPUT "OK? [Y/N]",Inp_char$
520 IF UPC$(Inp_char$)<>"Y" THEN Inp_start
530 OFF ERROR
540 SUBEND
```

50

I/OGPIB

70

Inp_file_nameFile$Inp_file_name

90110

11

130140

1Nop

150

1Nop.

170

File$

180260

File$F

280

ASCII

300

F11

340380

-
- HT Basic

HT Basicwrite_b.htb

```
10 REAL Freq,Fdata(1:1601,1:2)
20 DIM File$[300],Header$[10]
30 INTEGER Nop
40 !
50 ASSIGN @Agte507x TO 717
60 ASSIGN @Binary TO 717;FORMAT OFF
70 CALL Inp_file_name(File$)
80 !
90 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
100 OUTPUT @Agte507x;":INIT1:CONT OFF"
110 OUTPUT @Agte507x;":ABOR"
120 !
130 OUTPUT @Agte507x;":SENS1:SWE:POIN?"
140 ENTER @Agte507x;Nop
150 REDIM Fdata(1:Nop,1:2)
160 !
170 ON ERROR GOTO File_error
180 ASSIGN @File TO File$
190 ENTER @File USING "K";Buff$
200 ENTER @File USING "K";Buff$
210 ENTER @File USING "K";Buff$
220 FOR I=1 TO Nop
230 ENTER @File USING "19D,2X,19D,2X,19D";Freq,Fdata(I,1),Fdata
```

```
(I,2)
240 NEXT I
250 ASSIGN @File TO *
260 OFF ERROR
270 !
280 OUTPUT @Agte507x;":FORM:DATA REAL"
290 Header$="#6"&IVAL$(8*2*Nop,10)
300 OUTPUT @Agte507x;":CALC1:DATA:FDAT ";Header$;
310 OUTPUT @Binary;Fdata(*),END
320 GOTO Prog_end
330 !
340 File_error: OFF ERROR
350 PRINT "############# ERROR #############"
360 PRINT File$&" is NOT exist."
370 PRINT " or"
380 PRINT File$&" has UNSUITABLE data."
390 !
400 Prog_end: END
410 !==========================================
420 ! File Name Input Function
430 !==========================================
440 SUB Inp_file_name(Inp_name$)
450 DIM Inp_char$[9]
460 ON ERROR GOTO Inp_start
```

```
470 Inp_start: !
480 PRINT "Input File Name!"
490 INPUT "Name?",Inp_name$
500 PRINT "Input Name: "&Inp_name$
510 INPUT "OK? [Y/N]",Inp_char$
520 IF UPC$(Inp_char$)<>"Y" THEN Inp_start
530 OFF ERROR
540 SUBEND
```

5060

I/OGPIB

70

Inp_file_nameFile$

90110

11

130140

11

150

NopF

170

File$

180260

File$F

280

290

Header$

300

11Header$

310

F

I/O@

340380

440540Inp_file_name

460

480490

500510

/

520

870

-
- Excel VBA
- HT Basic

Excel VBA

```
Sub PeakSearch_Click()
    Dim defrm As Long
    Dim vi As Long
    Const TimeOutTime = 20000
    '
    Dim Buff As String, Img As String, Err_msg As String
    Dim Excursion As String, Freq As String * 20, Resp As Variant,
PeakPoint As Variant
    Dim Poin As String * 5, Result As String * 1000, errmsg As String * 20


    Excursion = "0.5"
    ' Open Analyzer
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    Call viVPrintf(vi, "*CLS" & vbLf, 0)
    '
    ' Setup Analyzer
    Call viVPrintf(vi, ":SENS1:FREQ:CENT 947.5E6" & vbLf, 0)
    Call viVPrintf(vi, ":SENS1:FREQ:SPAN 200E6" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:PAR1:DEF S11" & vbLf, 0)
    Call viVPrintf(vi, ":DISP:WIND1:TRAC1:Y:AUTO" & vbLf, 0)
    '
```

```
    Call viVPrintf(vi, ":CALC1:PAR1:SEL" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK1:FUNC:TYPE PEAK" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK1:FUNC:PEXC " & Excursion & vbLf,
0)
    Call viVPrintf(vi, ":CALC1:MARK1:FUNC:PPOL POS" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK1:FUNC:EXEC" & vbLf, 0)
    Call ErrorCheck(vi)
    Call viVPrintf(vi, ":CALC1:MARK1:X?" & vbLf, 0)
    Call viVScanf(vi, "%t", Freq)
    '
    Call viVPrintf(vi, ":CALC1:MARK1:Y?" & vbLf, 0)
    Call viVScanf(vi, "%t", Result)
    '
    Resp = Split(Result, ",")
    Cells(5, 5).Value = Val(Freq)
    Cells(5, 6).Value = Val(Resp(0))
    '
    Call viVPrintf(vi, ":CALC1:FUNC:DOM OFF" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:FUNC:TYPE APE" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:FUNC:PEXC " & Excursion & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:FUNC:PPOL NEG" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:FUNC:EXEC" & vbLf, 0)
    Call ErrorCheck(vi)
    '
    Call viVPrintf(vi, ":CALC1:FUNC:POIN?" & vbLf, 0)
```

```
    Call viVScanf(vi, "%t", Poin)
    Call viVPrintf(vi, ":CALC1:FUNC:DATA?" & vbLf, 0)
    Call viVScanf(vi, "%t", Result)
    PeakPoint = Split(Result, ",")
    '
    j = 1
        For i = 1 To Val(Poin) / 2
        Cells(6 + i, 5).Value = Val(PeakPoint(j))
        Cells(6 + i, 6).Value = Val(PeakPoint(j + 1))
        j = j + 2
    Next i
    '
    Call viClose(vi)
    Call viClose(defrm)
    '
End Sub
Sub ErrorCheck(vi)
    Dim err As String * 50, ErrNo As Variant, Response As
VbMsgBoxResult
    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)
    ErrNo = Split(err, ",")
    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)
    End If
End Sub
```

HT Basicsearch.htb

```
10 DIM Buff$[9],Img$[50],Err_msg$[100]
20 REAL Excursion,Freq,Resp,Result(1:100,1:2)
30 INTEGER Poin,Err_no
40 !
50 ASSIGN @Agte507x TO 717
60 Excursion=.5
70 !
80 OUTPUT @Agte507x;"*ESE 60"
90 OUTPUT @Agte507x;"*SRE 32"
100 OUTPUT @Agte507x;"*CLS"
110 OUTPUT @Agte507x;"*OPC?"
120 ENTER @Agte507x;Buff$
130 ON INTR 7 GOTO Err
140 ENABLE INTR 7;2
150 !
160 PRINT "Maximum Peak Search using Marker 1"
170 !
180 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
190 OUTPUT @Agte507x;":CALC1:MARK1:FUNC:TYPE PEAK"
200 OUTPUT @Agte507x;":CALC1:MARK1:FUNC:PEXC ";Excursion
210 OUTPUT @Agte507x;":CALC1:MARK1:FUNC:PPOL POS"
220 OUTPUT @Agte507x;":CALC1:MARK1:FUNC:EXEC"
230 OUTPUT @Agte507x;":CALC1:MARK1:X?"
```

```
240 ENTER @Agte507x;Freq
250 OUTPUT @Agte507x;":CALC1:MARK1:Y?"
260 ENTER @Agte507x;Resp
270 Img$="8A,MD.4DE,2X,MD.6DE"
280 PRINT " Frequency Response"
290 PRINT USING Img$;"Peak: ",Freq,Resp
300 !
310 PRINT "All Peaks Search using Command"
320 !
330 OUTPUT @Agte507x;":CALC1:FUNC:DOM OFF"
340 OUTPUT @Agte507x;":CALC1:FUNC:TYPE APE"
350 OUTPUT @Agte507x;":CALC1:FUNC:PEXC ";Excursion
360 OUTPUT @Agte507x;":CALC1:FUNC:PPOL POS"
370 OUTPUT @Agte507x;":CALC1:FUNC:EXEC"
380 OUTPUT @Agte507x;":CALC1:FUNC:POIN?"
390 ENTER @Agte507x;Poin
400 REDIM Result(1:Poin,1:2)
410 OUTPUT @Agte507x;":CALC1:FUNC:DATA?"
420 ENTER @Agte507x;Result(*)
430 Img$="4A,2D,2A,MD.4DE,2X,MD.6DE"
440 PRINT " Frequency Response"
450 FOR I=1 TO Poin
460 PRINT USING Img$;"Peak",I,": ",Result(I,2),Result(I,1)
470 NEXT I
```

```
480 GOTO No_err
490 Err: OFF INTR 7
500 OUTPUT @Agte507x;";:SYST:ERR?"
510 ENTER @Agte507x;Err_no,Err_msg$
520 PRINT "Error occurred!!"
530 PRINT " No:";Err_no,"Description: "&Err_msg$
540 No_err: OFF INTR 7
550 END
```

-
- HT Basic

1		847.5 MHz	905.0 MHz	-55.0 dBm	-55.0 dBm
2		935.0 MHz	960.0 MHz	-3.5 dBm	-3.5 dBm
3		935.0 MHz	960.0 MHz	0 dBm	0 dBm
4		980.0 MHz	1047.5 MHz	-25.0 dBm	-25.0 dBm

HT Basiclim_test.htb

```
10 DIM Param1$[9],Param2$[9],Fmt1$[9],Fmt2$[9],Buff$[9]
20 REAL Cent,Span,Lim1(1:4,1:5),Lim2(1:3,1:5),Fail_data(1:1601)
30 INTEGER Num_of_seg1,Num_of_seg2,Segment,Column,Fail_point
40 !
50 ASSIGN @Agte507x TO 717
60 !
70 Cent=9.475E+8
80 Span=2.00E+8
90 Param1$="S21"
100 Param2$="S11"
110 Fmt1$="MLOG"
120 Fmt2$="MLOG"
130 !
140 ! == Trace 1 Limit Line ==
150 Num_of_seg1=4 ! Number of segments: 4
160 ! -- Segment 1 --
170 Lim1(1,1)=1 ! Type : Maximum
180 Lim1(1,2)=8.475E+8 ! Frequency Start: 847.5 MHz
190 Lim1(1,3)=9.050E+8 ! Stop : 905.0 MHz
200 Lim1(1,4)=-55 ! Response Start: -55 dBm
210 Lim1(1,5)=-55 ! Stop : -55 dBm
220 ! -- Segment 2 --
230 Lim1(2,1)=2 ! Type : Minimum
```

```
240 Lim1(2,2)=9.350E+8 ! Frequency Start: 935.0 MHz
250 Lim1(2,3)=9.600E+8 ! Stop : 960.0 MHz
260 Lim1(2,4)=-3.5 ! Response Start: -3.5 dBm
270 Lim1(2,5)=-3.5 ! Stop : -3.5 dBm
280 ! -- Segment 3 --
290 Lim1(3,1)=1 ! Type : Maximum
300 Lim1(3,2)=9.350E+8 ! Frequency Start: 935.0 MHz
310 Lim1(3,3)=9.600E+8 ! Stop : 960.0 MHz
320 Lim1(3,4)=0 ! Response Start: 0 dBm
330 Lim1(3,5)=0 ! Stop : 0 dBm
340 ! -- Segment 4 --
350 Lim1(4,1)=1 ! Type : Maximum
360 Lim1(4,2)=9.800E+8 ! Frequency Start: 980.0 MHz
370 Lim1(4,3)=1.0475E+9 ! Stop : 1047.5 MHz
380 Lim1(4,4)=-25 ! Response Start: -25 dBm
390 Lim1(4,5)=-25 ! Stop : -25 dBm
400 ! == Trace 2 Limit Line ==
410 Num_of_seg2=3 ! Number of segments: 3
420 ! -- Segment 1 --
430 Lim2(1,1)=1 ! Type : Maximum
440 Lim2(1,2)=8.475E+8 ! Frequency Start: 847.5 MHz
450 Lim2(1,3)=9.250E+8 ! Stop : 925.0 MHz
460 Lim2(1,4)=0 ! Response Start: 0 dBm
470 Lim2(1,5)=0 ! Stop : 0 dBm
```

```
480 ! -- Segment 2 --
490 Lim2(2,1)=1 ! Type : Maximum
500 Lim2(2,2)=9.350E+8 ! Frequency Start: 935.0 MHz
510 Lim2(2,3)=9.600E+8 ! Stop : 960.0 MHz
520 Lim2(2,4)=-9.5 ! Response Start: -9.5 dBm
530 Lim2(2,5)=-9.5 ! Stop : -9.5 dBm
540 ! -- Segment 3 --
550 Lim2(3,1)=1 ! Type : Maximum
560 Lim2(3,2)=9.700E+8 ! Frequency Start: 970.0 MHz
570 Lim2(3,3)=1.0475E+9 ! Stop : 1047.5 MHz
580 Lim2(3,4)=0 ! Response Start: 0 dBm
590 Lim2(3,5)=0 ! Stop : 0 dBm
600 !
610 OUTPUT @Agte507x;":SENS1:FREQ:CENT ";Cent
620 OUTPUT @Agte507x;":SENS1:FREQ:SPAN ";Span
630 OUTPUT @Agte507x;":CALC1:PAR1:COUN 2"
640 OUTPUT @Agte507x;":DISP:WIND1:SPL D1_2"
650 OUTPUT @Agte507x;":TRIG:SOUR BUS"
660 OUTPUT @Agte507x;":INIT1:CONT ON"
670 !
680 ! Trace 1
690 !
700 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
710 !
```

```
720 OUTPUT @Agte507x;":CALC1:PAR1:DEF "&Param1$
730 OUTPUT @Agte507x;":CALC1:FORM "&Fmt1$
740 !
750 OUTPUT @Agte507x;":CALC1:LIM:DATA ";Num_of_seg1;
760 FOR Segment=1 TO Num_of_seg1
770 FOR Column=1 TO 5
780 OUTPUT @Agte507x;",";Lim1(Segment,Column);
790 NEXT Column
800 NEXT Segment
810 OUTPUT @Agte507x;""
820 OUTPUT @Agte507x;":CALC1:LIM:DISP ON"
830 OUTPUT @Agte507x;":CALC1:LIM ON"
840 !
850 ! Trace 2
860 !
870 OUTPUT @Agte507x;":CALC1:PAR2:SEL"
880 !
890 OUTPUT @Agte507x;":CALC1:PAR2:DEF "&Param2$
900 OUTPUT @Agte507x;":CALC1:FORM "&Fmt2$
910 !
920 OUTPUT @Agte507x;":CALC1:LIM:DATA ";Num_of_seg2;
930 FOR Segment=1 TO Num_of_seg2
940 FOR Column=1 TO 5
950 OUTPUT @Agte507x;",";Lim2(Segment,Column);
```

```
960 NEXT Column
970 NEXT Segment
980 OUTPUT @Agte507x;""
990 OUTPUT @Agte507x;":CALC1:LIM:DISP ON"
1000 OUTPUT @Agte507x;":CALC1:LIM ON"
1010 !
1020 ! Setting status registers
1030 !
1040 OUTPUT @Agte507x;":STAT:QUES:LIM:CHAN1:ENAB 6"
1050 OUTPUT @Agte507x;":STAT:QUES:LIM:CHAN1:PTR 6"
1060 OUTPUT @Agte507x;":STAT:QUES:LIM:CHAN1:NTR 0"
1070 OUTPUT @Agte507x;":STAT:QUES:LIM:PTR 2"
1080 OUTPUT @Agte507x;":STAT:QUES:LIM:NTR 0"
1090 OUTPUT @Agte507x;"*CLS"
1100 !
1110 OUTPUT @Agte507x;":TRIG:SING"
1120 OUTPUT @Agte507x;"*OPC?"
1130 ENTER @Agte507x;Buff$
1140 !
1150 ! Checking test results
1160 !
1170 OUTPUT @Agte507x;":STAT:QUES:LIM?"
1180 ENTER @Agte507x;Reg_val
1190 Ch1_judge=BIT(Reg_val,1)
```

```
1200 OUTPUT @Agte507x;":STAT:QUES:LIM:CHAN1?"
1210 ENTER @Agte507x;Reg_val
1220 Tr1_judge=BIT(Reg_val,1)
1230 Tr2_judge=BIT(Reg_val,2)
1240 !
1250 ! Displaying test results
1260 !
1270 IF Ch1_judge=0 THEN
1280 PRINT "## PASS! ##"
1290 ELSE
1300 PRINT "## FAIL! ##"
1310 IF Tr1_judge=0 THEN
1320 PRINT " Trace1(S21): PASS"
1330 ELSE
1340 PRINT " Trace1(S21): FAIL"
1350 !
1360 ! Reading and displaying frequency at failed points
1370 !
1380 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
1390 OUTPUT @Agte507x;":CALC1:LIM:REP:POIN?"
1400 ENTER @Agte507x;Fail_point
1410 REDIM Fail_data(1:Fail_point)
1420 OUTPUT @Agte507x;":CALC1:LIM:REP?"
1430 ENTER @Agte507x;Fail_data(*)
```

```
1440 PRINT " Frequency:"
1450 FOR I=1 TO Fail_point
1460 PRINT USING "3X,MD.4DE";Fail_data(I)
1470 NEXT I
1480 END IF
1490 IF Tr2_judge=0 THEN
1500 PRINT " Trace2(S11): PASS"
1510 ELSE
1520 PRINT " Trace2(S11): FAIL"
1530 !
1540 ! Reading and displaying frequency at failed points
1550 !
1560 OUTPUT @Agte507x;":CALC1:PAR2:SEL"
1570 OUTPUT @Agte507x;":CALC1:LIM:REP:POIN?"
1580 ENTER @Agte507x;Fail_point
1590 REDIM Fail_data(1:Fail_point)
1600 OUTPUT @Agte507x;":CALC1:LIM:REP?"
1610 ENTER @Agte507x;Fail_data(*)
1620 PRINT " Frequency:"
1630 FOR I=1 TO Fail_point
1640 PRINT USING "3X,MD.4DE";Fail_data(I)
1650 NEXT I
1660 END IF
1670 END IF
```

1680 END

50

I/OGPIB

70120

1212CentSpanParam1$Param2$Fmt1$Fmt2$

150

1Num_of_seg1

160390

1Lim1(*)

410

2Num_of_seg2

420590

2Lim2(*)

610620

CentSpan1

630660

112

700

11

720730

1Param1$Fmt1$

750810

1

750:Num_of_seg1

770790:

820830

1

870

12

890900

2Param2$Fmt2$

920980

2

9901000

2

10401060

160000000000000001100112“”1

1239“”12

10701080

11011

1090

1

11101130

11701190

1Ch1_judge

12001230

112Tr1_judgeTr2_judge

1280

1“”Ch1_judge0DUT

13001660

1“”Ch1_judge1

1300: “”

1320: 1“”Tr1_judge01

13401470:1“”Tr1_judge11“”Tr1_judge1

1340: 1“”

1380: 12

13901410:1

14201470: 1

1500: 2“”Tr2_judge02

15201650: 2“”Tr2_judge112

-
- Excel VBA
- HT Basic

Excel VBA

```
Sub Bandwid_click()
    Dim vi As Long
    Dim defrm As Long
    Dim Threshhold As Long
    Dim Result As String * 1000
    Dim Bdata As Variant
    Const TimeOutTime = 10000
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    Threshhold = -3
    Call viVPrintf(vi, ":SENS1:FREQ:CENT 947.5E6" & vbLf, 0)
    Call viVPrintf(vi, ":SENS1:FREQ:SPAN 200E6" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:PAR1:DEF S21" & vbLf, 0)
    Call viVPrintf(vi, ":DISP:WIND1:TRAC1:Y:AUTO" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:PAR1:SEL" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK1:FUNC:TYPE MAX" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK1:FUNC:EXEC" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK1:BWID:THR " + CStr(Threshhold) &
vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK1:BWID:DATA?" & vbLf, 0)
    Call viVScanf(vi, "%t", Result)
    Bdata = Split(Result, ",")
```

```
    Call ErrorCheck(vi)
    Cells(5, 2).Value = Val(Bdata(0))
    Cells(6, 2).Value = Val(Bdata(1))
    Cells(7, 2).Value = Val(Bdata(2))
    Cells(8, 2).Value = Val(Trim(Bdata(3)))
    Call viClose(vi)
    Call viClose(defrm)

End Sub
Sub ErrorCheck(vi)
    Dim err As String * 50, ErrNo As Variant, Response
    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)
    ErrNo = Split(err, ",")
    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)
    End If
End Sub
```

HT Basic.htb

```
10 DIM Buff$[9],Err_msg$[100]
20 REAL Threshold,Bwid,Cent,Q,Loss
30 INTEGER Err_no
40 !
50 ASSIGN @Agte507x TO 717
60 Threshold=-3
70 !
80 OUTPUT @Agte507x;"*ESE 60"
90 OUTPUT @Agte507x;"*SRE 32"
100 OUTPUT @Agte507x;"*CLS"
110 OUTPUT @Agte507x;"*OPC?"
120 ENTER @Agte507x;Buff$
130 ON INTR 7 GOTO Err
140 ENABLE INTR 7;2
150 !
160 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
170 OUTPUT @Agte507x;":CALC1:MARK1:FUNC:TYPE MAX"
180 OUTPUT @Agte507x;":CALC1:MARK1:FUNC:EXEC"
190 OUTPUT @Agte507x;":CALC1:MARK1:BWID:THR ";Threshold
200 OUTPUT @Agte507x;":CALC1:MARK1:BWID:DATA?"
210 WAIT .5
220 ENTER @Agte507x;Bwid,Cent,Q,Loss
230 !
```

```
240 PRINT "## Bandwidth Search ##"
250 PRINT "Bandwidth : ",Bwid
260 PRINT "Center Frequency: ",Cent
270 PRINT "Q : ",Q
280 PRINT "Loss : ",Loss
290 !
300 GOTO No_err
310 Err: OFF INTR 7
320 OUTPUT @Agte507x;";:SYST:ERR?"
330 ENTER @Agte507x;Err_no,Err_msg$
340 PRINT "Error occurred!!"
350 PRINT " No:";Err_no,"Description: "&Err_msg$
360 No_err: OFF INTR 7
370 END
```

-
- Excel VBA
- HT Basic

Excel VBA

```
Sub File_Save()
    ' Declare two string variables for file name and file type
    Dim File_Name As String
    Dim File_Type As String
    Dim defrm As Long
    Dim vi As Long
    Const TimeOutTime = 10000
    ' Check whether file name textbox is empty or not
    If TextBox1.Text <> "" Then
        File_Name = Trim(TextBox1.Text)
        File_Type = Trim(frmFileSave.ComboBox1.Value)
    ' Open connection to the E5071C
        Call viOpenDefaultRM(defrm)
        Call viOpen(defrm, "GPIB0::18::INSTR", 0, 0, vi)
        Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
        Select Case File_Type
            Case "1: State (State only)"
                Call viVPrintf(vi, ":MMEM:STOR:STYP STAT" + vbLf, 0)
                Call viVPrintf(vi, ":MMEM:STOR """ & File_Name & ".sta""" +
vbLf, 0)
            Case "2: State (State & Cal)"
                Call viVPrintf(vi, ":MMEM:STOR:STYP CST" + vbLf, 0)
                Call viVPrintf(vi, ":MMEM:STOR """ & File_Name & ".sta""" +
```

```
vbLf, 0)
            Case "3: State (State & Trace)"
                Call viVPrintf(vi, ":MMEM:STOR:STYP DST" + vbLf, 0)
                Call viVPrintf(vi, ":MMEM:STOR """ & File_Name & ".sta""" +
vbLf, 0)
            Case "4: State (State & Cal & Trace)"
                Call viVPrintf(vi, ":MMEM:STOR:STYP CDST" + vbLf, 0)
                Call viVPrintf(vi, ":MMEM:STOR """ & File_Name & ".sta""" +
vbLf, 0)
            Case "5: State (Trace Data (CSV))"
                Call viVPrintf(vi, ":MMEM:STOR:FDAT """ & File_Name &
".csv""" + vbLf, 0)
            Case "6: State (Screen)"
                Call viVPrintf(vi, ":MMEM:STOR:IMAG """ & File_Name &
".bmp""" + vbLf, 0)
            Case Else
                msgbox "Error in code"
        End Select
    Call viClose(defrm)
    Else
        msgbox "Please enter a filename"
    End If
End Sub
```

HT Basicfile_sav.htb

```
10 DIM File$[300],Inp_char$[30]
20 INTEGER Content
30 CLEAR SCREEN
40 ASSIGN @Agte507x TO 717
50 !
60 ON ERROR GOTO Content_select
70 Content_select: !
80 PRINT "## Save Content Selection ##"
90 PRINT "Select Content"
100 PRINT " 1: State (State only)"
110 PRINT " 2: State (State & Cal)"
120 PRINT " 3: State (State & Trace)"
130 PRINT " 4: State (State & Cal & Trace)"
140 PRINT " 5: Trace Data (CSV)"
150 PRINT " 6: Screen"
160 PRINT ""
170 PRINT "Input 1 to 6"
180 INPUT "Number?",Inp_char$
190 Content=IVAL(Inp_char$,10)
200 IF Content<1 OR Content>6 THEN Content_select
210 OFF ERROR
220 !
230 CALL Inp_file_name(File$)
```

```
240 !
250 SELECT Content
260 CASE 1
270 OUTPUT @Agte507x;":MMEM:STOR:STYP STAT"
280 OUTPUT @Agte507x;":MMEM:STOR """&File$&".sta"""
290 CASE 2
300 OUTPUT @Agte507x;":MMEM:STOR:STYP CST"
310 OUTPUT @Agte507x;":MMEM:STOR """&File$&".sta"""
320 CASE 3
330 OUTPUT @Agte507x;":MMEM:STOR:STYP DST"
340 OUTPUT @Agte507x;":MMEM:STOR """&File$&".sta"""
350 CASE 4
360 OUTPUT @Agte507x;":MMEM:STOR:STYP CDST"
370 OUTPUT @Agte507x;":MMEM:STOR """&File$&".sta"""
380 CASE 5
390 OUTPUT @Agte507x;":MMEM:STOR:FDAT """&File$&".csv"""
400 CASE 6
410 OUTPUT @Agte507x;":MMEM:STOR:IMAG """&File$&".bmp"""
420 END SELECT
430 !
440 END
450 !==========================================
460 ! File Name Input Function
470 !==========================================
```

```
480 SUB Inp_file_name(Inp_name$)
490 DIM Inp_char$[9]
500 ON ERROR GOTO Inp_start
510 Inp_start: !
520 PRINT "## File Name Input ##"
530 PRINT "Input Save File Name (without Extension)"
540 INPUT "Name?",Inp_name$
550 PRINT "Input Name: "&Inp_name$
560 INPUT "OK? [Y/N]",Inp_char$
570 IF UPC$(Inp_char$)<>"Y" THEN Inp_start
580 OFF ERROR
590 SUBEND
```

-
- HT Basic

E5071C

E5071CE5071C

HT Basicfile_xfr.htb

```
10 DIM Src_file$[50],Dst_file$[50],Src_size_char$[50],Inp_char$[30]
20 INTEGER Direction
30 ASSIGN @Agte507x TO 717
40 !
50 CLEAR SCREEN
60 ON ERROR GOTO Direct_select
70 Direct_select: !
80 PRINT "#### File Transfer ####"
90 PRINT " 1: E507x -> Controller"
100 PRINT " 2: Controller -> E507x"
110 PRINT ""
120 PRINT "Input 1 or 2"
130 INPUT "Number?",Inp_char$
140 Direction=IVAL(Inp_char$,10)
150 IF Direction<1 OR Direction>2 THEN Direct_select
160 OFF ERROR
170 !
180 PRINT ""
190 PRINT " Input source file name. ";
200 INPUT "Name?",Src_file$
210 PRINT ": "&Src_file$
220 !
230 IF Direction=2 THEN
```

```
240 PRINT " Input source file size. ";
250 INPUT "Size[Byte]?",Src_size_char$
260 PRINT ": "&Src_size_char$&"[Byte]"
270 END IF
280 !
290 PRINT " Input destination file name. ";
300 INPUT "Name?",Dst_file$
310 PRINT ": "&Dst_file$
320 PRINT ""
330 !
340 IF Direction=1 THEN
350 Copy_to_contr(@Agte507x,Src_file$,Dst_file$)
360 ELSE
370 Copy_to_e507x(@Agte507x,Src_file$,Src_size_char$,Dst_file$)
380 END IF
390 !
400 END
410 !==========================================
420 ! File Transfer Function (E507x -> Controller)
430 !==========================================
440 SUB Copy_to_contr(@Agte507x,Src_file$,Dst_file$)
450 DIM Img$[32],Src_size_char$[10],Buff$[9],Err_msg$[100]
460 INTEGER Max_bsize,Block_size,Err_no
470 REAL Src_size
```

```
480 !
490 ON ERROR GOTO Skip_purge
500 PURGE Dst_file$
510 Skip_purge: OFF ERROR
520 CREATE Dst_file$,1
530 ASSIGN @Dst_file TO Dst_file$
540 Max_bsize=24576 ! 24KByte
550 !
560 OUTPUT @Agte507x;"*ESE 60"
570 OUTPUT @Agte507x;"*SRE 32"
580 OUTPUT @Agte507x;"*CLS"
590 OUTPUT @Agte507x;"*OPC?"
600 ENTER @Agte507x;Buff$
610 !
620 ON INTR 7 GOTO Err
630 ENABLE INTR 7;2
640 PRINT "Now Copying: "&Src_file$&"(@E507x) -> "&Dst_file$&"
(@Contro ller)"
650 OUTPUT @Agte507x;":MMEM:TRAN? """&Src_file$&"""
660 WAIT .1
670 ENTER @Agte507x USING "#,A";Buff$
680 ENTER @Agte507x USING "#,A";Digit$
690 Img$="#,"&Digit$&"A"
700 ENTER @Agte507x USING Img$;Src_size_char$
710 !
```

```
720 Src_size=VAL(Src_size_char$)
730 WHILE Src_size>0
740 IF Src_size>Max_bsize THEN
750 Block_size=Max_bsize
760 ELSE
770 Block_size=Src_size
780 END IF
790 !
800 ALLOCATE Dat$[Block_size]
810 Img$="#,"&VAL$(Block_size)&"A"
820 ENTER @Agte507x USING Img$;Dat$
830 OUTPUT @Dst_file USING Img$;Dat$
840 DEALLOCATE Dat$
850 !
860 Src_size=Src_size-Block_size
870 END WHILE
880 !
890 PRINT "Done"
900 ENTER @Agte507x USING "#,A";Buff$
910 ASSIGN @Dst_file TO *
920 !
930 GOTO Skip_error
940 Err: OFF INTR 7
950 OUTPUT @Agte507x;";:SYST:ERR?"
```

```
960 ENTER @Agte507x;Err_no,Err_msg$
970 PRINT "Error occurred!!"
980 PRINT " No:";Err_no,"Description: "&Err_msg$
990 Skip_error: OFF INTR 7
1000 SUBEND
1010 !==============================================
1020 ! File Transfer Function (Controller -> E507x)
1030 !==============================================
1040 SUB
Copy_to_e507x(@Agte507x,Src_file$,Src_size_char$,Dst_file$)
1050 DIM Img$[32],Header$[10],Buff$[9],Err_msg$[100]
1060 INTEGER Max_bsize,Block_size,Err_no
1070 REAL Src_size
1080 !
1090 ON ERROR GOTO File_error
1100 ASSIGN @Src_file TO Src_file$
1110 OFF ERROR
1120 Max_bsize=24576 ! 24KByte
1130 !
1140 OUTPUT @Agte507x;"*CLS"
1150 OUTPUT @Agte507x;"*OPC?"
1160 ENTER @Agte507x;Buff$
1170 !
1180 PRINT "Now Copying: "&Src_file$&"(@Controller) -> "&Dst_file$&"
(@ E507x)"
```

```
1190 Header$="#"&VAL$(LEN(Src_size_char$))&Src_size_char$
1200 OUTPUT @Agte507x;":MMEM:TRAN """&Dst_file$&""","&Header$;
1210 !
1220 Src_size=VAL(Src_size_char$)
1230 WHILE Src_size>0
1240 IF Src_size>Max_bsize THEN
1250 Block_size=Max_bsize
1260 ELSE
1270 Block_size=Src_size
1280 END IF
1290 !
1300 ALLOCATE Dat$[Block_size]
1310 Img$="#,"&VAL$(Block_size)&"A"
1320 ENTER @Src_file USING Img$;Dat$
1330 OUTPUT @Agte507x USING Img$;Dat$
1340 DEALLOCATE Dat$
1350 !
1360 Src_size=Src_size-Block_size
1370 END WHILE
1380 !
1390 OUTPUT @Agte507x;"",END
1400 ASSIGN @Src_file TO *
1410 !
1420 OUTPUT @Agte507x;";:SYST:ERR?"
```

```
1430 ENTER @Agte507x;Err_no,Err_msg$
1440 IF Err_no=0 THEN
1450 PRINT "Done"
1460 ELSE
1470 PRINT "Error occurred!!"
1480 PRINT " No:";Err_no,"Description: "&Err_msg$
1490 END IF
1500 GOTO Skip_error
1510 File_error:OFF ERROR
1520 PRINT "File name NOT found!"
1530 Skip_error:!
1540 SUBEND
```

40

I/OGPIB

60130

80130

140150

180210

Src_fileSrc_file

180210

Src_fileSrc_file

230270

2E5071CSrc_size_charSrc_size_char

290320

Src_fileSrc_file

350

1E5071CCopy_to_contrE5071CSrc_fileDst_file

370

2Copy_to_e507xSrc_file$E5071CDst_file$

Copy_to_contr4401000E5071C

490520

File$File$

530

I/O

540

Max_bsize24 KHTBasic

560600

SRQ

620630

SRQSRQ

640650

E5071C

670680

#Digit$

690

Digit$

700

Src_size_char$

720

Src_size_char$

730870

Src_size0

740780: Src_sizeMax_bsizeMax_bsizeBlock_size
Src_sizeMax_bsizeSrc_sizeBlock_size

800Block_sizeDat$

810 Block_sizeDat$

820 E5071CDat$

830 Dat$

840860 Dat$Src_sizeBlock_size

890900

940980

Copy_to_e507x10401540E5071C

10901110

I/O

1120

24 KMax_bsize

11401160

1180

11901200

Src_size_char$E5071C

1220

Src_size_char$Src_size

12301370

Src_size0

12401280: Src_sizeMax_bsizeMax_bsiizeBlock_size
Src_sizeMax_bsizeSrc_sizeBlock_size

1300 Block_sizeDat$

1310 Block_size

1320 Dat$

1330 Dat$E5071C

13401360 Dat$Src_sizeBlock_size

1390

14201430

Err_noErr_msg$

14401490

Err_no0Err_no0Err_noErr_msg$

1510520

-
- Excel VBA
- HT Basic

3

Excel VBA

```
Sub Fixture_Click()
    Dim defrm As Long          'Session to Default Resource Manager
    Dim vi As Long             'Session to instrument
    Dim Para As String
    Dim Dev_Type As String
    Dim Dmc_c As Double
    Dim Dmc_g As Double
    Dim Dmc_l As Double
    Dim Dmc_r As Double
    Dim Z0_se As Double
    Dim Z0_diff As Double
    Dim Port(3) As Integer
    Dim Result As String

    Const TimeOutTime = 20000

    Dev_Type = "SBAL"                'Divice Type     : SE-Bal
    Port(1) = 1                      'Port1(SE)       : 1
    Port(2) = 2                      'Port2(Bal)      : 2,3
    Port(3) = 3                      '
    Para = "SDS21"                   'Meas. Parameter : Sds21
    Dmc_c = 0.000000000001           'Diff.        C : 1 pF
    Dmc_g = 0.002                    'Matching      G : 2 mS
```

```
Dmc_l = 0.00000003              'Circuit       L : 30 nH
Dmc_r = 0.004                   '              R : 4 mohm
Z0_se = 100                     'Z Conversion Z0 : 100 ohm
Z0_diff = 210                   'Diff. Z Conv.Z0 : 210 ohm

Call viOpenDefaultRM(defrm)         'Initializes the VISA system.
Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)        'Opens the
session to the specified instrument.
Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)       'The
state of an attribute for the specified session.

Call viVPrintf(vi, "*RST" & vbLf, 0)    'Presets the setting state of the
ENA.
Call viVPrintf(vi, "*CLS" & vbLf, 0)    'Clears the all status register.

Call viVPrintf(vi, ":CALC1:FSIM:BAL:DEV " & Dev_Type & vbLf, 0)
 'Sets the balance device type.
Call viVPrintf(vi, ":CALC1:FSIM:BAL:TOP:" & Dev_Type & " " & Port(1)
& "," & Port(2) & "," & Port(3) & vbLf, 0)  'Sets the port assignment.
Call viVPrintf(vi, ":CALC1:FSIM:BAL:PAR1:" & Dev_Type & " " & Para
& vbLf, 0)   'Sets the measurement parameter of port 1.
Call viVPrintf(vi, ":CALC1:FSIM:BAL:PAR1:STAT ON" & vbLf, 0)
 'Turns on balance-unbalance conversion.
Call ErrorCheck(vi)   'Checking the error.

Call viVPrintf(vi, ":CALC1:FSIM:BAL:DMC:BPOR1 PLPC" & vbLf, 0)
 'Specifies the type of differential matching circuit.
```

```
Call viVPrintf(vi, ":CALC1:FSIM:BAL:DMC:BPOR1:PAR:C " & Dmc_c & vbLf, 0)  'Sets the differential matching circuit constants C.

Call viVPrintf(vi, ":CALC1:FSIM:BAL:DMC:BPOR1:PAR:G " & Dmc_g & vbLf, 0)  'Sets the differential matching circuit constants G.

Call viVPrintf(vi, ":CALC1:FSIM:BAL:DMC:BPOR1:PAR:L " & Dmc_l & vbLf, 0)  'Sets the differential matching circuit constants L.

Call viVPrintf(vi, ":CALC1:FSIM:BAL:DMC:BPOR1:PAR:R " & Dmc_r & vbLf, 0)   'Sets the differential matching circuit constants R.

Call viVPrintf(vi, ":CALC1:FSIM:BAL:DMC:STAT ON" & vbLf, 0)  'Turns of differential matching circuit embedding.

Call ErrorCheck(vi)   'Checking the error.


Call viVPrintf(vi, ":CALC1:FSIM:SEND:ZCON:PORT2:Z0 " & Z0_se & vbLf, 0)   'Sets the port impedance of port 2.

Call viVPrintf(vi, ":CALC1:FSIM:SEND:ZCON:PORT3:Z0 " & Z0_se & vbLf, 0)   'Sets the port impedance of port 3.

Call viVPrintf(vi, ":CALC1:FSIM:SEND:ZCON:STAT ON" & vbLf, 0)
 'Turns on port impedance conversion.

Call ErrorCheck(vi)    'Checking the error.


Call viVPrintf(vi, ":CALC1:FSIM:BAL:DZC:BPOR1:Z0 " & Z0_diff & vbLf, 0)  'Sets the differential port impedance of balance port 1.

Call viVPrintf(vi, ":CALC1:FSIM:BAL:DZC:STAT ON" & vbLf, 0)   'Turns on differential port impedance conversion.


Call viVPrintf(vi, ":CALC1:FSIM:STAT ON" & vbLf, 0)  'Turns on the fixture simulator.

Call ErrorCheck(vi)   'Checking the error.
```

```
    Call viClose(vi)   'Closes the resource manager session.
    Call viClose(defrm)   'Breaks the communication and terminates the
VISA system.

    End
End Sub
Sub ErrorCheck(vi As Long)
    Dim err As String * 50, ErrNo As Variant, Response

    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err) 'Reads error
message.
    ErrNo = Split(err, ",")  'Gets the error code.

    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)  'Display the
message box.
    End If
End Sub
```

HT Basicfixture.htb

```
10 DIM Dev_type$[9],Para$[9]
20 REAL Dmc_c,Dmc_g,Dmc_l,Dmc_r,Z0_se,Z0_diff
30 INTEGER Port(1:3)
40 !
50 ASSIGN @Agte507x TO 717
60 !
70 Dev_type$="SBAL" ! Device Type : SE-Bal
80 Port(1)=1 ! Port1(SE) : 1
90 Port(2)=2 ! Port2(Bal) : 2,3
100 Port(3)=3 !
110 Para$="SDS21" ! Meas. Parameter : Sds21
120 Dmc_c=1.E-12 ! Diff. C: 1 pF
130 Dmc_g=2.E-3 ! Matching G: 2 mS
140 Dmc_l=3.E-8 ! Circuit L: 30 nH
150 Dmc_r=4.E-3 ! R: 4 mohm
160 Z0_se=100 ! Z Conversion Z0: 100 ohm
170 Z0_diff=210 ! Diff. Z Conv. Z0: 210 ohm
180 !
190 ! Balance-Unbalance Conversion Setting
200 !
210 OUTPUT @Agte507x;":CALC1:FSIM:BAL:DEV "&Dev_type$
220 OUTPUT @Agte507x;":CALC1:FSIM:BAL:TOP:"&Dev_type$&"
";Port(*)
```

```
230 OUTPUT @Agte507x;":CALC1:FSIM:BAL:PAR1:"&Dev_type$&"
"&Para$

240 OUTPUT @Agte507x;":CALC1:FSIM:BAL:PAR1:STAT ON"

250 !

260 ! Diff. Matching Circuit Setting

270 !

280 OUTPUT @Agte507x;":CALC1:FSIM:BAL:DMC:BPOR1 PLPC"

290 OUTPUT @Agte507x;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:C
";Dmc_c

300 OUTPUT @Agte507x;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:G
";Dmc_g

310 OUTPUT @Agte507x;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:L
";Dmc_l

320 OUTPUT @Agte507x;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:R
";Dmc_r

330 OUTPUT @Agte507x;":CALC1:FSIM:BAL:DMC:STAT ON"

340 !

350 ! Z Conversion Setting

360 !

370 OUTPUT @Agte507x;":CALC1:FSIM:SEND:ZCON:PORT2:Z0
";Z0_se

380 OUTPUT @Agte507x;":CALC1:FSIM:SEND:ZCON:PORT3:Z0
";Z0_se

390 OUTPUT @Agte507x;":CALC1:FSIM:SEND:ZCON:STAT ON"

400 !

410 ! Diff. Z Conversion Setting

420 !
```

```
430 OUTPUT @Agte507x;":CALC1:FSIM:BAL:DZC:BPOR1:Z0 ";Z0_diff
440 OUTPUT @Agte507x;":CALC1:FSIM:BAL:DZC:STAT ON"
450 !
460 ! Fixture Simulator On/Off
470 !
480 OUTPUT @Agte507x;":CALC1:FSIM:STAT ON"
490 !
500 END
```

-
- Excel VBA
- HT Basic

Excel VBA

```
Sub Time_Domain_Click()
    Dim defrm As Long                    'Session to Default Resource
Manager
    Dim vi As Long                       'Session to instrument
    Dim Para As String
    Dim Tran_Type As String
    Dim Stim_Type As String
    Dim stop_freq As Double
    Dim Win_Beta As Double
    Dim Star_Time As Double
    Dim Stop_Time As Double
    Dim Result As String

    Const TimeOutTime = 40000            'timeout time.

    stop_freq = 3000000000#              'Stop Frequendy  : 3 GHz
    Nop = 201                            'Nop             : 201
    Para = "S11"                         'Meas. Parameter : S11
    Tran_Type = "LPAS"                   'Transform Type  : Lowpass
    Stim_Type = "IMP"                    'Stimulus Type   : Impulse
    Win_Beta = 13                        'Window Beta     : 13 (Maximum Type)
    Star_Time = 0                        'Start time      : 0 sec
    Stop_Time = 0.00000001               'Stop time       : 10 nsec
```

```
Call viOpenDefaultRM(defrm)
                                        'Initializes the VISA system.
Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
                          'Opens the session to the specified instrument.
Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
                    'The state of an attribute for the specified session.


Call viVPrintf(vi, ":SYST:PRES" & vbLf, 0)
                              'Presets the setting state of the ENA.
Call viVPrintf(vi, "*CLS" & vbLf, 0)                                'Clears
the all status register.


Call viVPrintf(vi, ":SENS1:FREQ:STOP " & stop_freq & vbLf, 0)
              'Sets the sweep stop frequency.
Call viVPrintf(vi, ":SENS1:SWE:POIN " & Nop & vbLf, 0)
                    'Sets the number of points.


Call viVPrintf(vi, ":CALC1:TRAN:TIME:LPFR" & vbLf, 0)
                     'Sets a measurement point.


Call viVPrintf(vi, ":CALC1:PAR1:DEF " & Para & vbLf, 0)
                    'Sets the measurement parameter.
Call viVPrintf(vi, ":TRIG:SOUR BUS" & vbLf, 0)
                          'Sets the trigger source to BUS.
Call ErrorCheck(vi)                                            'Checking the
error.
```

```
    MsgBox "Connect Port1 to Ecal Module. Then click OK button.",
vbOKOnly          'Display the message box.


    Call viVPrintf(vi, ":SENS1:CORR:COLL:ECAL:SOLT1 1" & vbLf, 0)
              'Execute the 1-port calibration.

    Call viVPrintf(vi, "*OPC?" & vbLf, 0)                                  'Sets
the *OPC? command.

    Call viVScanf(vi, "%t", Result)                                       'Reads
the *OPC? result.

    Call ErrorCheck(vi)                                                 'Checking the
error.

    MsgBox "Set DUT. Then click OK button.", vbOKOnly
                          'Display the message box.


    Call viVPrintf(vi, ":TRIG:SING" & vbLf, 0)
                              'Execute the trigger.

    Call viVPrintf(vi, "*OPC?" & vbLf, 0)                                  'Sets
the *OPC? command.

    Call viVScanf(vi, "%t", Result)                                       'Reads
the *OPC? result.

    Call viVPrintf(vi, ":DISP:WIND1:TRAC1:Y:AUTO" & vbLf, 0)
                  'Execute auto scale.


    MsgBox "Click OK button. < Time Domain Transform >", vbOKOnly
              'Display the message box.


    Call viVPrintf(vi, ":CALC1:TRAN:TIME " & Tran_Type & vbLf, 0)
              'Sets the transformation type.

    Call viVPrintf(vi, ":CALC1:TRAN:TIME:STIM " & Stim_Type & vbLf, 0)
```

```
        'Sets the stimulus type.
   Call viVPrintf(vi, ":CALC1:TRAN:TIME:KBES " & Win_Beta & vbLf, 0)
         'Sets the beta value of the window.
   Call viVPrintf(vi, ":CALC1:TRAN:TIME:STAR " & Star_Time & vbLf, 0)
        'Sets the start value of the display range.
   Call viVPrintf(vi, ":CALC1:TRAN:TIME:STOP " & Stop_Time & vbLf, 0)
        'Sets the end value of the display range.
   Call viVPrintf(vi, ":CALC1:TRAN:TIME:STAT ON" & vbLf, 0)
                'Turns on the time domain function.
   Call ErrorCheck(vi)                                        'Checking the
error.


   Call viVPrintf(vi, ":CALC1:PAR1:SEL" & vbLf, 0)
                         'Sets the active trace.
   Call viVPrintf(vi, ":CALC1:FORM REAL" & vbLf, 0)
                        'Sets the real data format.
   Call viVPrintf(vi, ":DISP:WIND1:TRAC1:Y:AUTO" & vbLf, 0)
                'Execute auto scale.
   Call ErrorCheck(vi)                                        'Checking the
error.


   Call viClose(vi)                                          'Closes the
resource manager session.
   Call viClose(defrm)                                        'Breaks the
communication and terminates the VISA system.


   End                                                     'End
End Sub
```

HT Basic

```
10 DIM Para$[9],Tran_type$[9],Stim_type$[9],Buff$[9],Inp_ch ar$[9]
20 REAL Stop_freq,Win_beta,Star_time,Stop_time
30 INTEGER Nop
40 !
50 ASSIGN @Agte507x TO 717
60 !
70 Stop_freq=3.E+9 ! Stop Frequency : 3 GHz
80 Nop=201 ! Nop : 201
90 Para$="S11" ! Meas. Parameter : S11
100 !
110 Tran_type$="LPAS" ! Transform Type : Lowpass
120 Stim_type$="IMP" ! Stimulus Type : Impulse
130 Win_beta=13 ! Window Beta : 13 (Maximum Type)
140 Star_time=0 ! Start time : 0 s
150 Stop_time=1.E-8 ! Stop time : 10 ns
160 !
170 OUTPUT @Agte507x;":SYST:PRES"
180 OUTPUT @Agte507x;":SENS1:FREQ:STOP ";Stop_freq
190 OUTPUT @Agte507x;":SENS1:SWE:POIN ";Nop
200 !
210 OUTPUT @Agte507x;":CALC1:TRAN:TIME:LPFR"
220 !
230 OUTPUT @Agte507x;":CALC1:PAR1:DEF "&Para$
```

```
240 OUTPUT @Agte507x;":TRIG:SOUR BUS"
250 !
260 ! 1 Port Full Calibration (ECal)
270 !
280 PRINT "Connect Port 1 to ECal Module. Then push [Enter] key."
290 INPUT "",Buff$
300 OUTPUT @Agte507x;":SENS1:CORR:COLL:ECAL:SOLT1 1"
310 OUTPUT @Agte507x;":SYST:ERR?"
320 ENTER @Agte507x;Buff$
330 !
340 ! Measurement
350 !
360 PRINT "Set DUT. Then Push [Enter] key."
370 INPUT "",Inp_char$
380 !
390 OUTPUT @Agte507x;":TRIG:SING"
400 OUTPUT @Agte507x;"*OPC?"
410 ENTER @Agte507x;Buff$
420 !
430 OUTPUT @Agte507x;":DISP:WIND1:TRAC1:Y:AUTO"
440 PRINT "Push [Enter] key. -> [Time Domain Transform]"
450 INPUT "",Inp_char$
460 !
470 ! Time Domain Transform
```

```
480 !
490 OUTPUT @Agte507x;":CALC1:TRAN:TIME "&Tran_type$
500 OUTPUT @Agte507x;":CALC1:TRAN:TIME:STIM "&Stim_type$
510 OUTPUT @Agte507x;":CALC1:TRAN:TIME:KBES ";Win_beta
520 OUTPUT @Agte507x;":CALC1:TRAN:TIME:STAR ";Star_time
530 OUTPUT @Agte507x;":CALC1:TRAN:TIME:STOP ";Stop_time
540 OUTPUT @Agte507x;":CALC1:TRAN:TIME:STAT ON"
550 !
560 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
570 OUTPUT @Agte507x;":CALC1:FORM REAL"
580 OUTPUT @Agte507x;":DISP:WIND1:TRAC1:Y:AUTO"
590 END
```

SICL-LAN

-
- Excel VBA

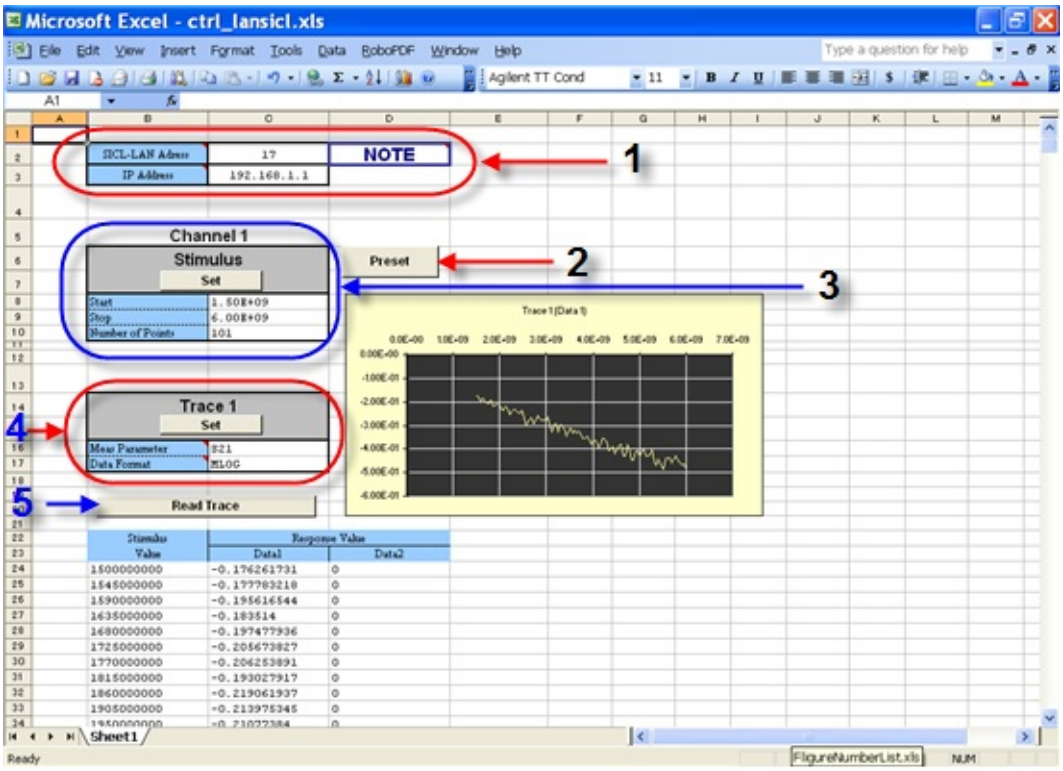
WindowsSICLE5071C

- SICL-LANE5071C SICL-LAN

Excel VBA

Microsoft Excel**ctrl_lansicl.xls**

ctrl_lansicl.xls



e5071c191

ctrl_lansicl.xls

1. 1SICL-LANSICL-LANE5071CXX **System** > **Misc Setup** > **Network Setup** > **SICL-LAN Address** [**XX**]IPE5071CIP VBA
2. 2“”
3. 31
4. 411
5. 5 11

VBA

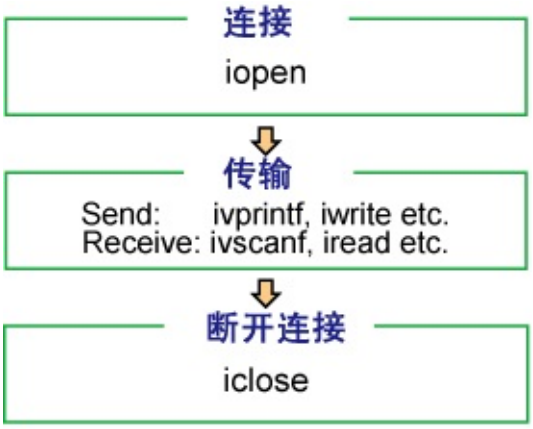
VBASICL

- VBASICLSICLVBSICL
- VBA“SICL”ctrl_sicllan.xls

SICLSICL

- SICL **ivprintf** **ivscanfiread**SICLsicl.hlpSICL

SICL



e5071c441

- SICLSICL

SICL

OpenSession OpenSession1 ctrl_lansicl.xlsSICL-LANIPSICL
iopenOpenSessionE5071C **iopen**E5071C

= iopen(*dev*)

	addr
	dev
	sicl-name [ip-address]:interface, sicl-lan-address

dev“lan[192.168.0.1]:hpib9,17,”SICL **lan** **hpib917**IP **192.168.0.1**E5071C

OpenSession

```
Function OpenSession() As Integer
Dim ServAddr As String
Dim IpAddr As String
On Error GoTo ErrHandler
'''Get Sicl-Lan Address
Sheets("Sheet1").Select
Range("C2").Select
ServAddr = ActiveCell.FormulaR1C1
'''Get Ip Address
Sheets("Sheet1").Select
Range("C3").Select
IpAddr = ActiveCell.FormulaR1C1
```

```
OpenSession = iopen("lan[" & IpAddr & "]:hpib9," & ServAddr)
Call itimeout(OpenSession, 10000)
Exit Function
ErrHandler:
MsgBox "*** Error : " & Error$
Call siclcleanup
End
End Function
```

OutputSiclLanOutputSiclLanSICL**ivprintf**SCPI **ivprintfiopen**

= ivprintf(*addr,mes*)

	Status
	addr
	mes

OutputSiclLan

```
Sub OutputSiclLan(addr As Integer, message As String)

Dim Status As Integer
Dim actualcnt As Long
Dim length As Long

On Error GoTo ErrHandler
length = Len(message)
Status = ivprintf(addr, message & Chr$(10))
Exit Sub

ErrHandler:
MsgBox "*** Error : " & Error$
Call siclcleanup
End
End Sub
```

ASCII EnterSiclLanEnterSiclLanSICL**ivscanf**ASCII **ivscanf** **iopen**

= ivscanf(*addr,fmt,ap*)

	fmt
	ap

Visual Basic **ivscanf**

EnterSiclLan

```
Sub EnterSiclLan(addr As Integer, Query As String)

Dim Status As Integer
Dim actualcnt As Long
Dim res As String * 256

On Error GoTo ErrHandler
Status = ivscanf(addr, "%t", res)
Query = Trim(res)
Exit Sub
```

```
ErrHandler:
MsgBox "*** Error : " & Error$
Call siclcleanup
End

End Sub
```

EnterSiclLanArrayReal64SICL**iread**IEEE 64-bit **ireadiopen**

= iread(*addr,buf,bufsize,reason,actual*)

	buf
	bufsize
	reason
	actual

EnterSiclLanArrayReal64

(1)

(2)

(3) 11databuf

(4)

EnterSiclLanArrayReal64

Function EnterSiclLanArrayReal64(addr As Integer, databuf() As Double) As Long

```
Dim Status As Integer
Dim actualcnt As Long
Dim buf As String * 8
Dim size As Long
On Error GoTo ErrHandler
'''Read header info of "#6NNNNNN"
Status = iread(addr, buf, 8, I_TERM_MAXCNT, actualcnt) '.................(1)
size = Val(Mid$(buf, 3, 6)) '.................(2)
'''Read data
Status = iread(addr, databuf, size, I_TERM_MAXCNT, actualcnt)
```

```
'.................(3)
'''Read ending LF
Status = iread(addr, buf, 1, I_TERM_MAXCNT, actualcnt) '.................(4)
EnterSiclLanArrayReal64 = size / 8
Exit Function

ErrHandler:
MsgBox "*** Error : " & Error$
Call siclcleanup
End
End Function
```

SICL**iclose** **icloseiopen**

= iclose(*addr*)

E5071C SICL “”“”

```
Sub Preset()

''' Open Session
E507x = OpenSession
```

```
'''Presetting the analyzer
Call OutputSiclLan(E507x, ":SYST:PRES")
'''Close Session
Call iclose(E507x)

End Sub
```

Telnet

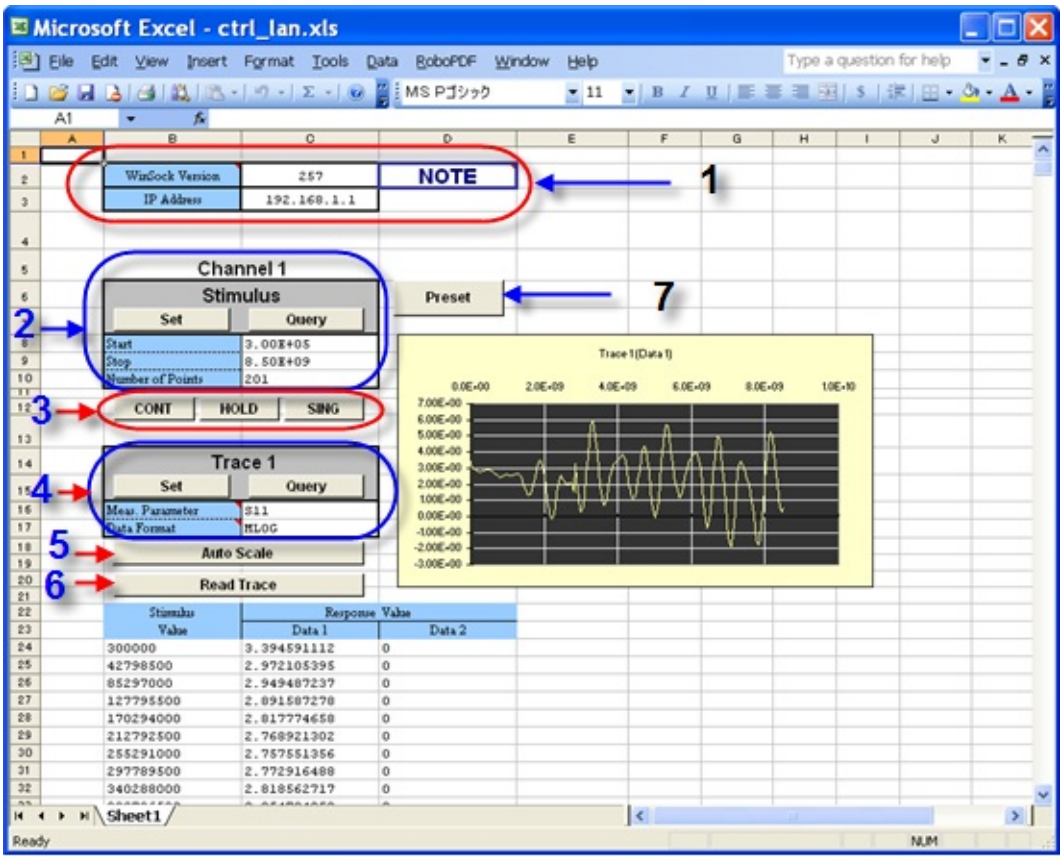
-
- Excel VBA

WindowsWinSock APIE5071C

Excel VBA

Microsoft Excel**ctrl_lan.xls**

ctrl_lan.xls



e5071c192

, ctrl_lan.xls

1. “Winsock”WinSock API256Winsock “IP”E5071CIPVBA
2. 2 “”E5071C
3. 3
4. 411 “”E5071C
5. 5 11
6. 611

7.

VBA

VBAWinSock API

WinSock APIWinSock APIWinSock API

WinSock API

'This is the Winsock API definition file for Visual Basic

'Setup the variable type 'hostent' for the WSAStartup command

Type Hostent

h_name As Long

h_aliases As Long

h_addrtype As String * 2

h_length As String * 2

h_addr_list As Long

End Type

Public Const SZHOSTENT = 16

'Set the Internet address type to a long integer (32-bit)

Type in_addr

s_addr As Long

End Type

'A note to those familiar with the C header file for Winsock

'Visual Basic does not permit a user-defined variable type

```
'to be used as a return structure. In the case of the
'variable definition below, sin_addr must
'be declared as a long integer rather than the user-defined
'variable type of in_addr.
Type sockaddr_in
sin_family As Integer
sin_port As Integer
sin_addr As Long
sin_zero As String * 8
End Type

Public Const WSADESCRIPTION_LEN = 256
Public Const WSASYS_STATUS_LEN = 128
Public Const WSA_DescriptionSize = WSADESCRIPTION_LEN + 1
Public Const WSA_SysStatusSize = WSASYS_STATUS_LEN + 1

'Setup the structure for the information returned from
'the WSAStartup() function.
Type WSAData
wVersion As Integer
wHighVersion As Integer
szDescription As String * WSA_DescriptionSize
szSystemStatus As String * WSA_SysStatusSize
iMaxSockets As Integer
```

```
iMaxUdpDg As Integer
lpVendorInfo As String * 200
End Type

'Define socket return codes
Public Const INVALID_SOCKET = &HFFFF
Public Const SOCKET_ERROR = -1

'Define socket types
Public Const SOCK_STREAM = 1 'Stream socket
Public Const SOCK_DGRAM = 2 'Datagram socket
Public Const SOCK_RAW = 3 'Raw data socket
Public Const SOCK_RDM = 4 'Reliable Delivery socket
Public Const SOCK_SEQPACKET = 5 'Sequenced Packet socket

'Define address families
Public Const AF_UNSPEC = 0 'unspecified
Public Const AF_UNIX = 1 'local to host (pipes, portals)
Public Const AF_INET = 2 'internetwork: UDP, TCP, etc.
Public Const AF_IMPLINK = 3 'arpanet imp addresses
Public Const AF_PUP = 4 'pup protocols: e.g. BSP
Public Const AF_CHAOS = 5 'mit CHAOS protocols
Public Const AF_NS = 6 'XEROX NS protocols
Public Const AF_ISO = 7 'ISO protocols
```

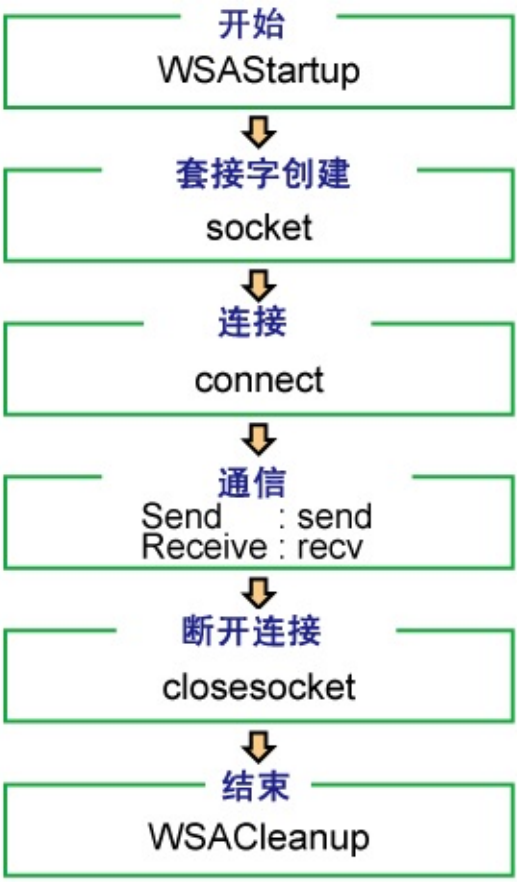
```
Public Const AF_OSI = AF_ISO 'OSI is ISO
Public Const AF_ECMA = 8 'european computer manufacturers
Public Const AF_DATAKIT = 9 'datakit protocols
Public Const AF_CCITT = 10 'CCITT protocols, X.25 etc
Public Const AF_SNA = 11 'IBM SNA
Public Const AF_DECnet = 12 'DECnet
Public Const AF_DLI = 13 'Direct data link interface
Public Const AF_LAT = 14 'LAT
Public Const AF_HYLINK = 15 'NSC Hyperchannel
Public Const AF_APPLETALK = 16 'AppleTalk
Public Const AF_NETBIOS = 17 'NetBios-style addresses
Public Const AF_MAX = 18 'Maximum # of address families

'Setup sockaddr data type to store Internet addresses
Type sockaddr
sa_family As Integer
sa_data As String * 14
End Type
Public Const SADDRLEN = 16

'Declare Socket functions

Public Declare Function closesocket Lib "wsock32.dll" (ByVal s As Long)
As Long
```

```
Public Declare Function connect Lib "wsock32.dll" (ByVal s As Long, addr As sockaddr_in, ByVal namelen As Long) As Long

Public Declare Function htons Lib "wsock32.dll" (ByVal hostshort As Long) As Integer

Public Declare Function inet_addr Lib "wsock32.dll" (ByVal cp As String) As Long

Public Declare Function recv Lib "wsock32.dll" (ByVal s As Long, ByVal buf As Any, ByVal buflen As Long, ByVal flags As Long) As Long

Public Declare Function recvB Lib "wsock32.dll" Alias "recv" (ByVal s As Long, buf As Any, ByVal buflen As Long, ByVal flags As Long) As Long

Public Declare Function send Lib "wsock32.dll" (ByVal s As Long, buf As Any, ByVal buflen As Long, ByVal flags As Long) As Long

Public Declare Function socket Lib "wsock32.dll" (ByVal af As Long, ByVal socktype As Long, ByVal protocol As Long) As Long

Public Declare Function WSAStartup Lib "wsock32.dll" (ByVal wVersionRequired As Long, lpWSAData As WSAData) As Long

Public Declare Function WSACleanup Lib "wsock32.dll" () As Long

Public Declare Function WSAUnhookBlockingHook Lib "wsock32.dll" () As Long
```

Public Declare Sub CopyMemory Lib "kernel32" Alias "RtlMoveMemory" (hpvDest As Any, hpvSource As Any, ByVal cbCopy As Long)

WinSock API

WinSock API



e5071c372

WinSock API

StartItStartIt **WSAStartup**WinSock API ctrl_lan.xls1W WSAStartup

StartIt

```
Sub StartIt()

Dim StartUpInfo As WSAData
'Version 1.1 (1*256 + 1) = 257
'version 2.0 (2*256 + 0) = 512
'Get WinSock version
Sheets("Sheet1").Select
Range("C2").Select
version = ActiveCell.FormulaR1C1
'Initialize Winsock DLL
x = WSAStartup(version, StartUpInfo)

End Sub
```

Socket CreationConnection

SocketCreationConnectionOpenSocketOpenSocketIPsocket
OpenSocket

1WinSock APIinet_aadr“.”IPInternet

2WinSock APIsocketsocketsocketSocket
socket type

3socket addresshtonsWinSock APIWindows2

4WinSock APIconnectE5071Cconnectsocket
socketsocket

OpenSocket

```
Function OpenSocket(ByVal Hostname As String, ByVal PortNumber As
Intege r) As Integer

Dim I_SocketAddress As sockaddr_in

Dim ipAddress As Long

ipAddress = inet_addr(Hostname) '...........(1)


'Create a new socket

socketId = socket(AF_INET, SOCK_STREAM, 0) '

If socketId = SOCKET_ERROR Then '

MsgBox ("ERROR: socket = " + Str$(socketId)) '...........(2)

OpenSocket = COMMAND_ERROR '

Exit Function '

End If '


'Open a connection to a server


I_SocketAddress.sin_family = AF_INET '

I_SocketAddress.sin_port = htons(PortNumber) '...........(3)

I_SocketAddress.sin_addr = ipAddress '

I_SocketAddress.sin_zero = String$(8, 0) '


x = connect(socketId, I_SocketAddress, Len(I_SocketAddress)) '

If socketId = SOCKET_ERROR Then '

MsgBox ("ERROR: connect = " + Str$(x)) '..(4)

OpenSocket = COMMAND_ERROR '
```

```
Exit Function '
End If '

OpenSocket = socketId

End Function
```

SendCommandSendCommand“”SCPIWinSock
E5071C **send**socket

SendCommand

```
Function SendCommand(ByVal command As String) As Integer

Dim strSend As String
strSend = command + vbCrLf
count = send(socketId, ByVal strSend, Len(strSend), 0)
If count = SOCKET_ERROR Then
MsgBox ("ERROR: send = " + Str$(count))
SendCommand = COMMAND_ERROR
Exit Function
End If
SendCommand = NO_ERROR

End Function
```

RecvAscii RecvAsciiASCIIRecvAscii

1WinSock APIrecvE5071CSCPIrecvsocket

2ASCII10NULLASCII0

3

RecvAscii

Function RecvAscii(dataBuf As String, ByVal maxLength As Integer) As Integer

```
Dim c As String * 1
Dim length As Integer
dataBuf = ""
While length < maxLength
DoEvents
count = recv(socketId, c, 1, 0) '
If count < 1 Then '
RecvAscii = RECV_ERROR '.............(1)
dataBuf = Chr$(0) '
Exit Function '
End If '
If c = Chr$(10) Then '
dataBuf = dataBuf + Chr$(0) '.............(2)
RecvAscii = NO_ERROR '
Exit Function '
End If '
length = length + count '.............(3)
dataBuf = dataBuf + c '
```

Wend

RecvAscii = RECV_ERROR

End Function

CloseConnectionCloseConnectionWinSock API **closesoc**
Closesocketsocket

c

Sub CloseConnection()

x = closesocket(socketId)

If x = SOCKET_ERROR Then

MsgBox ("ERROR: closesocket = " + Str$(x))

Exit Sub

End If

End Sub

End

EndItEndItWinSock APIWSACleanupWinSock APIWinSock
WSACleanup

EndIt

Sub EndIt()

'Shutdown Winsock DLL

x = WSACleanup()

```
End Sub
```

E5071C WinSock API

```
Sub autoscale()
'
' auto scaling
'
Call StartIt
Call get_hostname
x = OpenSocket(Hostname$, ScpiPort)
x = SendCommand(":DISP:WIND1:TRAC1:Y:AUTO")
Call CloseConnection
Call EndIt
End Sub
```

socket

LCD

-
- HT Basic

LCD

/9/9

LCD10

HT Basiccont_upd.htb

```
10 REAL Trace1(1:201,1:2),Trace2(1:201,1:2)
20 DIM Buff$[9],Img$[30]
30 INTEGER Nop,I
40 !
50 ASSIGN @Agte507x TO 717
60 ASSIGN @Binary TO 717;FORMAT OFF
70 !
80 OUTPUT @Agte507x;":SENS1:SWE:TYPE LIN"
90 OUTPUT @Agte507x;":SENS1:FREQ:CENT 950E6"
100 OUTPUT @Agte507x;":SENS1:FREQ:SPAN 100E6"
110 OUTPUT @Agte507x;":SENS1:SWE:POIN 201"
120 OUTPUT @Agte507x;":TRIG:SOUR BUS"
130 OUTPUT @Agte507x;":INIT1:CONT ON"
140 FOR I=2 TO 9
150 OUTPUT @Agte507x;":INIT"&VAL$(I)&":CONT OFF"
160 NEXT I
170 !
180 OUTPUT @Agte507x;":DISP:SPL D1"
190 OUTPUT @Agte507x;":DISP:WIND1:SPL D1_2"
200 !
210 OUTPUT @Agte507x;":CALC1:PAR:COUN 2"
220 OUTPUT @Agte507x;":CALC1:PAR1:DEF S21"
230 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
```

```
240 OUTPUT @Agte507x;":CALC1:FORM MLOG"
250 OUTPUT @Agte507x;":CALC1:PAR2:DEF S11"
260 OUTPUT @Agte507x;":CALC1:PAR2:SEL"
270 OUTPUT @Agte507x;":CALC1:FORM MLOG"
280 !
290 OUTPUT @Agte507x;":DISP:ENAB OFF"
300 OUTPUT @Agte507x;":FORM:DATA REAL"
310 !
320 FOR I=1 TO 10
330 OUTPUT @Agte507x;":TRIG:SING"
340 OUTPUT @Agte507x;"*OPC?"
350 ENTER @Agte507x;Buff$
360 !
370 ! Read Trace Data
380 !
390 OUTPUT @Agte507x;":CALC1:PAR1:SEL"
400 OUTPUT @Agte507x;":CALC1:DATA:FDAT?"
410 ENTER @Agte507x USING "#,8A";Buff$
420 ENTER @Binary;Trace1(*)
430 ENTER @Agte507x USING "#,1A";Buff$
440 !
450 OUTPUT @Agte507x;":CALC1:PAR2:SEL"
460 OUTPUT @Agte507x;":CALC1:DATA:FDAT?"
470 ENTER @Agte507x USING "#,8A";Buff$
```

```
480 ENTER @Binary;Trace2(*)
490 ENTER @Agte507x USING "#,1A";Buff$
500 !
510 ! Update Display
520 !
530 OUTPUT @Agte507x;":DISP:UPD"
540 NEXT I
550 END
```

5060

I/OGPIB

80110

950 MHz100 MHz201

120160

129

180190

1

210270

121S21Log Mag2

290

LCD

300

320540

340360

40044011

46050012

540LCD

-
-

I/O

520I/OAC3

/

Excel VBA

```
Sub Handler_Click()
    Dim defrm As Long           'Session to Default Resource Manager.
    Dim vi As Long              'Session to instrument.
    Dim Out_Data As Integer     'Decimal value.
    Dim In_Data As Long
    Dim Bit_stat As Integer
    Dim Flag_bit As Integer
    Dim Out_Data_Bin As String
    Dim Ret As Long             'Return value.
    Dim i As Long
    Dim X As Long
    Const TimeOutTime = 40000    'timeout time.
    Out_Data_Bin = "00000101"      'Store the output data on the port A
(binaly).
    Flag_bit = 3                            'Bit location (bit 3).


    Call viOpenDefaultRM(defrm)    'Initializes the VISA system.
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)   'Opens the session
to the specified instrument.
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)   'The
state of an attribute for the specified session.


    Call viVPrintf(vi, "*RST" & vbLf, 0)    'Presets the setting state of the
```

```
ENA.
    Call viVPrintf(vi, "*CLS" & vbLf, 0)   'Clears the all status register.

    Call viVPrintf(vi, ":CONT:HAND:C:MODE INP" & vbLf, 0)     'Configures
the port C to input port.
    Call viVPrintf(vi, ":CONT:HAND:IND:STAT ON" & vbLf, 0)    'Line
enable /INDEX signal.
    Call viVPrintf(vi, ":CONT:HAND:RTR:STAT ON" & vbLf, 0)   'Line
enable /READY FOR TRIGGER signal.

    For i = 1 To Len(Out_Data_Bin)    'Convert Out_Dara_Bin to a decimal
value.
        If Mid(Out_Data_Bin, Len(Out_Data_Bin) - i + 1, 1) = "1" Then
            X = 2 ^ (i - 1)
            Ret = Ret + X
        End If
    Next i
    Out_Data = Ret       'Sets the decimal value.
    Call viVPrintf(vi, ":CONT:HAND:A " & Ret & vbLf, 0)    'Sets to the port
A.

    Call viVPrintf(vi, ":CONT:HAND:C?" & vbLf, 0)   'Outputs data to output
port C.
    Call viVScanf(vi, "%t", In_Data)    'Reads data from the port C.

    Call ErrorCheck(vi)      'Checking the error.
```

```
    Call viClose(vi)            'Closes the resource manager session.
    Call viClose(defrm)     'Breaks the communication and terminates the
VISA system.

    End
End Sub
Sub ErrorCheck(vi As Long)
    Dim err As String * 50, ErrNo As Variant, Response

    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)    'Reads error
message.
    ErrNo = Split(err, ",")  'Gets the error code.

    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)  'Display the
message box.
    End If
End Sub
```

HT Basic (handler.htb)

```
10 INTEGER Out_data,In_data,Bit_stat
20 DIM Out_data_bin$[9]
30 !
40 ASSIGN @Agte507x TO 717
50 !
```

```
60 Out_data_bin$="00000101"
70 Flag_bit=3
80 !
90 OUTPUT @Agte507x;":CONT:HAND:C:MODE INP"
100 OUTPUT @Agte507x;":CONT:HAND:IND:STAT ON"
110 OUTPUT @Agte507x;":CONT:HAND:RTR:STAT ON"
120 !
130 Out_data=IVAL(Out_data_bin$,2)
140 OUTPUT @Agte507x;":CONT:HAND:A ";Out_data
150 !
160 REPEAT
170 OUTPUT @Agte507x;":CONT:HAND:C?"
180 ENTER @Agte507x;In_data
190 Bit_stat=BIT(In_data,Flag_bit)
200 UNTIL Bit_stat=1
210 END
```

E5091A

-
- Excel VBA
- HT Basic

E5091A1A2T23R2+4R2-13

E5091A

Excel VBA

```
Sub e5091ctrl_Click()
    Dim defrm As Long          'Session to Default Resource Manager
    Dim vi As Long             'Session to instrument
    Dim Port1 As String
    Dim Port2 As String
    Dim Port3 As String
    Dim Port4 As String
    Dim Line1 As String
    Dim Line2 As String
    Dim Line3 As String
    Dim Line4 As String
    Dim Line5 As String
    Dim Line6 As String
    Dim Line7 As String
    Dim Line8 As String
    Dim Data_Bin As String
    Dim Data_Dec As Long
    Dim i As Integer
    Dim X As Long
    Dim Model As String

    Const TimeOutTime = 20000
```

```
Model = "E5091_9"                                   'Model     E5091A-
009

Port1 = "A"                                         'Port1 = A
Port2 = "T2"                                        'Port2 = T2
Port3 = "R2"                                        'Port3 = R2+
Port4 = "R2"                                        'Port4 = R2-

Line1 = "1"                                         'Line1 = High
Line2 = "0"                                         'Line2 = Low
Line3 = "1"                                         'Line3 = High
Line4 = "0"                                         'Line4 = Low
Line5 = "0"                                         'Line5 = Low
Line6 = "0"                                         'Line6 = Low
Line7 = "0"                                         'Line7 = Low
Line8 = "0"                                         'Line8 = Low

Call viOpenDefaultRM(defrm)
                                    'Initializes the VISA system.
Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
                        'Opens the session to the specified instrument.
Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
                  'The state of an attribute for the specified session.

Call viVPrintf(vi, "*RST" & vbLf, 0)                    'Presets
the setting state of the ENA.
```

```
    Call viVPrintf(vi, "*CLS" & vbLf, 0)                                    'Clears
the all status register.


    Call viVPrintf(vi, ":SENS1:MULT1:NAME " & Model & vbLf, 0)
                    'Specifies the test set name.


    Call viVPrintf(vi, ":SENS:MULT:TSET9:PORT1 " & Port1 & vbLf, 0)
            'Sets the port assigned to port 1 to A
    Call viVPrintf(vi, ":SENS:MULT:TSET9:PORT2 " & Port2 & vbLf, 0)
            'Sets the port assigned to port 2 to T2
    Call viVPrintf(vi, ":SENS:MULT:TSET9:PORT3 " & Port3 & vbLf, 0)
            'Sets the port assigned to port 3 to R2+
    Call viVPrintf(vi, ":SENS:MULT:TSET9:PORT4 " & Port4 & vbLf, 0)
            'Sets the port assigned to port 4 to R2-


    Call ErrorCheck(vi)                                                     'Checking the
error.


    Data_Bin = Line8 & Line7 & Line6 & Line5 & Line4 & Line3 & Line2 &
Line1


    For i = 1 To Len(Data_Bin)                                              'Creates a
decimal setting value.
        If Mid(Data_Bin, Len(Data_Bin) - i + 1, 1) = "1" Then
            X = 2 ^ (i - 1)
            Data_Dec = Data_Dec + X
        End If
    Next i
```

```
    Call viVPrintf(vi, ":SENS:MULT:TSET9:OUTP " & CStr(Data_Dec) &
vbLf, 0)        'Sets the control line.


    Call viVPrintf(vi, ":SENS:MULT:DISP ON" & vbLf, 0)
                        'Turns on the E5091A property display.
    Call viVPrintf(vi, ":SENS:MULT:STAT ON" & vbLf, 0)
                        'Turns on the control of the E5091A.


    Call ErrorCheck(vi)                                         'Checking the
error.


    Call viClose(vi)                                          'Closes the
resource manager session.
    Call viClose(defrm)                                         'Breaks the
communication and terminates the VISA system.


    End                                                     'End
End Sub
Sub ErrorCheck(vi As Long)
    Dim err As String * 50, ErrNo As Variant, Response


    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)
                        'Reads error message.
    ErrNo = Split(err, ",")                                   'Gets the error
code.


    If Val(ErrNo(0)) <> 0 Then
```

```
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)
                                    'Display the message box.
    End If
End Sub
```

HT Basice5091ctr_2.htb

```
10 DIM Port1$[3],Port2$[3],Port3$[3],Port4$[3],Data_bin$[9]
20 DIM Line1$[3],Line2$[3],Line3$[3],Line4$[3]
30 DIM Line5$[3],Line6$[3],Line7$[3],Line8$[3]
40 DIM Model$[8]
50 INTEGER Data_dec
60 !
70 ASSIGN @Agte507x TO 717
80 !
90 Model$="E5091_9" ! Model:E5091A-009
100 !
110 Port1$="A" ! Port1: A
120 Port2$="T2" ! Port2: T2
130 Port3$="R2" ! Port3: R2+
140 Port4$="R2" ! Port4: R2-
150 !
160 Line1$="1" ! Line1: HIGH
170 Line2$="0" ! Line2: Low
180 Line3$="1" ! Line3: HIGH
190 Line4$="0" ! Line4: Low
200 Line5$="0" ! Line5: Low
210 Line6$="0" ! Line6: Low
220 Line7$="0" ! Line7: Low
230 Line8$="0" ! Line8: Low
```

```
240 !
250 OUTPUT @Agte507x;":SENS1:MULT1:NAME "&Model$
260 !
270 OUTPUT @Agte507x;":SENS1:MULT1:PORT1 "&Port1$
280 OUTPUT @Agte507x;":SENS1:MULT1:PORT2 "&Port2$
290 OUTPUT @Agte507x;":SENS1:MULT1:PORT3 "&Port3$
300 OUTPUT @Agte507x;":SENS1:MULT1:PORT4 "&Port4$
310 !
320
Data_bin$=Line8$&Line7$&Line6$&Line5$&Line4$&Line3$&Line2$&Line
330 Data_dec=IVAL(Data_bin$,2)
340 OUTPUT @Agte507x;":SENS1:MULT1:OUTP ";Data_dec
350 !
360 OUTPUT @Agte507x;":SENS:MULT1:DISP ON"
370 OUTPUT @Agte507x;":SENS:MULT1:STAT ON"
380 !
390 END
```

VBA

- VBA
-
- E5071C
-
-
-
-

VBA

- E5071C
-
- E5071C COM

E5071C

-
-

VBA

E5071CE5071CVBAVisual BasicE5071C

VBAVBVisual BasicVBAVBVBAVBE5071C
VBAVBVBA

E5071CVBAE5071CVBAVBAVBAVBAVBA

E5071C

1. E5071CSubSub
2. E5071C VBA E5071C

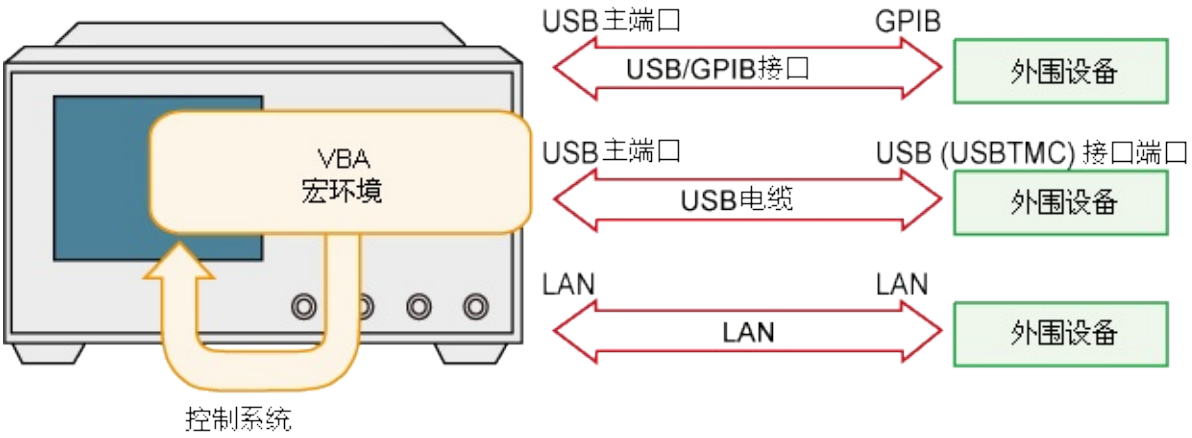
-
-
-
-

VBA

E5071CVBAE5071C

VBAE5071CVBAUSB/GPIBUSBLANE5071C
VISAVISA

Progr



e5071c478

- E5071C
-
- USB/GPIBUSBLAN

E5071C

E5071C

E5071CE5071C VBACOME5071CCOMCOM7SCPICOM

E5071C VBAVISA

VISA Controlling Peripherals VISAVISA

GPIB

E5071C COM

-
- COM
-
-
-
- COME5071C
- COMSCPI

VBA

E5071C VBAE5071CCOMCOME5071CCOM

COMCOM

COM

E5071CCOME5071C COM

E5071CE5071CE5071C

COM

E5071C

COM

E5071CUserMenu_OnPressByVal Key_id

COME5071C

COMCOMSCPIE5071C SCPIE5071CVBA EditorCOM E5071CVBA

COMSCPI

SCPISRQCOMSRQ WaitOnSRQSRQE5071C

- Visual Basic Editor
- Visual Basic Editor
- E5071C
-
- VBA
- VBA
- VBA
- VBA
- VBA
-
- Echo
-
- VBA

Visual Basic Editor

-
- Visual Basic Editor

Visual Basic EditorBasic

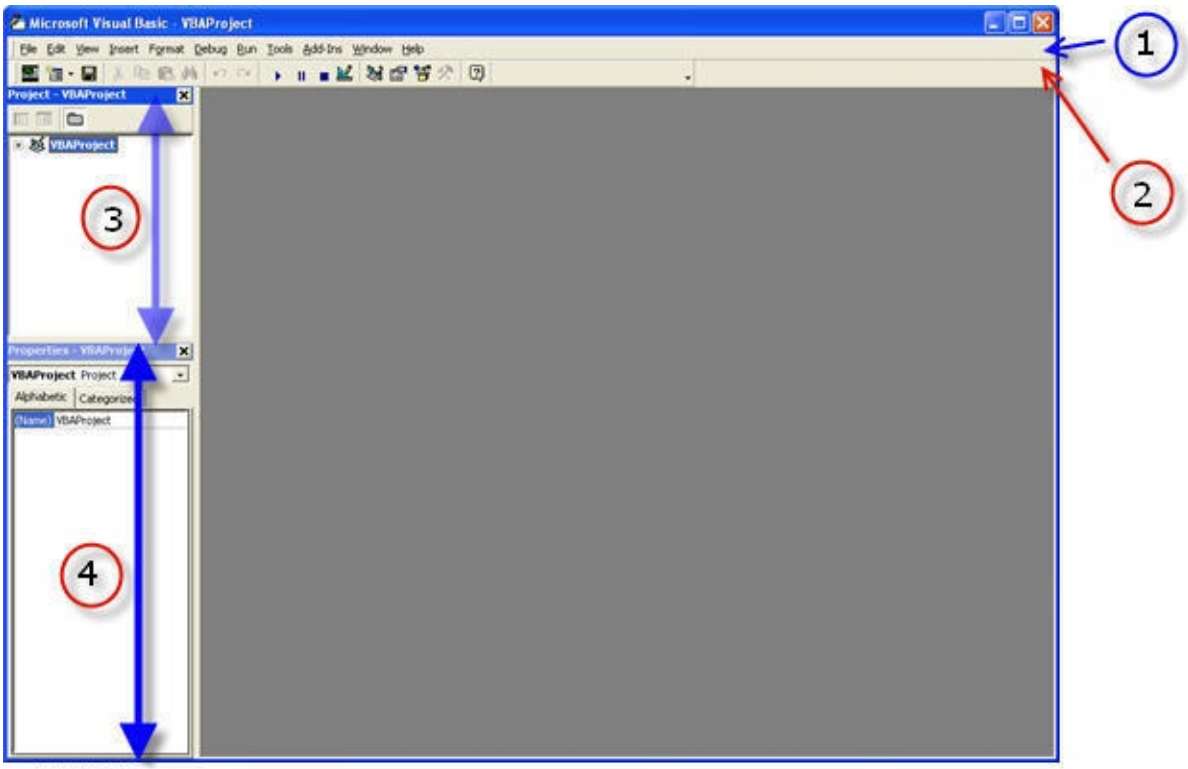
E5071CVisual Basic Editor

1. > **VBA Editor**
2. **Alt + F11**

Visual Basic Editor

Visual Basic EditorGUI

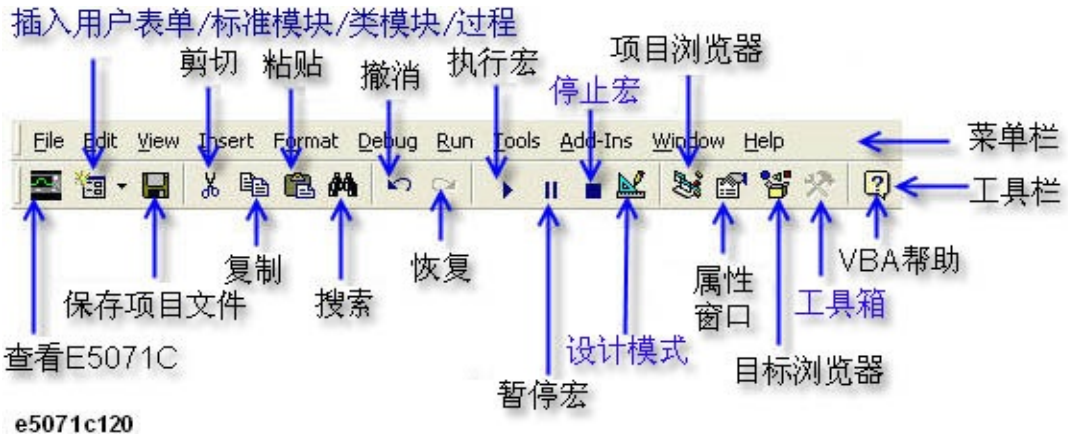
Visual Basic Editor



e5071c119

1.
2.
3. Project Explorer
4. Property Window

E5071CVBA



Project Explorer

E5071CVBAProject ExplorerVisual

A Project and Three Types of Module

1. **View** **Project Explorer**
2. **Ctrl + R**
3. Project Explorer.

Property Window

User Form

1. **View** **Properties Window**
2. **F4**
3. Properties Window.

Visual Basic Editor

Visual Basic Editor Visual Basic Editor

- Visual Basic Editor**File** **Close and Return to E5071** E5071
- Visual Basic Editor**Alt** + **Q**
- **Macro Setup** > **Close EditorEditor**

Visual Basic EditorE5071CVBA

E5071C

Visual Basic EditorE5071C

- **View** **E5071**
- **Alt** + **F11**
- “E5071C”
- E5071C**Focus**

-
-

Project Explorer E5071C VBA“”“”“”“”

E5071CVBAVBA".vba"

".frm"

SubEnd Sub".bas"

".cls"

Visual Basic Editor

E5071CVBAVisual Basic EditorE5071C

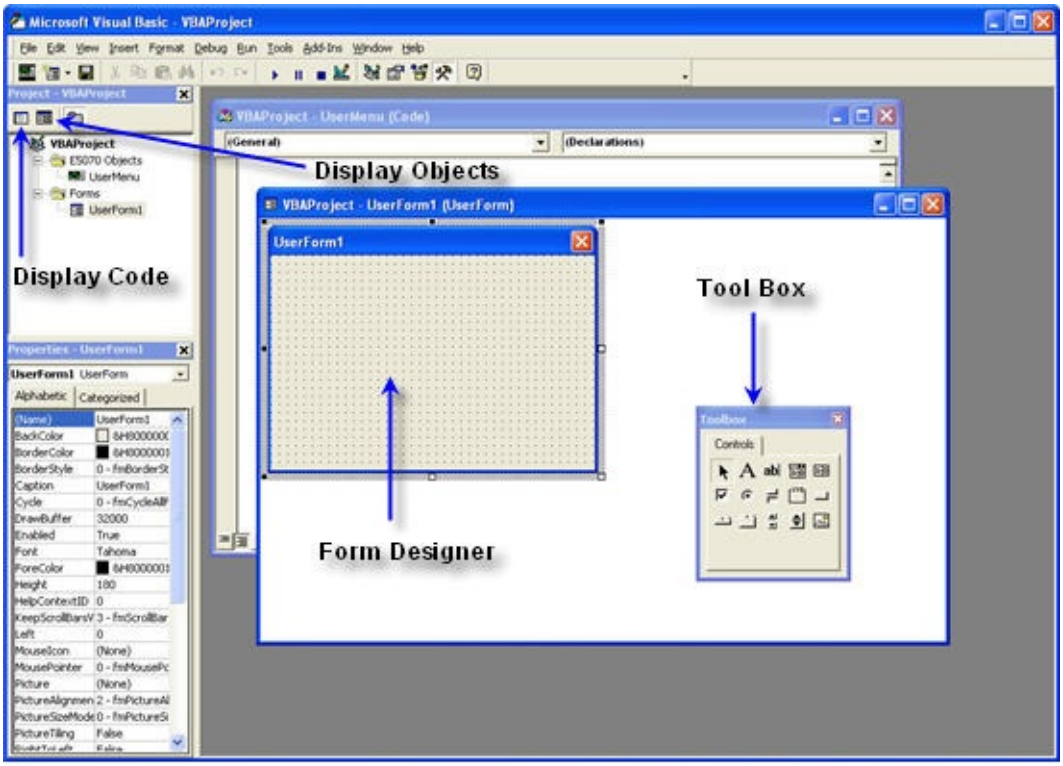
1. **Macro Setup** > **New Project**

“” **Yes** Saving a Project

Visual Basic Editor“”

1. **Insert** **UserForm**
2. "Insert User Form/Standard Module/Class Module/Procedure"/// **UserF**
3. Project Explorer"VBAProject" **Insert** > **UserForm**

"Display

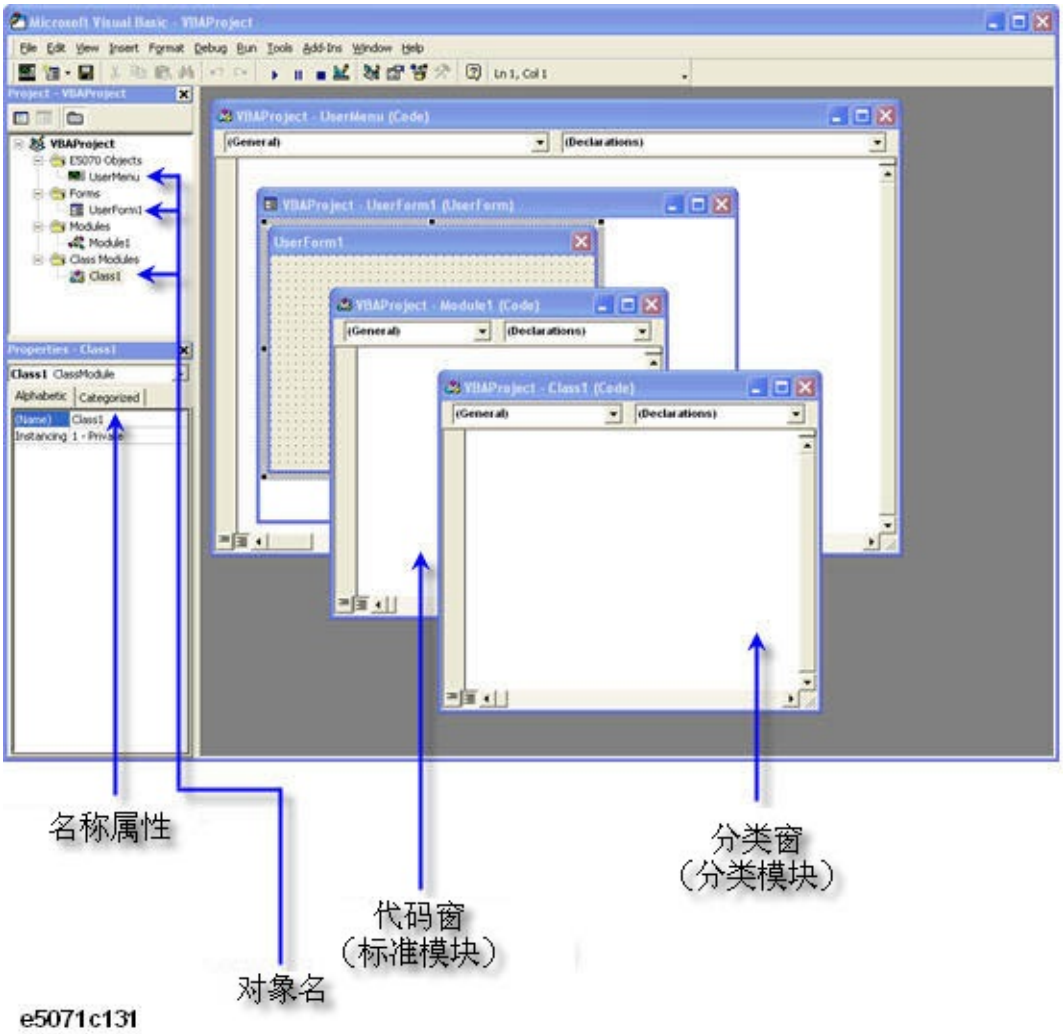


e5071c130

Visual Basic Editor“”

1. **Insert** **Module**
2. "Insert User Form/Standard Module/Class Module/Procedure"/// **Module**
3. Project Explorer"VBAProject" **Insert** > **Module**

/



Visual Basic Editor“”

1. **Insert** **ClassModule**
2. "Insert User Form/Standard Module/Class Module/Procedure"/// **ClassMc**
3. In Project Explorer"VBAProject" **Insert** > **ClassModule**

Visual Basic Editor"Class1"

1. Project Explorer"Class1" class module under the "Class Modules""""Class1"

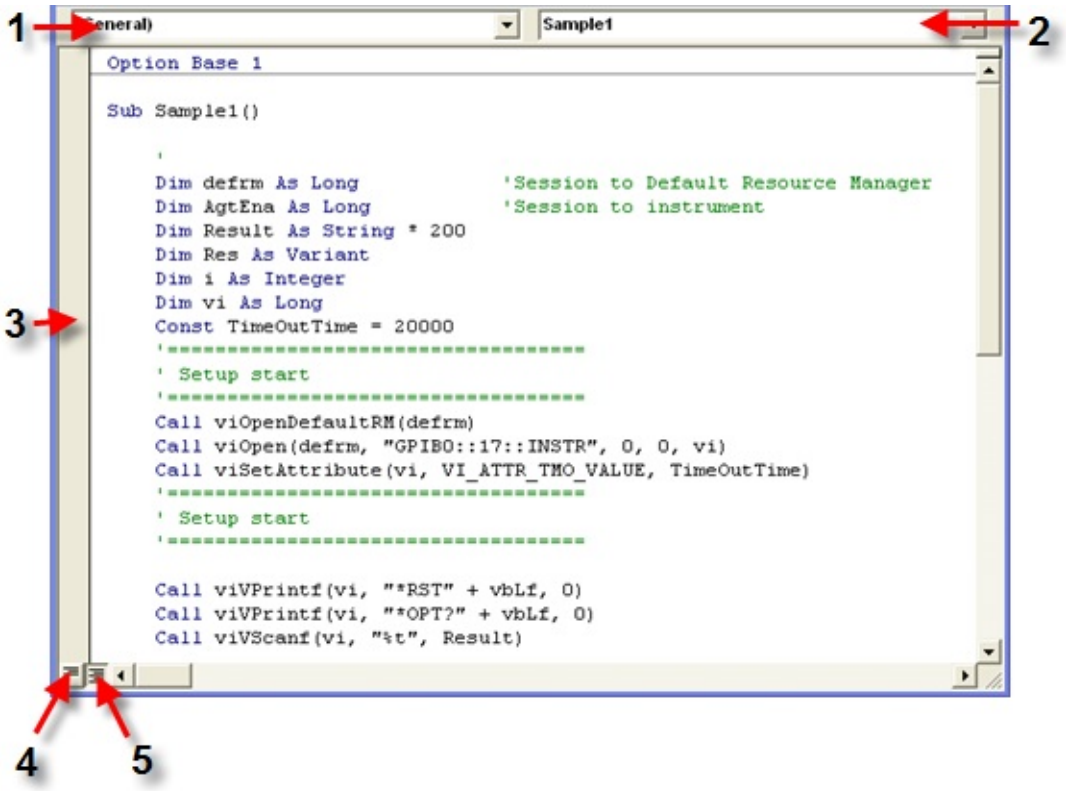
2. "Class1"
    a. **File** **Remove Class1...**.
    b. **Remove Class1...**.
3. "Class1" **No** **Yes**

VBA

-
-
- VBA
-

VBA

VBA



e5071c251

1.

2.

3.

4.

5.

VBA

```
10| Option Explicit
20|
30| Sub Maximum()
40|
50| Dim q As Variant
60| Dim x(100) As Integer
70| Dim i As Integer, n As Integer
80| Dim Start As Integer, Last As Integer, Num As Integer
90| Dim Maximum As Integer
100|
110| ' Defining the array
120| q = Array(7, -2, 3, -20, 15, -6, 27, -12, 9, -5, 18, 23, _
130| 9, -16, 22, 0)
140|
150| Start = LBound(q)
160| Last = UBound(q)
170| Num = Last - Start + 1
180|
190| For i = Start To Last
200| x(i) = q(i)
210| Next i
```

220|

230| Maximum = x(Start)

240|

250| For n = Start + 1 To Last

260| If x(n) > Maximum Then Maximum = x(n)

270| Next n

280|

290| MsgBox Maximum

300|

310| End Sub

10

30310

SubMaximumEnd SubE5071C“Maximum”

5090

Dim“q”Variant
“x100”“i”“n”“Start”“Last”“Num”“Maximum”IntegerVBA
VBA

110

'

120130

VBAq0_2

150

q

160

q

170

qNum

190210250270

For?/span> ?/span> ?/span> Next

200

qVariantxInterger

230

x1

260

290

VBAVBA

Visual Basic EditorCOMeditor

SCPI.INITiate(Ch).CONTinuous

1. **sub mainEnter End Sub**
2. **scpi**(**.**)SCPI
3. **inINITiate**
4. **(**
5. “1).”INITiate
6. **cCONTinuous**
7. =Boolean(True/False)/
8. tTrue
9. **Enter**SCPI.INITiate(1).CONTinuous = True

VBA

-
-
- _____

VBAVBA

.vba

Visual Basic Editor

1.
    - **FileSave xxx.VBAxxx.VBA“ xxx”**
    - “”
    - **Ctrl** + **S**
2. **Save**

E5071CE5071C

1. E5071C E5071C E5071C
2.
    - **Macro Setup > Save Project**
    - **>**
3. **Save As dialog box** Save

VBAVisual Basic Editor.frm.bas.cls

a.

b.

- **File** **Export File** ...
- **Export File** **....**
- **Ctrl** + **E**

c. **Export File dialog box** **Save**

VBA

-
-

VBA

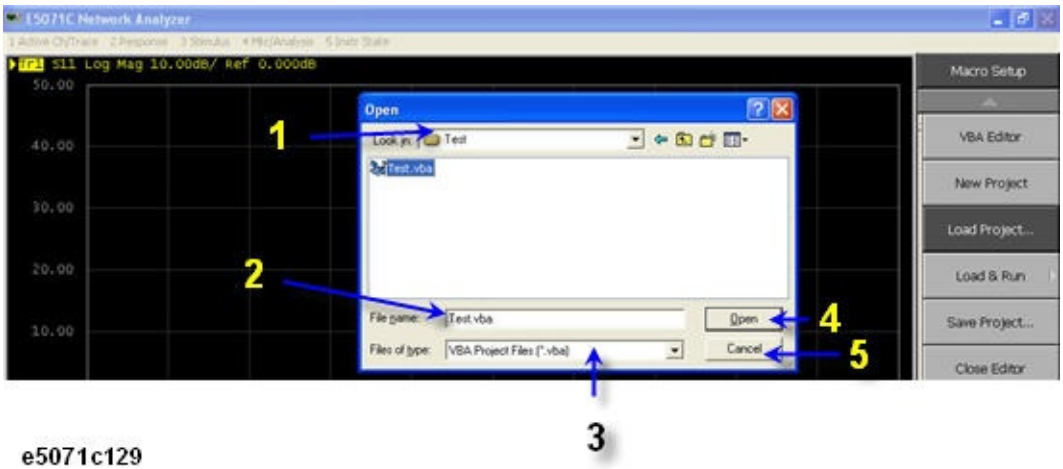
E5071C

E5071C

1. **Macro Setup** **Load Project**

注意 Visual Basic Editor **Yes**

2. **Open**.



e5071c129

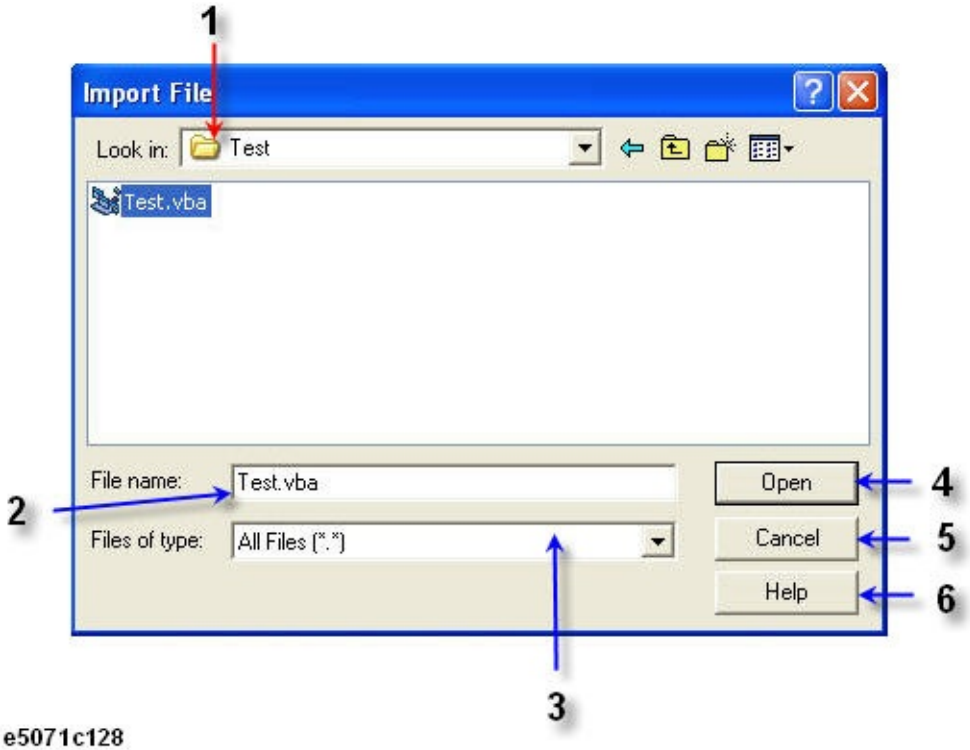
1.
2.
3. : **VBA Project Files [*.vba]**VBA[*.vba]
4.
5.

	A: or **D:**
	autoload.vba

AD“autoload.vba”A

Visual Basic Editor

1.
2.
   - **File** **Import File**
   - “VBAProject” **Import File** .
   - **Ctrl** + **M**
3. **Open**
4.



e5071c128

1. :
2.
3. **VB [*.frm,*.bas,*.cls]**
4.
5.
6. VBA

VBA

-
- VBA
-

E5071C2VBAVBA“”“”

VBA

e5071c132

VBA

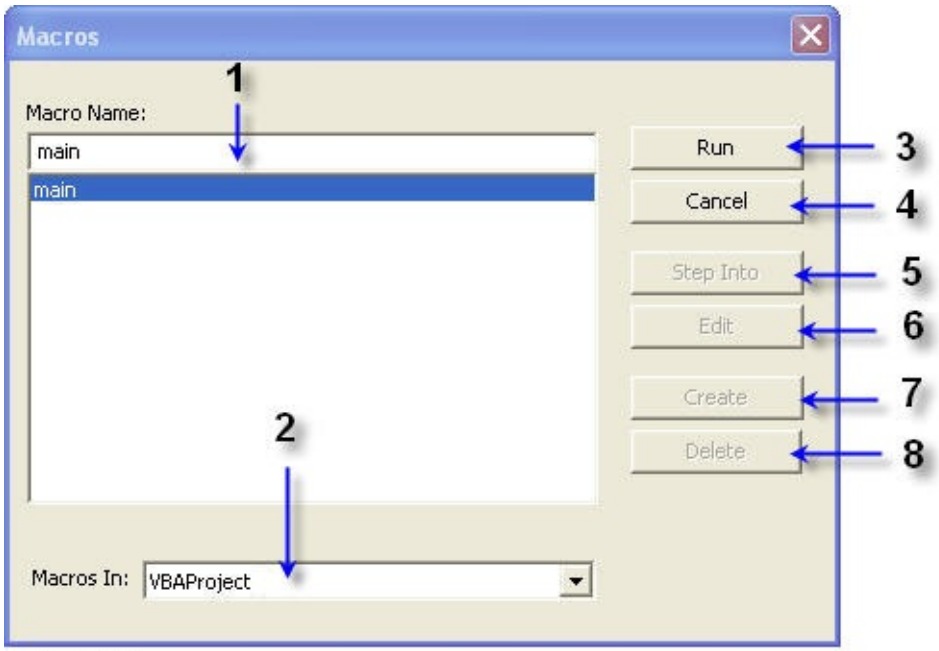
E5071C4VBA

Visual Basic Editor

1.

- **RunRun Macro**
- **ToolsMacros**....
- “”
- **F5**

2. VBA **Run**



e5071c133

1. VBA

2. **In** VBA
3. VBA
4.
5. Visual Basic EditorVBAVBA
6. VBA
7.
8. VBAVBAVBA

SubEnd Sub

E5071C

E5071C4E5071C

1. E5071CE5071C
2. VBA

- **Macro Setup** > **Select Macro - Module xxx**
- > **-**xxx
- “ ”“ **xxx**”

- E5071C**Macro Run** “Main”Sub “Module1”

E5071CE5071CSubEnd

VBA

1. VBAVBA
   D:\VBA

注意 **D:\VBA** **D:\VBA**

注意 VBAD:\VBAVBAD:\VBAVBA

2. **Macro Seup**
3. **Load & Run**
4. VBAVBA“Main”Sub
   “Module”

D:\VBAVBA 50

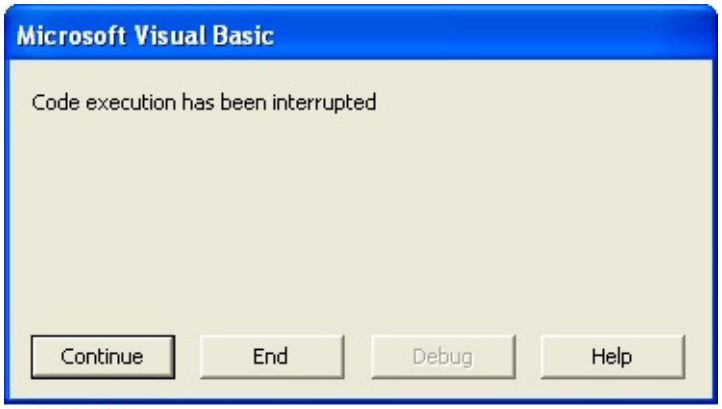
- **D:\VBA**VBA
- 121312“...”“...”.vba

VBA

-
- VBA

VBA

1. VBA
   - **Run** **Break**
   - “”
   - **Ctrl** + **Break**
   - >**Macro Setup** **Stop** E5071C
   - E5071C **Macro Break**
2.



e5071c250

   -
   - VBA
   -
   - VBA

VBA

VBALCD“”

VBA

- **Run** **Reset**
- ,“”
- *End*

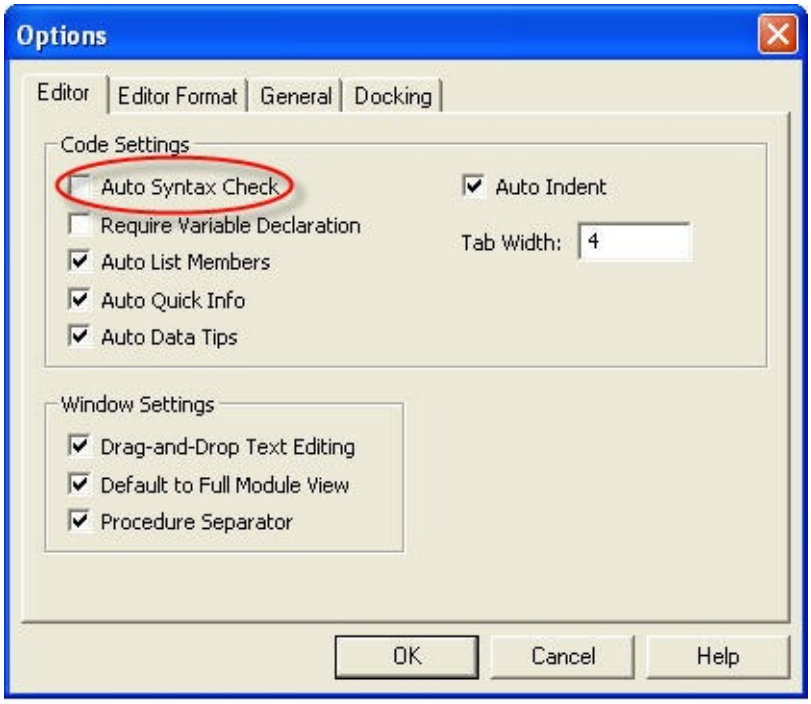
-
-

VBA

Visual Basic EditorVisual Basic **HELP**

E5071C VBA

1. **Tools** **Options...**.
2. **Editor** **Auto Syntax Check**



e5071c121

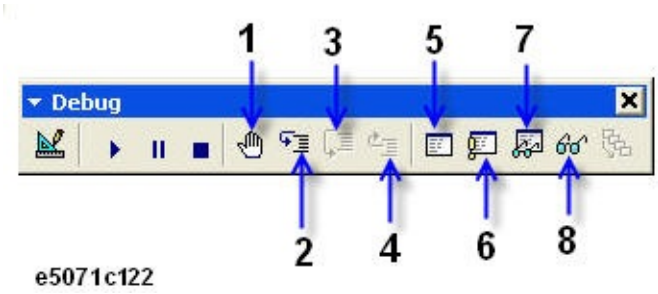
3. **OK**

VBA ENDDEBUG

"Target
module not in RF path""EcalRF"

E5071CVBAVBAVBA

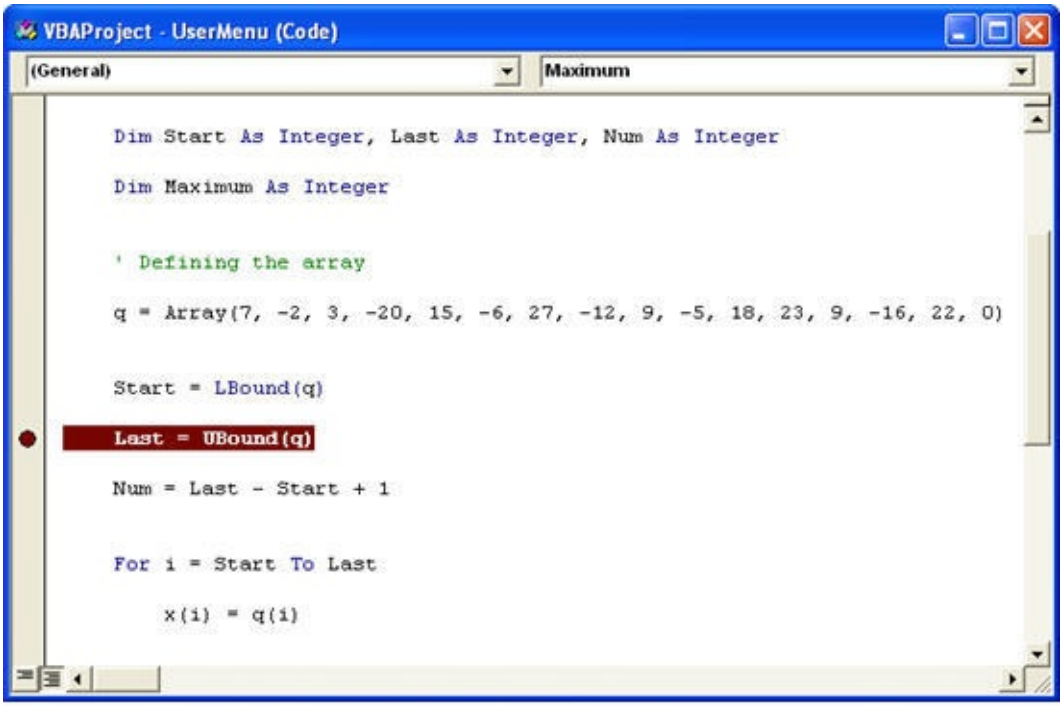
View **Toolbars** > **Debug**



1. / **F9**
2. **F8**VBA
3. **Shift** + **F8** VBA
4. **Ctrl** + **Shift** + **F8**
5.
6. **Ctrl** + **G**
7.
8. **Shift** + **F9**

VBA

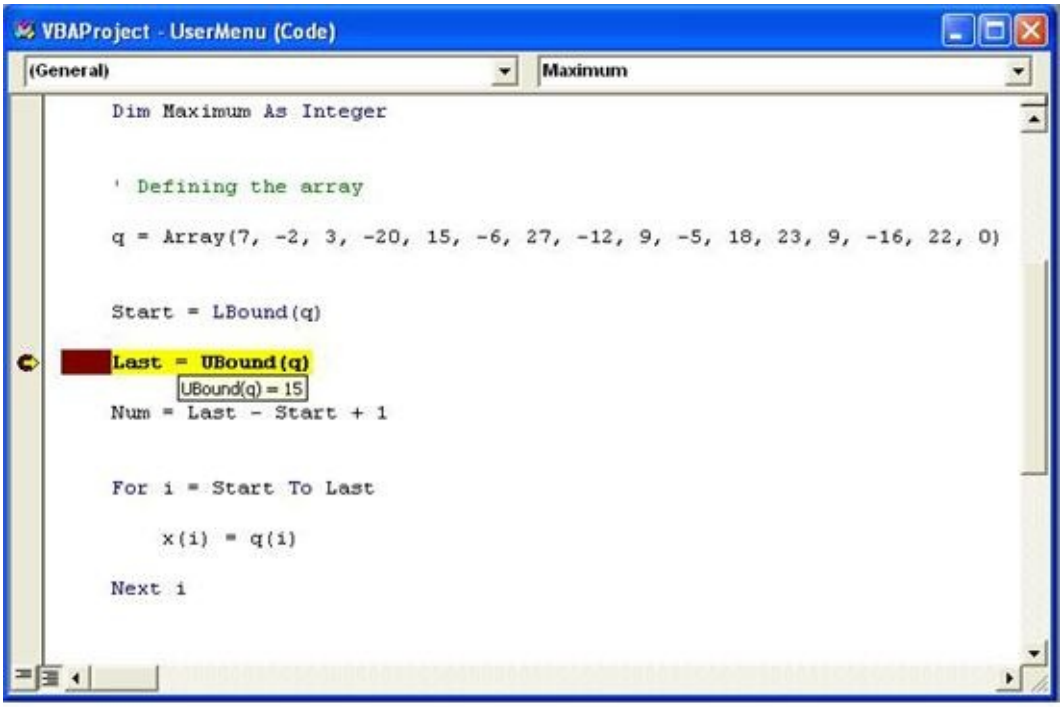
- "Set/clear break points" “/”
-



e5071c123

VBAVBA

Data Hint



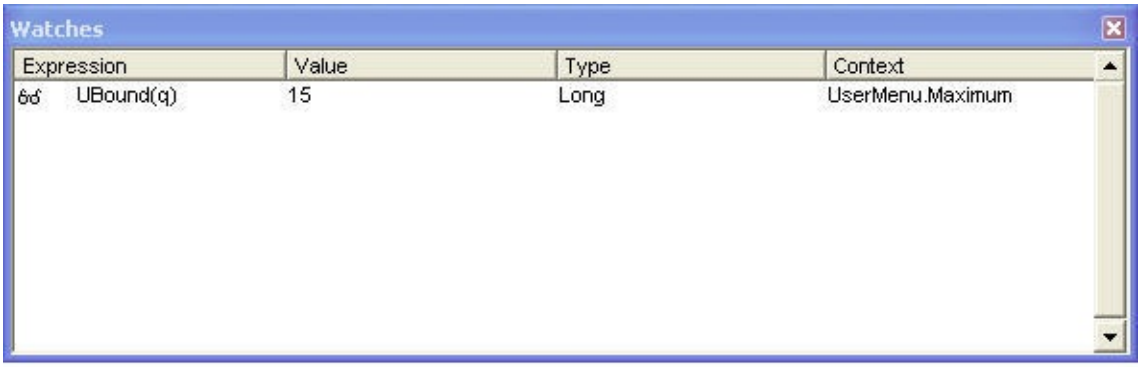
e5071c127

Enter

e5071c124

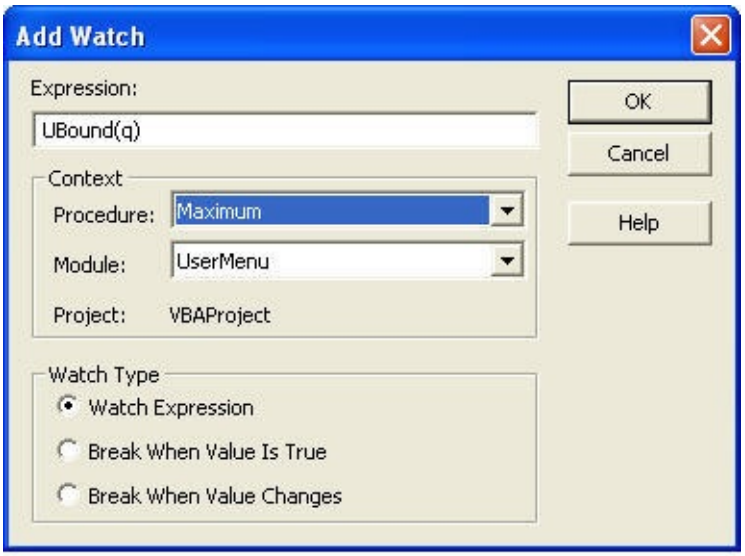
“”

.



e5071c126

1. **Debug**A **Add Watch...**....
2.
3. **OK**



e5071c125

“”

Add

Echo

-
- Echo
- Echo
- Echo

E5071Cecho

Echo

COMecho

- ECHO
- SCPI.DISPlay.ECHO.DATA

Echo

COMecho

- SCPI.DISPlay.TABLe.TYPE
- SCPI.DISPlay.TABLe.STATe

echo

Macro Setup > **Echo Window (ON)**

> **Echo**

Echo

COMecho

- SCPI.DISPlay.ECHO.CLEar

echo

Macro Setup > **Clear Echo**

> **Echo**

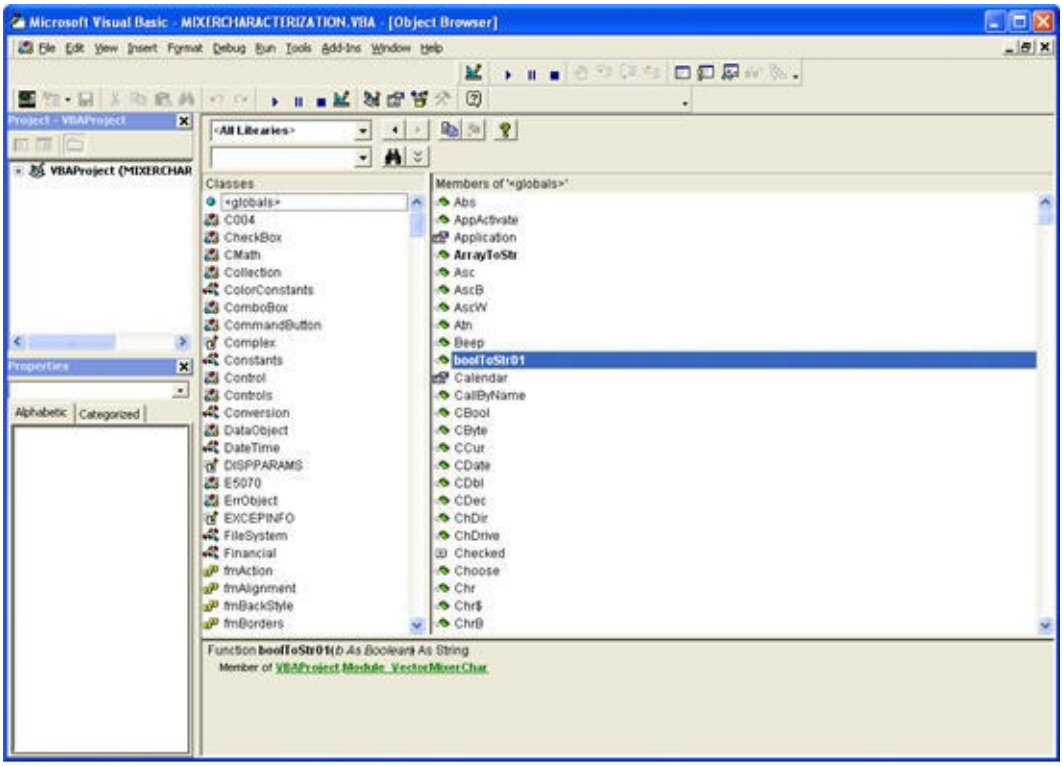
- E5071C COM
-

E5071C COM

E5071C VBAE5071CCOME5071C COMVisual Basic Editor E5071C COM

1.
    - **View Object Browser**
    - , “”
2. / **E5071Lib**E5071C

COME5071C VBAE5071C COME5071C VBACOMCOM



e5071c134

Macro Setup > **New Project**)

	D:\Agilent
	olb, tlb, dll, or ocx

Visual Basic Editor

- **Tools** **References...**

VBA

-
- VBA

VBAVBAVBA

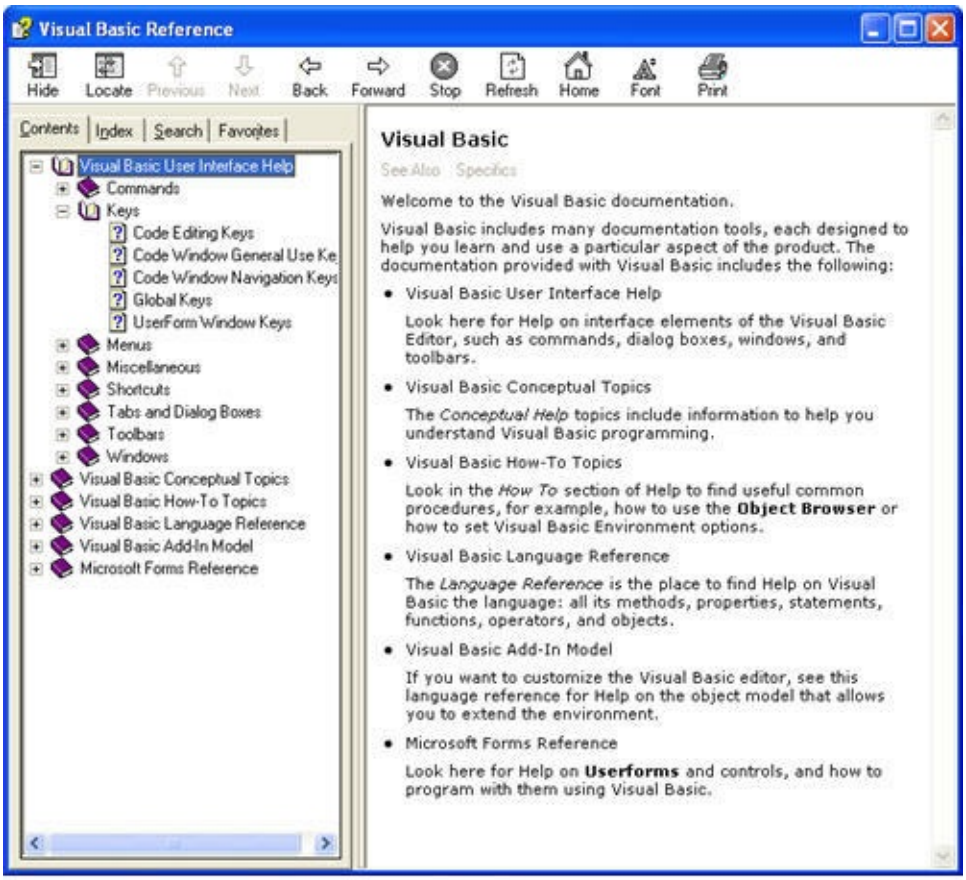
VBA

Visual Basic EditorVBA

- **Help　　Microsoft Visual Basic HelpMicrosoft Visual Basic**

- **F1**
- "VBA"

VBA



e5071c135

VBA**Contents**　　E5071C VBA

-

- Visual Basic
- Visual Basic
- Visual Basic
- Visual Basic
-

Visual Basic EditorVisual BasicVBAVisual BasicVisual Basic VBA

“VBA” **Index** “Sub”“With”

Visual Basic Editor

F1

E5071C

-
-
-

-
-
- SCPI.TRIGger.SEQuence.SINGle

E5071C

/

SCPI.TRIGger.SEQuence.SINGle

E5071CSPCISRQ

COMSRQSRQ

- WaitOnSRQ

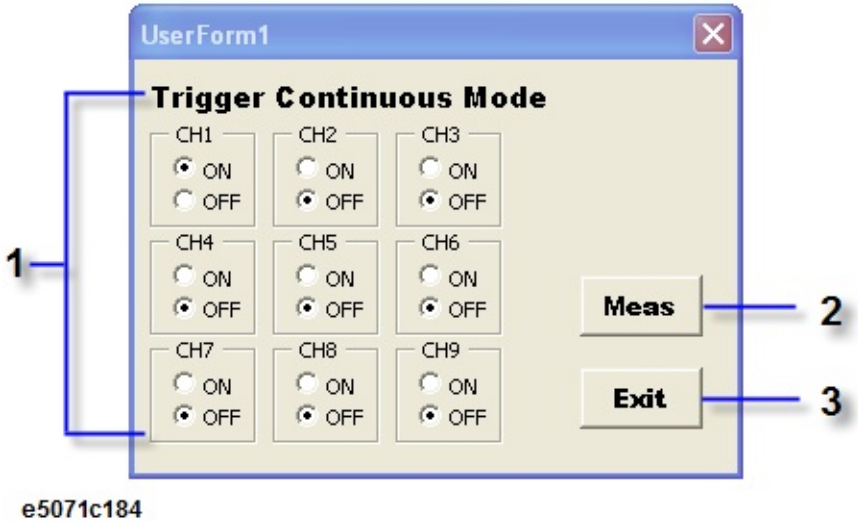
meas_srq.vbaVBA

frmSrqMeas	UserForm	
mdlSrqMeas		UserForm.

/9/9

VBAUserForm

meas_srq.vbaUserForm



1.

2.
3. UserForm

Visual Basic EditorUserFormfrmSrqMeas **Meas** **Exit**

Exit

30

UserForm

UserForm70390 **Meas**

110

UserFormfrmSrqMeas

130

9

140

“bus”

160240

1

260270

bit 4bit 4101

280

bit 4

290

bit 7

300

310

320

100

330350

370

UserFormfrmSrqMeas

SRQfrmSrqMeas

```
10| Private Sub cmdExit_Click()
20|
30| Unload Me
40|
50| End Sub
60|
70| Private Sub cmdMeas_Click()
80|
90| Dim Cond As Boolean
100|
110| frmSrqMeas.Hide
120|
130| SCPI.DISPlay.Split = "d123_456_789"
140| SCPI.TRIGger.SEQuence.Source = "bus"
```

```
150|
160| SCPI.INITiate(1).CONTinuous = optOn1.Value
170| SCPI.INITiate(2).CONTinuous = optOn2.Value
180| SCPI.INITiate(3).CONTinuous = optOn3.Value
190| SCPI.INITiate(4).CONTinuous = optOn4.Value
200| SCPI.INITiate(5).CONTinuous = optOn5.Value
210| SCPI.INITiate(6).CONTinuous = optOn6.Value
220| SCPI.INITiate(7).CONTinuous = optOn7.Value
230| SCPI.INITiate(8).CONTinuous = optOn8.Value
240| SCPI.INITiate(9).CONTinuous = optOn9.Value
250|
260| SCPI.STATus.OPERation.PTRansition = 0
270| SCPI.STATus.OPERation.NTRansition = 16
280| SCPI.STATus.OPERation.ENABle = 16
290| SCPI.IEEE4882.SRE = 128
300| SCPI.IEEE4882.CLS
310| SCPI.IEEE4882.TRG
320| WaitOnSRQ Cond, 100000
330| If Cond = True Then
340| MsgBox "Measurement Completion"
350| End If
360|
370| frmSrqMeas.Show
380|
```

```
390| End Sub
```

SCPI.TRIGger.SEQuence.SINGle

SCPI.TRIGger.SEQuence.SINGle SCPI.IEEE4882.OPC

“meas_srq.vba” SCPI.TRIGger.SEQuence.SINGleVBA

frmSingMeas	UserForm	SCPI.TRIGger.SEQuence.SINGleSCPI.IEE
mdlSingMeas		

/9/9

VBA Userform

Visual Basic EditorUserFormfrmSrqMeas **Meas**
Exit

UserForm1050 **Exit**

30

UserForm

UserForm70330Meas

110

UserFormfrmSrqMeas

130

9

140

“bus”

160240

1

260

270

SCPI.IEEE4882.OPC1

290

310

UserFormfrmSrqMeas

SCPI.TRIGger.SEQuence.SINGlefrmSingMeas

10| Private Sub cmdExit_Click()

20|

30| Unload Me

40|

50| End Sub

60|

70| Private Sub cmdMeas_Click()

80|

90| Dim Dmy As Long

100|

110| frmSingMeas.Hide

120|

130| SCPI.DISPlay.Split = "d123_456_789"

140| SCPI.TRIGger.SEQuence.Source = "bus"

```
150|
160| SCPI.INITiate(1).CONTinuous = optOn1.Value
170| SCPI.INITiate(2).CONTinuous = optOn2.Value
180| SCPI.INITiate(3).CONTinuous = optOn3.Value
190| SCPI.INITiate(4).CONTinuous = optOn4.Value
200| SCPI.INITiate(5).CONTinuous = optOn5.Value
210| SCPI.INITiate(6).CONTinuous = optOn6.Value
220| SCPI.INITiate(7).CONTinuous = optOn7.Value
230| SCPI.INITiate(8).CONTinuous = optOn8.Value
240| SCPI.INITiate(9).CONTinuous = optOn9.Value
250|
260| SCPI.TRIGger.SEQuence.SINGle
270| Dmy = SCPI.IEEE4882.OPC
280|
290| MsgBox "Measurement Completion"
300|
310| frmSingMeas.Show
320|
330| End Sub
```

-

-
-

E5071C

E5071C Internal Data Processing

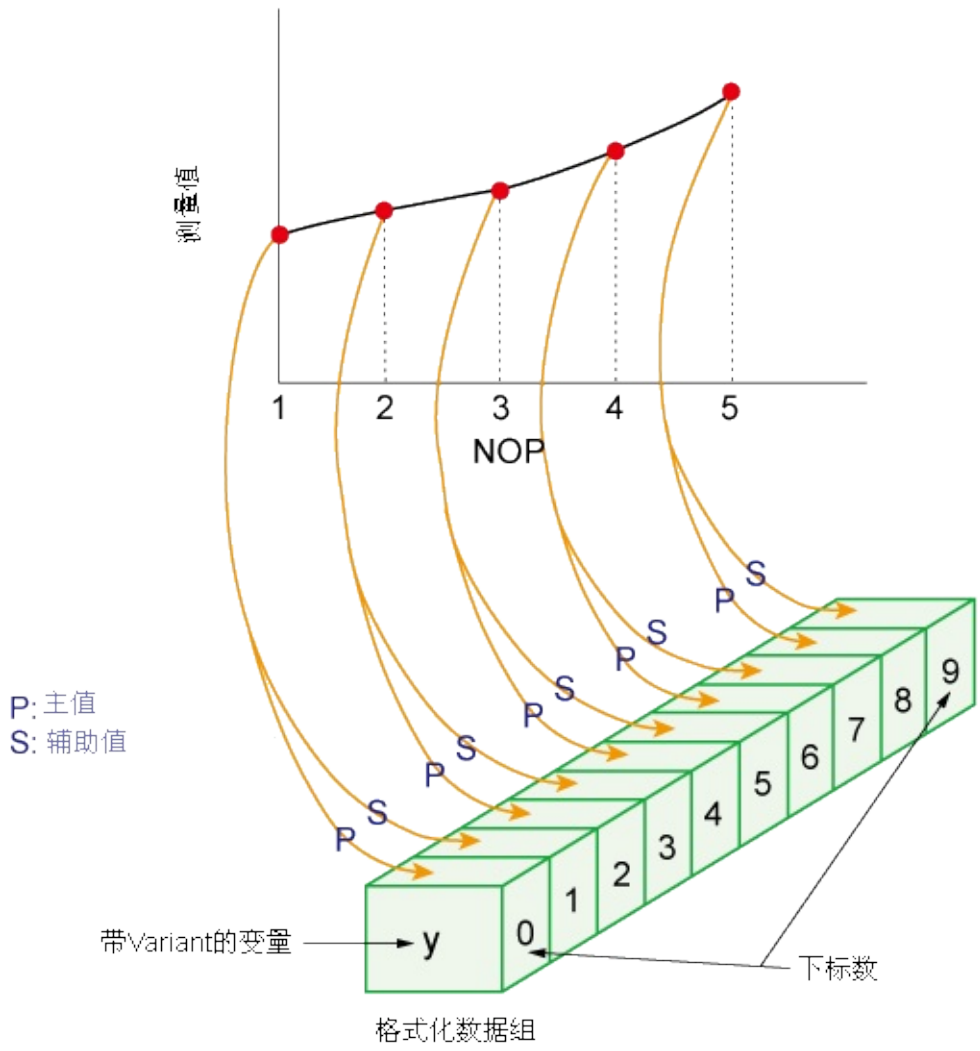
-

- SCPI.CALCulate(Ch).SELected.DATA.FDATa
- SCPI.CALCulate(Ch).SELected.DATA.FMEMory
- SCPI.CALCulate(Ch).SELected.DATA.SDATa
- SCPI.CALCulate(Ch).SELected.DATA.SMEMory

- SCPI.SENSe(Ch).FREQuency.DATA

E5071C VBAVariantVariant52
Internal Data Processing

Variant



e5071c436

0"Option

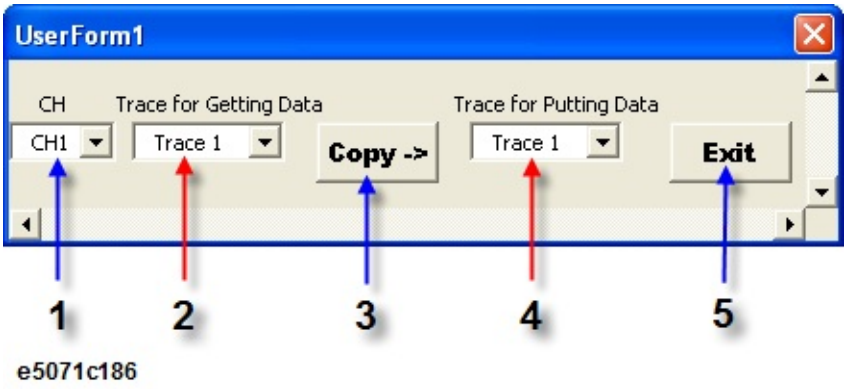
"read_write.vba"VBA

frmReadWrite	UserForm	
mdlReadWrite		UserForm.

/9/9

VBA

read_write.vbaUserForm



1.
2.
3.
4.

Visual Basic EditorUserFormfrmReadWriteUserForm **Copy ->Exit**

UserForm**Copy**10520

90160

TrGetTrPutActCh

180210

230250

(ActCh)TrGet:

260

(ActCh)Nop

280

FmtData

290

(ActCh)Freq

330

Fmt

340350

LCD

360470

Fmt"MLOG""PHAS""GDEL""MLIN""SWR""REAL""IMAG"
"UPH"0Fmt$

490

(ActCh)TrPut:

500

(FmtData)

UserForm**Exit**540580

560

UserForm

UserForm6001020

6201000

//(read_write.frm)

```
10| Private Sub cmdCopy_Click()
20|
30| Dim X As Integer, Y As Integer, Z As Integer, I As Integer
40| Dim ActCh As Long, TrGet As Long, TrPut As Long
50| Dim TrCont As Long, Nop As Long
60| Dim FmtData As Variant, Freq As Variant
70| Dim Fmt As String
80|
90| X = cboCh.ListIndex
100| ActCh = X + 1
110|
120| Y = cboGet.ListIndex
130| TrGet = Y + 1
140|
150| Z = cboPut.ListIndex
160| TrPut = Z + 1
170|
180| TrCont = SCPI.CALCulate(ActCh).PARameter.Count
190| If TrCont < TrPut Then
200| SCPI.CALCulate(ActCh).PARameter.Count = TrPut
210| End If
220|
```

```
230| SCPI.CALCulate(ActCh).PARameter(TrGet).SELect
240| SCPI.INITiate(ActCh).CONTinuous = False
250| SCPI.ABORt
260| Nop = SCPI.SENSe(ActCh).SWEep.POINts
270|
280| FmtData = SCPI.CALCulate(ActCh).SELected.Data.FDATa
290| Freq = SCPI.SENSe(ActCh).FREQuency.Data
300|
310| '''Displays the formatted data
320|
330| Fmt = SCPI.CALCulate(ActCh).SELected.Format
340| SCPI.DISPlay.TABLe.TYPE = "ECHO"
350| SCPI.DISPlay.TABLe.STATe = True
360| Select Case Fmt
370| Case "MLOG", "PHAS", "GDEL", "MLIN", "SWR", "REAL", "IMAG",
"UPH"
380| ECHO "Nop", "Frequency(GHz)", "Data"
390| For I = 0 To Nop - 1
400| ECHO I + 1, Freq(I) / 1000000000#, FmtData(2 * I)
410| Next I
420| Case Else
430| ECHO "Nop", "Frequency(GHz)", "Data1", "Data2"
440| For I = 0 To Nop - 1
450| ECHO I + 1, Freq(I) / 1000000000#, FmtData(2 * I), FmtData(2 * I +
1)
```

```
460| Next I
470| End Select
480|
490| SCPI.CALCulate(ActCh).PARameter(TrPut).SELect
500| SCPI.CALCulate(ActCh).SELected.Data.FDATa = FmtData
510|
520| End Sub
530|
540| Private Sub cmdExit_Click()
550|
560| Unload Me
570|
580| End Sub
590|
600| Private Sub UserForm_Initialize()
610|
620| With cboCh
630| .AddItem "CH1"
640| .AddItem "CH2"
650| .AddItem "CH3"
660| .AddItem "CH4"
670| .AddItem "CH5"
680| .AddItem "CH6"
690| .AddItem "CH7"
```

```
700| .AddItem "CH8"
710| .AddItem "CH9"
720| End With
730|
740| With cboGet
750| .AddItem "Trace 1"
760| .AddItem "Trace 2"
770| .AddItem "Trace 3"
780| .AddItem "Trace 4"
790| .AddItem "Trace 5"
800| .AddItem "Trace 6"
810| .AddItem "Trace 7"
820| .AddItem "Trace 8"
830| .AddItem "Trace 9"
840| End With
850|
860| With cboPut
870| .AddItem "Trace 1"
880| .AddItem "Trace 2"
890| .AddItem "Trace 3"
900| .AddItem "Trace 4"
910| .AddItem "Trace 5"
920| .AddItem "Trace 6"
930| .AddItem "Trace 7"
```

```
940| .AddItem "Trace 8"
950| .AddItem "Trace 9"
960| End With
970|
980| cboCh.ListIndex = 0
990| cboGet.ListIndex = 0
1000| cboPut.ListIndex = 0
1010|
1020| End Sub
```

-
-
-
-

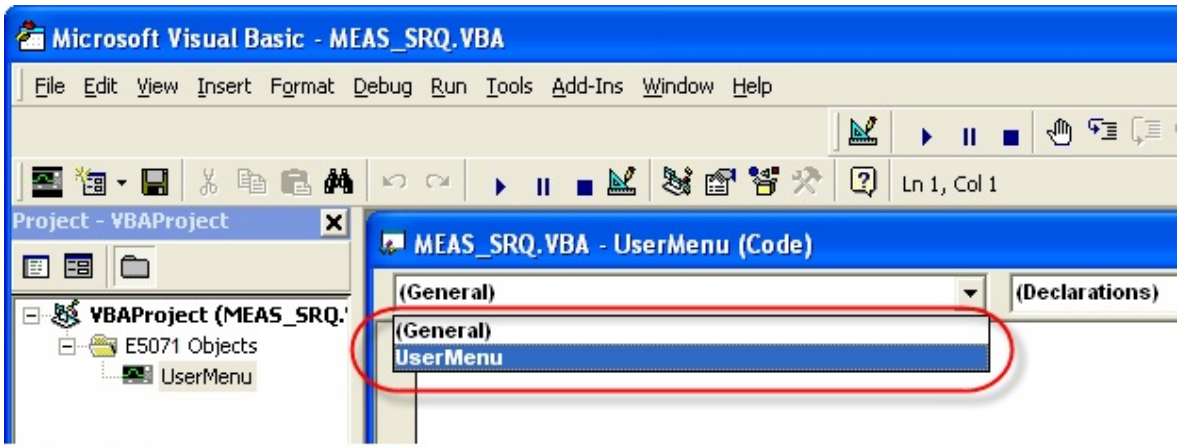
E5071C

E5071C **Macro Setup** > **User Menu()** > **1/2/3/4/5/6/7/8/9/10**

VBA

“UserMenu”“E5071”

1. UserMenu
2. UserMenu



e5071c185

3. UserMenu_OnPressByVal Key_id As Longid70
   430

“UerMenu”

COM

- UserMenu.Item(Key_id).Caption

COM/

- UserMenu.Item(Key_id).Enabled

COM

- UserMenu.PRESet

E5071C**Macro Setup** > **Preset User Menu**

COM

Macro Setup > **User Menu** > **Button <No>.**

Macro Setup > **User Menu** > **Button <No><>**

“No.” “Button No.”“/”

COM

COM

- UserMenu.Press(Key_id)

meas_user.vbaVBA" UserMenu"

mdlUserMenu		

mdlUserMenu

70

True

90150

id13id410

170190

id:1“Setup”id:2“Meas”id:3“Exit”

210

230250

True=True

240

mdlUserMenu)

10| Public State As Boolean

20|

30| Sub Main()

40|

```
50| Dim I As Long, J As Long
60|
70| State = True
80|
90| For I = 1 To 3
100| UserMenu.Item(I).Enabled = True
110| Next I
120|
130| For J = 4 To 10
140| UserMenu.Item(J).Enabled = False
150| Next J
160|
170| UserMenu.Item(1).Caption = "Setup"
180| UserMenu.Item(2).Caption = "Meas"
190| UserMenu.Item(3).Caption = "Exit"
200|
210| UserMenu.Show
220|
230| Do While State
240| DoEvents
250| Loop
260|
270| End Sub
```

UserMenu

“UserMenu”

70190

id: 1

90E5071C

11013011.73 GHz1.83 GHz51

1501701

190

210320

id: 2

230240 SCPI.IEEE4882.OPC1

2601Nop

28029011FmtData

310320LCD

3403602

380430

id: 3

400

410Falsesta

“UserMenu”

10| Private Sub UserMenu_OnPress(ByVal id As Long)

20|

30| Dim I As Integer

40| Dim Nop As Long, Dmy As Long

```
50| Dim FmtData As Variant
60|
70| If id = 1 Then
80|
90| SCPI.SYSTem.PRESet
100|
110| SCPI.SENSe(1).FREQuency.STARt = 1730000000#
120| SCPI.SENSe(1).FREQuency.STOP = 1830000000#
130| SCPI.SENSe(1).SWEep.POINts = 51
140|
150| SCPI.ABORt
160| SCPI.TRIGger.SEQuence.Source = "BUS"
170| SCPI.INITiate(1).CONTinuous = True
180|
190| UserMenu.Show
200|
210| ElseIf id = 2 Then
220|
230| SCPI.TRIGger.SEQuence.SINGle
240| Dmy = SCPI.IEEE4882.OPC
250|
260| Nop = SCPI.SENSe(1).SWEep.POINts
270|
280| SCPI.CALCulate(1).PARameter(1).SELect
```

```
290| FmtData = SCPI.CALCulate(1).SELected.DATA.FDATa
300|
310| SCPI.DISPlay.TABLe.TYPE = "ECHO"
320| SCPI.DISPlay.TABLe.State = True
330|
340| For I = 1 To Nop - 1
350| ECHO FmtData(2 * I - 2), FmtData(2 * I - 1)
360| Next I
370|
380| ElseIf id = 3 Then
390|
400| MsgBox "Program ended!"
410| State = False
420|
430| End If
440|
450| End Sub
```

-
- VISA

-
-

E5071CE5071C VBAUSB/GPIB

E5071CE5071C VBAE5071CCOMVISA

E5071CE5071C VBAVISAVisual Basic editor2VISA

- **visa32.bas**
- **vpptype.bas**

VISA

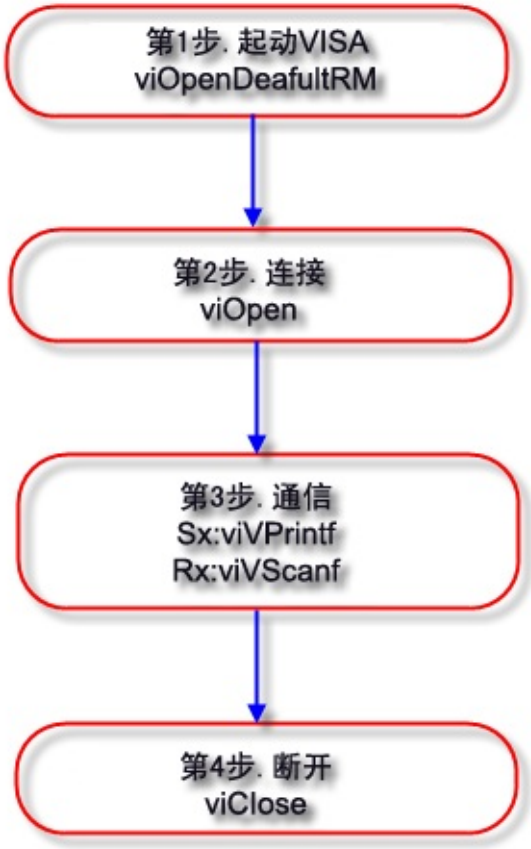
-
- VISA
-
-
-
-

VISABasicVisual Basic languageeVISA

VISAE5071AE5071A VBAVISAIOCD-ROM

- **visa.hlp** VISA
- **vbreadme.txt** VBVISA

VISA



e5071c183

1VISA

ctrl_ext.vba. 90VISAVISAviOpenDefaultRMVISA viOpenDefaultRM VISA

viOpenDefaultRM(*param*)

	(*param*)

2

130VISA viOpen viOpenVISADefrm **ctrl_ext.vba** “GPIB17INSTR” **ctrl_ext.vba**0 **ctrl_ext.vba**0 **ctrl_** Equip

viOpen(*param1*, *param2*, *param3*, *param4*, *param5*)

	(*param1*)

	(*param2*)
	"GPIB0::*gpib address*::INSTR" "USB0::*manufacturer ID*::*model code*::*serial number*::0::INSTR" (ex. "USB0::2391::2312::MY12345678::0::INSTR") "TCPIP0::*IP address*::inst0::INSTR"

	(*param3*)
	0

	(*param4*)
	0

	(*param5*)

	(*param5*)

3.

170VISA viVPrintfGPIBEquip*EDN? **ctrl_ext.vba**0

/GPIB viVPrintf viVScanfVISA

viVPrintf(*param1*, *param2*, *param3*)

	(*param1*)

	(*param2*)
	GPIB **ctrl_ext.vba**Chr$10

	(*param3*)
	0

210VISA viVScanf **ctrl_ext.vba**Equip **ctrl_ext.vba**%t **ctrl_ext.vba**Prod

viVScanf(*param1*, *param2*, *param3*)

	(*param1*)

	(*param2*)

	(*param3*)

4

280VISA viClose VISA **ctrl_ext.vba**Defrm

viClose(*param*)

Parameter

	(*param*)

ctrl_ext.vba E5071CUSB/GPIBVBA

mdlVisa		
1 2		VISA

E5071C VBAGPIBVISAE5071C

90100

VISADefrm320360

130140

GPIBGPIB17Equip320360

170180

VISAUSB/GPIB320360

210250

VISAProd320360

280

VISA

320360

VISA

ctrl_ext.vba

10| Sub Main()

20|

```
30| Dim status As Long 'VISA function status return code
40| Dim Defrm As Long 'Session to Default Resource Manager
50| Dim Equip As Long 'Session to instrument
60| Dim Prod As String * 100 'String to receive the result
70|
80| ' Initializes the VISA system.
90| status = viOpenDefaultRM(Defrm)
100| If (status <> VI_SUCCESS) Then GoTo VisaErrorHandler
110|
120| ' Opens the session to the specified instrument.
130| status = viOpen(Defrm, "GPIB0::17::INSTR", 0, 0, Equip)
140| If (status <> VI_SUCCESS) Then GoTo VisaErrorHandler
150|
160| ' Asks for the instrument's product information.
170| status = viVPrintf(Equip, "*IDN?" & Chr$(10), 0)
180| If (status <> VI_SUCCESS) Then GoTo VisaErrorHandler
190|
200| ' Reads the result.
210| status = viVScanf(Equip, "%t", Prod)
220| If (status <> VI_SUCCESS) Then GoTo VisaErrorHandler
230|
240| ' Displays the result.
250| MsgBox Prod
260|
```

```
270| ' Closes the resource manager session (which closes everything)
280| Call viClose(Defrm)
290|
300| GoTo Prog_end
310|
320| VisaErrorHandler:
330| Dim VisaErr As String * 200
340| Call viStatusDesc(Defrm, status, VisaErr)
350| MsgBox "Error : " & VisaErr, vbExclamation
360| Exit Sub
370|
380| Prog_end:
390|
400| End Sub
```

- [_____]
- [.]
- [E5091AFEM____]
- [.]
- [PC_____]

-
-
-

apl_bsc.vbaVBAVBA

mdlBscMeas		

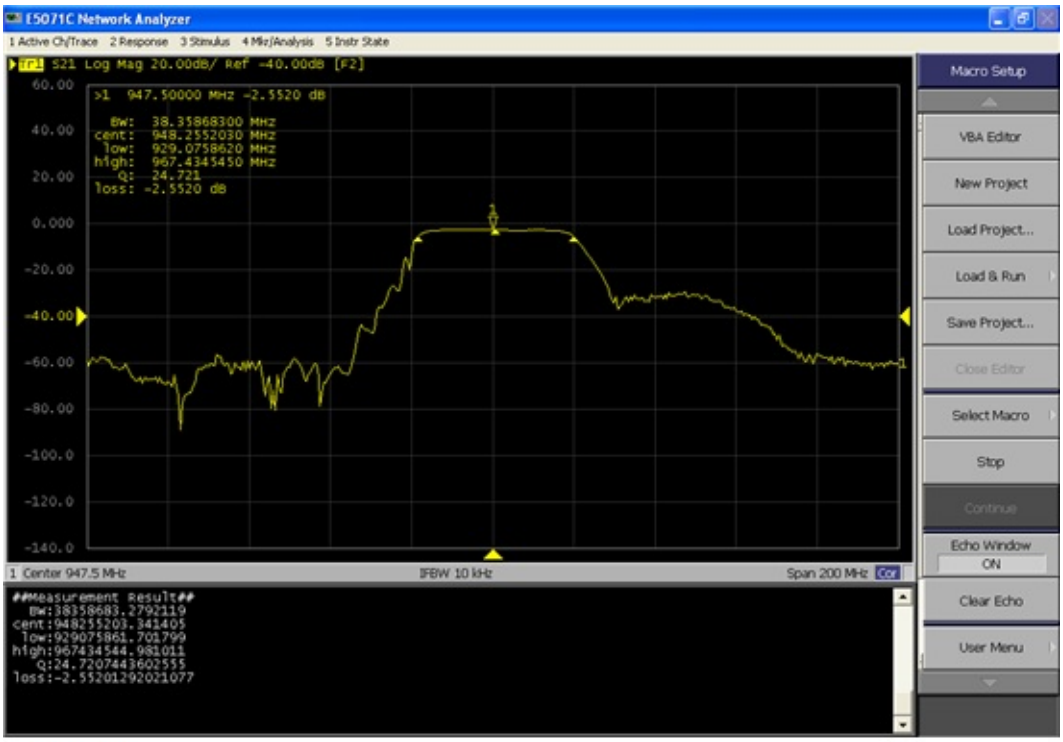
85032F947.5 MHz

VBA“” **YesNo**

Agilent 85032F **OK** **Cancel** “”

“ **Macro Setup** > **Continue** ”LCD **Macro Setup** > **Continue**

"apl_bsc.vba"



e5071c188

mdlBscMeas

120160

947.5 MHz200 MHz40110 kHz-10 dBmCenter, Span, Nop, IfBwPow

170210

(1)(S21)(4: 85032F)(State08.sta)NumTrac, File

250

E5071C

290300

1

320360

1CenterSpanNopIfBwPow

380410

1NumTracParFmt

450

1CalKit

460

12port

480

Calib_Solt12002130

520530

File

580

LCDDUT **Macro Setup** > **Continue**

620630

1 SCPI.IEEE4882.OPC

650

11

690710

11MkrVal

730

Bw_Err890950

750770

-3 dBQBwData

790840

BwCentQQfacLossSkip_Bw_Err

880960

0BwCentQfac1MkrVal(0)Loss

10001010

BwCent2CutLowCutHigh

10301110

Bw, Cent, CutLow, CutHigh, QfacLoss

11301160

Yes **No**

Calib_Solt (12002130).

12601300

nSoltType **cancel**

13201410

n

14501520

OK **Cancel**

15401610

OK **Cancel**

16301700

OK **Cancel**

17501840

OK **Cancel**

18802060

1SoltType1 **Yes** I-1J-1 **OK** **Cancel**

20802090

mdlBscMeas

10| Sub Main()

20|

```
30| Dim Par As String, Fmt As String, File As String

40| Dim Center As Double, Span As Double, IfBw As Double, Pow As Double

50| Dim Bw As Double, Cent As Double

60| Dim CutLow As Double, CutHigh As Double

70| Dim Qfac As Double, Loss As Double

80| Dim MkrVal As Variant, BwData As Variant

90| Dim Nop As Long, NumTrac As Long, CalKit As Long, Buff As Long

100| Dim Port As Variant, Error As Variant

110|

120| Center = 947500000# 'Center freq : 947.5 MHz

130| Span = 200000000# 'Span freq : 200 MHz

140| Nop = 401 'Number of points : 401

150| IfBw = 10000# 'IF bandwidth : 10 kHz

160| Pow = -10 'Power level : -10dBm

170| NumTrac = 1 'Number of traces : 1

180| Par = "S21" 'Meas. parameter : S21

190| Fmt = "MLOG" 'Data format : Log Mag

200| CalKit = 4 'Calibration kit : 85032F

210| File = "State08.sta" 'Saved file name : State08.sta

220|

230| "'Presetting the E5071C

240|

250| SCPI.SYSTem.PRESet

260|
```

```
270| '''Setting measurement conditions
280|
290| SCPI.INITiate(1).CONTinuous = True
300| SCPI.TRIGger.SEQuence.Source = "BUS"
310|
320| SCPI.SENSe(1).FREQuency.Center = Center
330| SCPI.SENSe(1).FREQuency.Span = Span
340| SCPI.SENSe(1).SWEep.POINts = Nop
350| SCPI.SENSe(1).BANDwidth.RESolution = IfBw
360| SCPI.Source(1).POWer.LEVel.IMMediate.AMPLitude = Pow
370|
380| SCPI.CALCulate(1).PARameter.Count = NumTrac
390| SCPI.CALCulate(1).PARameter(1).DEFine = Par
400| SCPI.CALCulate(1).PARameter(1).Select
410| SCPI.CALCulate(1).SELected.Format = Fmt
420|
430| '''Performing full 2-port calibration
440|
450| SCPI.SENSe(1).CORRection.COLLect.CKIT.Select = CalKit
460| Port = Array(1, 2)
470|
480| Calib_Solt 1, 2, Port
490|
500| '''Saving state & cal data
```

```
510|

520| SCPI.MMEMory.STORe.STYPe = "CST"

530| SCPI.MMEMory.STORe.STATe = File

540|

550| '''Connecting DUT

560|

570| Meas_Start:

580| Prompt ("Connect DUT, and then press [Macro Setup]-Continue button.")

590|

600| '''Performing single sweep

610|

620| SCPI.TRIGger.SEQuence.SINGle

630| Dmy = SCPI.IEEE4882.OPC

640|

650| SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.AUTO

660|

670| '''Analyzing the results

680|

690| SCPI.CALCulate(1).SELected.MARKer(1).STATe = True

700| SCPI.CALCulate(1).SELected.MARKer(1).X = Center

710| MkrVal = SCPI.CALCulate(1).SELected.MARKer(1).Y

720|

730| On Error GoTo Bw_Err

740|
```

```
750| SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.THReshold = -3
760| SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.STATe = True
770| BwData = SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.DATA
780|
790| Bw = BwData(0)
800| Cent = BwData(1)
810| Qfac = BwData(2)
820| Loss = BwData(3)
830|
840| GoTo Skip_Bw_Err
850|
860| Bw_Err:
870|
880| Error = SCPI.SYSTem.Error
890| MsgBox "Error No:" & Error(0) & " , Description:" & Error( 1)
900|
910| Bw = 0
920| Cent = 0
930| Qfac = 0
940| Loss = MkrVal(0)
950|
960| Resume Skip_Bw_Err
970|
980| Skip_Bw_Err:
```

```
990|
1000| CutLow = Cent - Bw / 2
1010| CutHigh = Cent + Bw / 2
1020|
1030| ECHO "##Measurement Result##"
1040| ECHO " BW:" & Bw
1050| ECHO "cent:" & Cent
1060| ECHO " low:" & CutLow
1070| ECHO "high:" & CutHigh
1080| ECHO " Q:" & Qfac
1090| ECHO "loss:" & Loss
1100| SCPI.DISPlay.TABLe.TYPE = "ECHO"
1110| SCPI.DISPlay.TABLe.STATe = True
1120|
1130| Buff = MsgBox("Do you make another measurement?", vbYesNo,
"Bandpass fileter measurement")
1140| If Buff = vbYes Then
1150| GoTo Meas_Start
1160| End If
1170|
1180| End Sub
1190|
1200| Private Sub Calib_Solt(Chan As Long, SoltType As Long, Port As
Variant)
1210|
```

```
1220| Dim Dmy As Long, I As Long, J As Long, Buff As Long

1230|

1240| Cal_Start:

1250|

1260| Buff = MsgBox("Perform the full " & SoltType & "-port cali bration.",
vbOKCancel, "Full" & SoltType & "-port calibration")

1270|

1280| If Buff = vbCancel Then

1290| GoTo Cal_Skip

1300| End If

1310|

1320| Select Case SoltType

1330| Case 1

1340| SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT1 =
Port(0)

1350| Case 2

1360| SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT2 = Port

1370| Case 3

1380| SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT3 = Port

1390| Case 4

1400| SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT4 = Port

1410| End Select

1420|

1430| For I = 1 To SoltType

1440|
```

```
1450| Buff = MsgBox("Connect the Open standard to Port " & CS tr(Port(I - 1)) & ".", _

1460| vbOKCancel, "Full" & SoltType & "-port calibration")

1470| If Buff = vbOK Then

1480| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.OPEN = Port(I - 1)

1490| Dmy = SCPI.IEEE4882.OPC

1500| Else

1510| GoTo Cal_Start

1520| End If

1530|

1540| Buff = MsgBox("Connect the Short standard to Port " & CStr(Port(I - 1)) & ".", _

1550| vbOKCancel, "Full" & SoltType & "-port calibration")

1560| If Buff = vbOK Then

1570| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.Short = Port(I - 1)

1580| Dmy = SCPI.IEEE4882.OPC

1590| Else

1600| GoTo Cal_Start

1610| End If

1620|

1630| Buff = MsgBox("Connect the Load standard to Port " & CStr(Port(I - 1)) & ".", _

1640| vbOKCancel, "Full" & SoltType & "-port calibration")

1650| If Buff = vbOK Then
```

```
1660| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.Load = Port(I
- 1)

1670| Dmy = SCPI.IEEE4882.OPC

1680| Else

1690| GoTo Cal_Start

1700| End If

1710| Next I

1720|

1730| For I = 1 To SoltType - 1

1740| For J = I + 1 To SoltType

1750| Buff = MsgBox("Connect the Thru standard between Por t " &
CStr(Port(I - 1)) & _

1760| " and Port " & CStr(Port(J - 1)) & ".", vbOKCancel, "Full" & SoltType
& "-port calibration")

1770| If Buff = vbOK Then

1780| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.THRU =
Array(Port(I - 1 ), Port(J - 1))

1790| Dmy = SCPI.IEEE4882.OPC

1800| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.THRU =
Array(Port(J - 1 ), Port(I - 1))

1810| Dmy = SCPI.IEEE4882.OPC

1820| Else

1830| GoTo Cal_Start

1840| End If

1850| Next J

1860| Next I
```

```
1870|

1880| If SoltType <> 1 Then

1890| Buff = MsgBox("Do you measure the Isolation (Optional) ?", vbYesNo, "Full" & SoltType & "-port calibration")

1900| If Buff = vbYes Then

1910| For I = 1 To SoltType - 1

1920| For J = I + 1 To SoltType

1930| Buff = MsgBox("Connect the Load standard to Port " & Port(I - 1) & " and Port " & Port(J - 1) & ".", _

1940| vbOKCancel, "Full" & Solt Type & "-port calibration")

1950| If Buff = vbOK Then

1960| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.ISOLation = Array(Port( I - 1), Port(J - 1))

1970| Dmy = SCPI.IEEE4882.OPC

1980| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.ISOLation = Array(Port( J - 1), Port(I - 1))

1990| Dmy = SCPI.IEEE4882.OPC

2000| Else

2010| GoTo Cal_Start

2020| End If

2030| Next J

2040| Next I

2050| End If

2060| End If

2070|

2080| SCPI.SENSe(1).CORRection.COLLect.SAVE
```

```
2090| MsgBox "All calibration data completion."
2100|
2110| Cal_Skip:
2120|
2130| End Sub
```

-
-
-
-

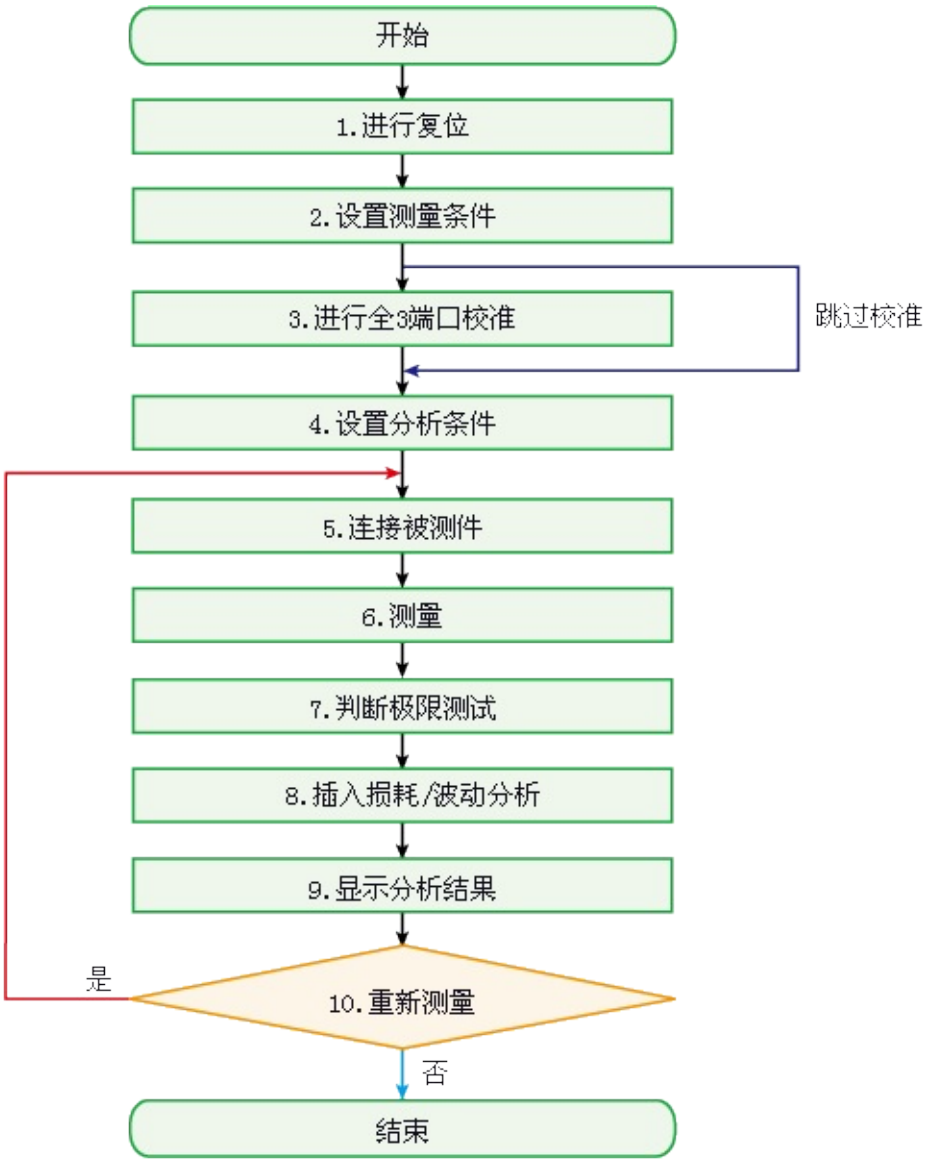
= 4 && typeof(BSPSPopupOnMouseOver) == 'function') BSPSPopupOnMouseOver(event)" class=BSSCPopup onclick="BSSCPopup('Application_Programs.htm',400,300);return false">

apl_sys.vba 3VBAVBA

2SE5071CE5071C

frmDupRes		
mdlDupMeas		

85032F333



e5071c435

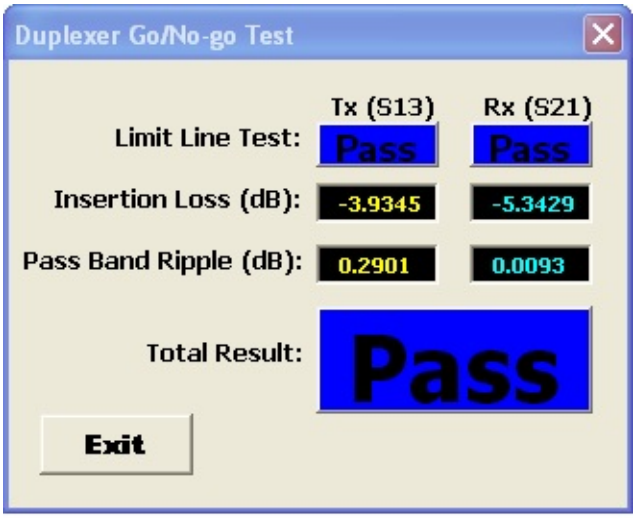
VBA"Perform the full 3-port
Yes **No**

Agilent 85032FOK **Cancel()** "All
data completion" “”

"Set the DUT between test cables."“” **Yes()**
1Tx: S13Rx: S21/

Exit() Yes()No()frmDupRes

“apl_sys.vba”



e5071c187

mdlDupMeas

90

4: 85032FCalKit()

110

140

E5071C

170180

1

200

Setup_Parameter9101140Setup_Parameter

220

Setup_Segment11601530Setup_Segment

250

1CalKit()

260

123Port()3

280

Calib_Solt

310

Setup_Limitline24203180Setup_Limitline

330

Setup_Register32003260Setup_Register

380

“DUT”OK

410

1

420430

= 4 && typeof(BSPSPopupOnMouseOver) == 'function') BSPSPopupOnMouseOver(event)" class=BSSCPopup onclick="BSSCPopup('../../Command_Reference/IEEE/SCPI_IEEE4882_(false">SCPI.IEEE4882.OPC1

450460

12

490500

AND211Test_Ch1

510530

1AND211Test_Tr1AND421Test_Tr2

560610

1(1.85 GHz1.91 GHz)()(0)

620640

IlossTx

650

RipTx

670720

2(1.93 GHz1.99 GHz)()(0)

730750

IlossRx

760

RipRx

790

Display_Update(32803620)Display_Update

810

(frmDupRes

830870

Yes() (DUT) **No()**

: Setup_Parameter (9101140)

9701020

(“D1”1)(“D1_2”/)1(S13)2(S21)1(MLOG)2(MLOG)ChDisp
TracDispPar(0)Par(1)Fmt(0)Fmt(1)

10401060

1~2ChDisp1TracDisp

10801120

1Par(0)1Fmt(0)2Par(1)2Fmt(1)

Setup_Segment (11601530)

12001260

SegmData(0)1SegmData(6)/(IF)
5

12801470

SegmData(7)11~5SegmData(21)

1490

1SegmData

1500

1“segment”

1510

1()

Calib_Solt (15502420)

Calib_Solt12002130

Setup_Limitline (24403180)

2490

1(5)LimDataS13(0)

25002790

LimDataS13(1)1LimDataS13(25)

2820

2(4)LimDataS21(0)

28303060

LimDataS21(1)2LimDataS21(20)

30803110

11LimDataS131

31303160

22LimDataS212

Setup_Register (32003260)

32203230

1201()121

3240

12

Display_Update (32803620)

3300

LCD

33203380

1(Test_Tr1=2)(frmDupRes)Tx(S13)“Limit
1(Test_Tr11 ?2)(frmDupRes)Tx(S13)“Limit
)

34003460

2(Test_Tr2 = 4)(frmDupResRx(S21)
)1(Test_Tr2 ? 4)(frmDupRes) Rx(S21)
Pass" ()

34803540

1(Test_Ch1 = 2)(frmDupRes)"Overall
)1 (Test_Ch1 ? 2)(frmDupRes) "Overall
result: Pass" ()

35603600

(frmDupRes)12()Tx(S13)Rx(S21)

(mdlDupMeas)

```
10| Sub Main()
20|
30| Dim CalKit As Long, Dmy As Long, Rgst As Long, I As Long, Buff As Long
40| Dim Test_Ch1 As Integer, Test_Tr1 As Integer, Test_Tr2 As Integer
50| Dim IlossTx As Variant, IlossRx As Variant
60| Dim RipTx As Double, RipRx As Double
70| Dim Port As Variant
80|
90| CalKit = 4 'Calibration kit :85032F
100|
110| SCPI.DISPlay.ENABle = False
120|
130| '''Presetting the E5071C
140| SCPI.SYSTem.PRESet
150|
160| '''Setting measurement conditions
170| SCPI.INITiate(1).CONTinuous = True
180| SCPI.TRIGger.SEQuence.Source = "BUS"
190|
200| Setup_Parameter
210|
```

```
220| Setup_Segment
230|
240| '''Full 3-port calibration
250| SCPI.SENSe(1).CORRection.COLLect.CKIT.Select = CalKit
260| Port = Array(1, 2, 3)
270|
280| Calib_Solt 1, 3, Port
290|
300| '''Setting analysis conditions
310| Setup_Limitline
320|
330| Setup_Register
340|
350| Meas_Start:
360|
370| '''Connecting DUT
380| MsgBox "Connect DUT between test cables."
390|
400| '''Performing single sweep
410| SCPI.IEEE4882.CLS
420| SCPI.TRIGger.SEQuence.SINGle
430| Dmy = SCPI.IEEE4882.OPC
440|
450| SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.AUTO
```

```
460| SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.AUTO

470|

480| '''Judging limit test

490| Rgst = SCPI.STATus.QUEStionable.LIMit.EVENt

500| Test_Ch1 = CInt(Rgst And 2)

510| Rgst = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).EVENt

520| Test_Tr1 = CInt(Rgst And 2)

530| Test_Tr2 = CInt(Rgst And 4)

540|

550| '''Analyzing insertion loss/bandpass ripple

560| SCPI.CALCulate(1).PARameter(1).Select

570| SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt =
1850000000#

580| SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP =
1910000000#

590| SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STATe = True

600| SCPI.CALCulate(1).SELected.FUNCtion.PPOLarity = "both"

610| SCPI.CALCulate(1).SELected.FUNCtion.PEXCursion = 0

620| SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "MIN"

630| SCPI.CALCulate(1).SELected.FUNCtion.EXECute

640| IlossTx = SCPI.CALCulate(1).SELected.FUNCtion.DATA

650| RipTx = MaxPeakToPeak(1)

660|

670| SCPI.CALCulate(1).PARameter(2).Select

680| SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt =
```

```
1930000000#

690| SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP = 1990000000#

700| SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STATe = True

710| SCPI.CALCulate(1).SELected.FUNCtion.PPOLarity = "both"

720| SCPI.CALCulate(1).SELected.FUNCtion.PEXCursion = 0

730| SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "MIN"

740| SCPI.CALCulate(1).SELected.FUNCtion.EXECute

750| IlossRx = SCPI.CALCulate(1).SELected.FUNCtion.DATA

760| RipRx = MaxPeakToPeak(1)

770|

780| '''Displaying the results

790| Display_Update Test_Tr1, Test_Tr2, Test_Ch1, IlossTx, IlossRx, RipTx, RipRx

800|

810| frmDupRes.Show

820|

830| Buff = MsgBox("Do you make another measurement?", vbYesNo, "Duplexer Measurement")

840|

850| If Buff = vbYes Then

860| GoTo Meas_Start

870| End If

880|

890| End Sub
```

```
900|
910| Private Sub Setup_Parameter()
920|
930| Dim I As Long
940| Dim ChDisp As String, TracDisp As String
950| Dim Par(1) As String, Fmt(1) As String
960|
970| ChDisp = "D1"
980| TracDisp = "D1_2"
990| Par(0) = "S13"
1000| Par(1) = "S21"
1010| Fmt(0) = "MLOG"
1020| Fmt(1) = "MLOG"
1030|
1040| SCPI.CALCulate(1).PARameter.Count = 2
1050| SCPI.DISPlay.Split = ChDisp
1060| SCPI.DISPlay.WINDow(1).Split = TracDisp
1070|
1080| For I = 1 To 2
1090| SCPI.CALCulate(1).PARameter(I).DEFine = Par(I - 1)
1100| SCPI.CALCulate(1).PARameter(I).Select
1110| SCPI.CALCulate(1).SELected.Format = Fmt(I - 1)
1120| Next I
1130|
```

```
1140| End Sub
1150|
1160| Private Sub Setup_Segment()
1170|
1180| Dim SegmData(21) As Variant
1190|
1200| SegmData(0) = 5 'Anytime 5 is set at seg ment settings
1210| SegmData(1) = 0 'Allows stimulus range to be set using Start/Stop
frequency
1220| SegmData(2) = 0 'Not allows IF bandwidth to be set
1230| SegmData(3) = 0 'Not allows power to be set
1240| SegmData(4) = 0 'Not allows delay time to be set
1250| SegmData(5) = 0 'Not allows sweep time to be set
1260| SegmData(6) = 5 'Number of segments
1270|
1280| "'Segment 1
1290| SegmData(7) = 1730000000# 'Start frequency
1300| SegmData(8) = 1830000000# 'Stop frequency
1310| SegmData(9) = 50 'Number of points
1320| "'Segment 2
1330| SegmData(10) = 1830000000# 'Start frequency
1340| SegmData(11) = 2030000000# 'Stop frequency
1350| SegmData(12) = 400 'Number of points
1360| "'Segment 3
1370| SegmData(13) = 2030000000# 'Start frequency
```

```
1380| SegmData(14) = 2130000000# 'Stop frequency
1390| SegmData(15) = 50 'Number of points
1400| "'Segment 4
1410| SegmData(16) = 3650000000# 'Start frequency
1420| SegmData(17) = 4030000000# 'Stop frequency
1430| SegmData(18) = 38 'Number of points
1440| "'Segment 5
1450| SegmData(19) = 5500000000# 'Start frequency
1460| SegmData(20) = 6020000000# 'Stop frequency
1470| SegmData(21) = 52 'Number of points
1480|
1490| SCPI.SENSe(1).SEGMent.DATA = SegmData
1500| SCPI.SENSe(1).SWEep.TYPE = "SEGM"
1510| SCPI.DISPlay.WINDow(1).X.SPACing = "OBAS"
1520|
1530| End Sub
1540|
1550| Private Sub Calib_Solt(Chan As Long, SoltType As Long, Port A s
Variant)
1560|
1570| Dim Dmy As Long, I As Long, J As Long, Buff As Long
1580|
1590| Cal_Start:
1600|
1610| Buff = MsgBox("Perform the full " & SoltType & "-port calib ration.",
```

```
vbOKCancel, "Full" & SoltType & "-port calibration")
1620| If Buff = vbCancel Then
1630| GoTo Cal_Skip
1640| End If
1650|
1660| Select Case SoltType
1670| Case 1
1680| SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT1 =
Port(0)
1690| Case 2
1700| SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT2 = Port
1710| Case 3
1720| SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT3 = Port
1730| Case 4
1740| SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT4 = Port
1750| End Select
1760|
1770| For I = 1 To SoltType
1780|
1790| Buff = MsgBox("Connect the Open standard to Port " & CS tr(Port(I
- 1)) & ".", vbOKCancel, "Full" & SoltType & "-port calibr ation")
1800| If Buff = vbOK Then
1810| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.OPEN =
Port(I - 1)
1820| Dmy = SCPI.IEEE4882.OPC
```

```
1830| Else

1840| GoTo Cal_Start

1850| End If

1860|

1870| Buff = MsgBox("Connect the Short standard to Port " & C Str(Port(I
- 1)) & ".", vbOKCancel, "Full" & SoltType & "-port calib ration")

1880| If Buff = vbOK Then

1890| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.Short = Port(I
- 1)

1900| Dmy = SCPI.IEEE4882.OPC

1910| Else

1920| GoTo Cal_Start

1930| End If

1940|

1950| Buff = MsgBox("Connect the Load standard to Port " & CS tr(Port(I
- 1)) & ".", vbOKCancel, "Full" & SoltType & "-port calibr ation")

1960| If Buff = vbOK Then

1970| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.Load = Port(I
- 1)

1980| Dmy = SCPI.IEEE4882.OPC

1990| Else

2000| GoTo Cal_Start

2010| End If

2020| Next I

2030|

2040| For I = 1 To SoltType - 1
```

```
2050| For J = I + 1 To SoltType

2060| Buff = MsgBox("Connect the Thru standard between Por t " & CStr(Port(I - 1)) & " and Port " & CStr(Port(J - 1)) & ".", vb OKCancel, "Full" & SoltType & "-port calibration")

2070| If Buff = vbOK Then

2080| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.THRU = Array(Port(I - 1), Port(J - 1))

2090| Dmy = SCPI.IEEE4882.OPC

2100| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.THRU = Array(Port(J - 1), Port(I - 1))

2110| Dmy = SCPI.IEEE4882.OPC

2120| Else

2130| GoTo Cal_Start

2140| End If

2150| Next J

2160| Next I

2170|

2180| If SoltType <> 1 Then

2190| Buff = MsgBox("Do you measure the Isolation (Optional)? ", vbYesNo, "Full" & SoltType & "-port calibration")

2200| If Buff = vbYes Then

2210| For I = 1 To SoltType - 1

2220| For J = I + 1 To SoltType

2230| Buff = MsgBox("Connect the Load standard to Port " & Port(I - 1) & " and Port " & Port(J - 1) & ".", vbOKCancel, "Full" & SoltType & "-port calibration")

2240| If Buff = vbOK Then
```

```
2250| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.ISOLation = Array(Port( I - 1), Port(J - 1))

2260| Dmy = SCPI.IEEE4882.OPC

2270| SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.ISOLation = Array(Port( J - 1), Port(I - 1))

2280| Dmy = SCPI.IEEE4882.OPC

2290| Else

2300| GoTo Cal_Start

2310| End If

2320| Next J

2330| Next I

2340| End If

2350| End If

2360|

2370| SCPI.SENSe(1).CORRection.COLLect.SAVE

2380| MsgBox "All calibration data completion."

2390|

2400| Cal_Skip:

2410|

2420| End Sub

2430|

2440| Private Sub Setup_Limitline()

2450|

2460| Dim LimDataS13(25) As Variant, LimDataS21(20) As Variant

2470|
```

```
2480| '''Limit line for S13
2490| LimDataS13(0) = 5 'Number of segment
2500| '''Limit_line 1
2510| LimDataS13(1) = 1 'Maximum
2520| LimDataS13(2) = 1730000000# 'Beginning of stimulus
2530| LimDataS13(3) = 1930000000# 'End of stimulus
2540| LimDataS13(4) = 0 'Beginning of response
2550| LimDataS13(5) = 0 'End of response
2560| '''Limit_line 2
2570| LimDataS13(6) = 2 'Minimum
2580| LimDataS13(7) = 1850000000# 'Beginning of stimulus
2590| LimDataS13(8) = 1910000000# 'End of stimulus
2600| LimDataS13(9) = -8 'Beginning of response
2610| LimDataS13(10) = -8 'End of response
2620| '''Limit_line 3
2630| LimDataS13(11) = 1 'Maximum
2640| LimDataS13(12) = 1930000000# 'Beginning of stimulus
2650| LimDataS13(13) = 1990000000# 'End of stimulus
2660| LimDataS13(14) = -35 'Beginning of response
2670| LimDataS13(15) = -35 'End of response
2680| '''Limit_line 4
2690| LimDataS13(16) = 1 'Maximum
2700| LimDataS13(17) = 1990000000# 'Beginning of stimulus
2710| LimDataS13(18) = 2130000000# 'End of stimulus
```

```
2720| LimDataS13(19) = -40 'Beginning of response
2730| LimDataS13(20) = -40 'End of response
2740| '''Limit_line 5
2750| LimDataS13(21) = 1 'Maximum
2760| LimDataS13(22) = 2130000000# 'Beginning of stimulus
2770| LimDataS13(23) = 6020000000# 'End of stimulus
2780| LimDataS13(24) = -20 'Beginning of response
2790| LimDataS13(25) = -20 'End of response
2800|
2810| '''Limit line for S21
2820| LimDataS21(0) = 4 'Number of segment
2830| '''Limit_line 1
2840| LimDataS21(1) = 1 'Maximum
2850| LimDataS21(2) = 1730000000# 'Beginning of stimulus
2860| LimDataS21(3) = 1850000000# 'End of stimulus
2870| LimDataS21(4) = -40 'Beginning of response
2880| LimDataS21(5) = -40 'End of response
2890| '''Limit_line 2
2900| LimDataS21(6) = 1 'Maximum
2910| LimDataS21(7) = 1850000000# 'Beginning of stimulus
2920| LimDataS21(8) = 1910000000# 'End of stimulus
2930| LimDataS21(9) = -40 'Beginning of response
2940| LimDataS21(10) = -40 'End of response
2950| '''Limit_line 3
```

```
2960| LimDataS21(11) = 1 'Maximum
2970| LimDataS21(12) = 1910000000# 'Beginning of stimulus
2980| LimDataS21(13) = 6020000000# 'End of stimulus
2990| LimDataS21(14) = 0 'Beginning of response
3000| LimDataS21(15) = 0 'End of response
3010| '''Limit_line 4
3020| LimDataS21(16) = 2 'Minimum
3030| LimDataS21(17) = 1930000000# 'Beginning of stimulus
3040| LimDataS21(18) = 1990000000# 'End of stimulus
3050| LimDataS21(19) = -10 'Beginning of response
3060| LimDataS21(20) = -10 'End of response
3070|
3080| SCPI.CALCulate(1).PARameter(1).Select
3090| SCPI.CALCulate(1).SELected.LIMit.DATA = LimDataS13
3100| SCPI.CALCulate(1).SELected.LIMit.DISPlay.STATe = True
3110| SCPI.CALCulate(1).SELected.LIMit.STATe = True
3120|
3130| SCPI.CALCulate(1).PARameter(2).Select
3140| SCPI.CALCulate(1).SELected.LIMit.DATA = LimDataS21
3150| SCPI.CALCulate(1).SELected.LIMit.DISPlay.STATe = True
3160| SCPI.CALCulate(1).SELected.LIMit.STATe = True
3170|
3180| End Sub
3190|
```

```
3200| Private Sub Setup_Register()
3210|
3220| SCPI.STATus.QUEStionable.LIMit.CHANnel(1).PTRansition = 6
3230| SCPI.STATus.QUEStionable.LIMit.CHANnel(1).NTRansition = 0
3240| SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ENABle = 6
3250|
3260| End Sub
3270|
3280| Sub Display_Update(Test_Tr1 As Integer, Test_Tr2 As Integer,
Test_Ch1 As Integer, IlossTx As Variant, IlossRx As Variant, RipTx As
Variant, RipRx As Variant)
3290|
3300| SCPI.DISPlay.UPDate.IMMediate
3310|
3320| If Test_Tr1 = 2 Then
3330| frmDupRes.lblJudgeS13.BackColor = RGB(255, 0, 0)
3340| frmDupRes.lblJudgeS13.Caption = "Fail"
3350| Else
3360| frmDupRes.lblJudgeS13.BackColor = RGB(0, 0, 255)
3370| frmDupRes.lblJudgeS13.Caption = "Pass"
3380| End If
3390|
3400| If Test_Tr2 = 4 Then
3410| frmDupRes.lblJudgeS21.BackColor = RGB(255, 0, 0)
3420| frmDupRes.lblJudgeS21.Caption = "Fail"
```

```
3430| Else
3440| frmDupRes.lblJudgeS21.BackColor = RGB(0, 0, 255)
3450| frmDupRes.lblJudgeS21.Caption = "Pass"
3460| End If
3470|
3480| If Test_Ch1 = 2 Then
3490| frmDupRes.lblResult.BackColor = RGB(255, 0, 0)
3500| frmDupRes.lblResult.Caption = "Fail"
3510| Else
3520| frmDupRes.lblResult.BackColor = RGB(0, 0, 255)
3530| frmDupRes.lblResult.Caption = "Pass"
3540| End If
3550|
3560| frmDupRes.txtIlossS13.Text = Format(IlossTx(0), "0.#### ")
3570| frmDupRes.txtIlossS21.Text = Format(IlossRx(0), "0.#### ")
3580|
3590| frmDupRes.txtRipS13.Text = Format(RipTx, "0.####")
3600| frmDupRes.txtRipS21.Text = Format(RipRx, "0.####")
3610|
3620| End Sub
```

E5091AFEM

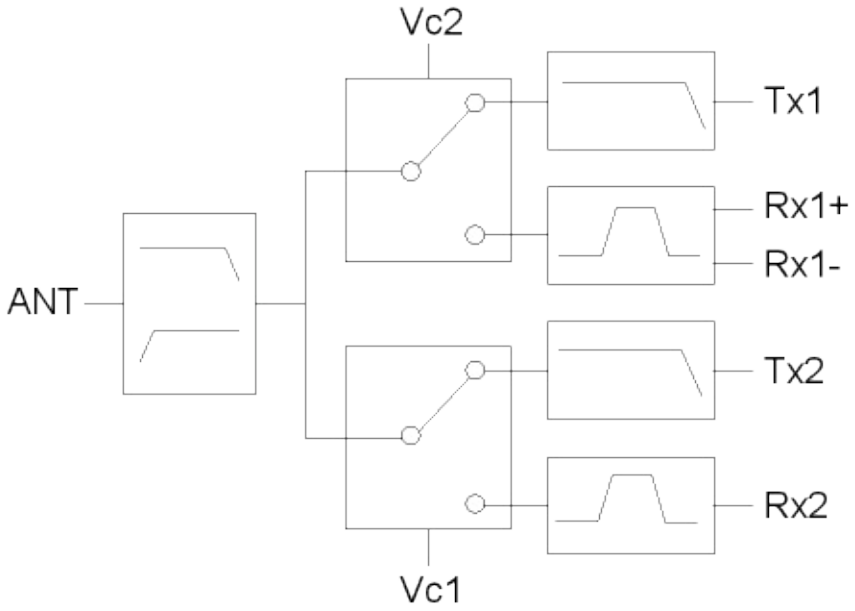
-
-

apl_fem.vbaFEME5091A

- E5071C4SE5071C
- VBAE5091A-016

mdlFemMeas		FEM

ECal6FEMEGSM:Tx-Antenna1EGSM:Antenna-Rx2
GSM1800:Tx-Antenna3GSM1800:Antenna-Rx4



e5070buj073

“Connect A and T1 to ECal Module”AT1ECalE5091AAT1Ecal
OK1ECal **Retry()** **Cancel()** 24

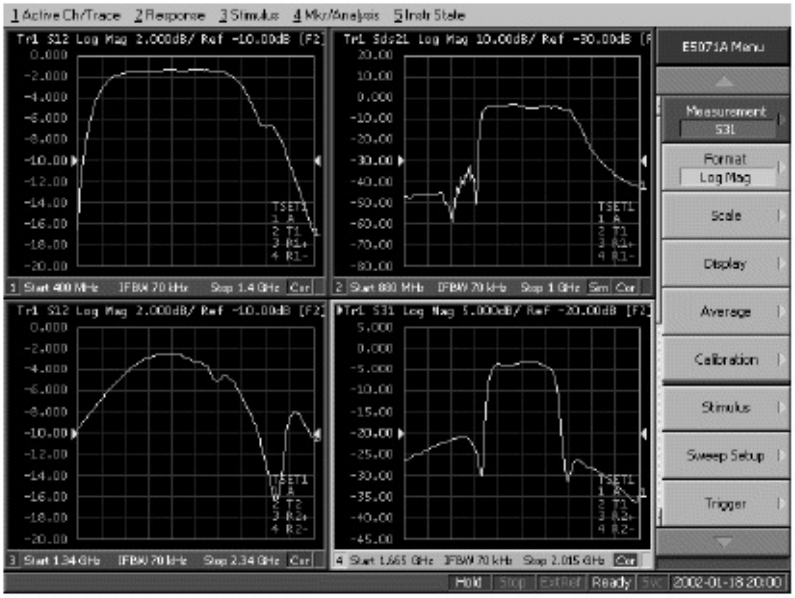
“Set DUT”FEME5091A **OK**

FEM	**E5091A**

		A
EGSM	Tx	T1
	Rx+	R1+
	Rx-	R1-
GSM1800	Tx	T2
	Rx	R2+
Vc1		1
Vc2		2

LCD

"apl_fem.vba"



FEM(mdlFemMeas)

140330

E5091A14()

	1	**2**	**3**	**4**	
1	A	T1	R1+	R1-	0 (00000000)
2	A	T1	R1+	R1-	2 (00000010)
3	A	T2	R2+	R2-	0 (00000000)
4	A	T2	R2+	R2-	1 (00000001)

340660

1	400 MHz	1.4 GHz	51	1	S12
2	880 MHz	1 GHz	101	1	Sds21
3	1.34 GHz	2.34 GHz	201	1	S12
4	1.665 GHz	2.015 GHz	101	1	S31

	I		

1		E/span> E/span>	[EGSM] Tx-Antenna
2		SE:1, Bal:3,4	[EGSM] Antenna-Rx
3		E/span> E/span>	[GSM1800] Antenna-Rx
4		E/span> E/span>	[GSM1800] Tx-Antenna

710

720

7401020

1~4Ch

780810ID1E5091A1Port1(Ch-1)2Port2(Ch-1)3Port3(Ch-1)4Port4(Ch-1)

820ID()1E5091AClines(Ch-1)

860890Freq_star(Ch-1)Freq_stop(Ch-1)Nop(Ch-1)N_tr(Ch-1)

910950() (Fsim (Ch-1))ONDev
S)Trc (Ch-1)

970(Fsim (i))(S)Trc (Ch-1)

9901010Ttl (Ch-1)ONON

1040

"Bus."()

10501060

ID1E5091AONON

11001120

ECal_solt (14101770) ECal1(AT12)

11301210

2

1260

DUT() **OK**

12801290

(SCPI.IEEE4882.OPC object1)

13101330

1~41

1350

1360

Yes() DUT

ECal_solt (14101770)

1460

14601480

E5091ATset_PortECal **OK**

1500

ECal_Err(16701740)

1540

solt1ChAna_port(0)1ECal

1560

solt2ChAna_port2ECal

1580

solt3ChAna_port3ECal

1600

solt4ChAna_port4ECal

1630

ECal_solt0

16701740

780810ID()1E5091A1Port1 (Ch-1)2Port2
1)4Port4 (Ch-1)

1670

1680

1700RetryECal

17201730 **Cancel** ECal_solt

FEM(mdlFemMeas)

```
10| Sub Main()
20|
30| Dim Port1(3) As String, Port2(3) As String
40| Dim Port3(3) As String, Port4(3) As String
50| Dim Trc(3) As String, Fsim(3) As Boolean, Dev(3) As String
60| Dim Tpl(3) As Variant, Ttl(3) As String, Inp_char As String
70| Dim Freq_star(3) As Double, Freq_stop(3) As Double
80| Dim CLines(3) As Long, Nop(3) As Long, N_tr(3) As Long
90| Dim Ch As Long, Res As Long, Buff As Long, Dmy As Long
100| Dim AnaPort As Variant
110| '
120| ' E5091A Setup
130| '
140| Port1(0) = "A" '[Ch1] Port1: A
150| Port2(0) = "T1" ' Port2: T1
160| Port3(0) = "R1" ' Port3: R1+
170| Port4(0) = "R1" ' Port4: R1-
180| CLines(0) = 0 ' Control Lines: 0
190| Port1(1) = "A" '[Ch2] Port1: A
200| Port2(1) = "T1" ' Port2: T1
210| Port3(1) = "R1" ' Port3: R1+
220| Port4(1) = "R1" ' Port3: R1-
```

```
230| CLines(1) = 2 ' Control Lines: 2 (Line1:HIGH)
240| Port1(2) = "A" '[Ch3] Port1: A
250| Port2(2) = "T2" ' Port2: T2
260| Port3(2) = "R2" ' Port3: R2+
270| Port4(2) = "R2" ' Port4: R2- (Dummy)
280| CLines(2) = 0 ' Control Lines: 2 (Line1:HIGH)
290| Port1(3) = "A" '[Ch4] Port1: A
300| Port2(3) = "T2" ' Port2: T2
310| Port3(3) = "R2" ' Port3: R2+
320| Port4(3) = "R2" ' Port4: R2- (Dummy)
330| CLines(3) = 1 ' Control Lines: 1 (Line0:HIGH)
340| '
350| ' Measurement Condition
360| ' [Ch1]
370| Freq_star(0) = 400000000# ' Start frequency : 400 MHz
380| Freq_stop(0) = 1400000000# ' Stop frequency : 1.4 GHz
390| Nop(0) = 51 ' Number of points : 51
400| N_tr(0) = 1 ' Number of traces : 1
410| Fsim(0) = False ' Fixture Simulator : OFF
420| Trc(0) = "S12" ' Meas. parameter : S12
430| Ttl(0) = "[EGSM] Tx-Antenna" ' Title
440| ' [Ch2]
450| Freq_star(1) = 880000000# ' Start frequency : 880 MHz
460| Freq_stop(1) = 1000000000# ' Stop frequency : 1 GHz
```

```
470| Nop(1) = 101 ' Number of points : 101
480| N_tr(1) = 1 ' Number of traces : 1
490| Fsim(1) = True ' Fixture Simulator : ON
500| Dev(1) = "SBAL" ' Bal. Device Type : SE-Bal
510| Tpl(1) = Array(1, 3, 4) ' Topology : SE:1, Bal:3-4
520| Trc(1) = "SDS21" ' Meas. parameter : Sds21
530| Ttl(1) = "[EGSM] Antenna-Rx" ' Title
540| ' [Ch3]
550| Freq_star(2) = 1340000000# ' Start frequency : 1.34 GHz
560| Freq_stop(2) = 2340000000# ' Stop frequency : 2.34 GHz
570| Nop(2) = 201 ' Number of points : 201
580| N_tr(2) = 1 ' Number of traces : 1
590| Fsim(2) = False ' Fixture Simulator : OFF
600| Trc(2) = "S12" ' Meas. parameter : S12
610| Ttl(2) = "[GSM1800] Tx-Antenna" ' Title
620| ' [Ch4]
630| Freq_star(3) = 1665000000# ' Start frequency : 1.665 GHz
640| Freq_stop(3) = 2015000000# ' Stop frequency : 2.015 GHz
650| Nop(3) = 101 ' Number of points : 101
660| N_tr(3) = 1 ' Number of traces : 1
670| Fsim(3) = False ' Fixture Simulator : OFF
680| Trc(3) = "S31" ' Meas. parameter : S31
690| Ttl(3) = "[GSM1800] Antenna-Rx" ' Title
700|
```

```
710| SCPI.SYSTem.PRESet
720| SCPI.DISPlay.Split = "D12_34"
730|
740| For Ch = 1 To 4
750| '
760| ' E5091A Setup
770| '
780| SCPI.SENSe(Ch).MULTiplexer(1).TSET9.Port1 = Port1(Ch - 1)
790| SCPI.SENSe(Ch).MULTiplexer(1).TSET9.Port2 = Port2(Ch - 1)
800| SCPI.SENSe(Ch).MULTiplexer(1).TSET9.Port3 = Port3(Ch - 1)
810| SCPI.SENSe(Ch).MULTiplexer(1).TSET9.Port4 = Port4(Ch - 1)
820| SCPI.SENSe(Ch).MULTiplexer(1).TSET9.OUTPut.DATA =
CLines(Ch - 1)
830| '
840| ' Measurement Condition
850| '
860| SCPI.SENSe(Ch).FREQuency.STARt = Freq_star(Ch - 1)
870| SCPI.SENSe(Ch).FREQuency.STOP = Freq_stop(Ch - 1)
880| SCPI.SENSe(Ch).SWEep.POINts = Nop(Ch - 1)
890| SCPI.CALCulate(Ch).PARameter.Count = N_tr(Ch - 1)
900| If Fsim(Ch - 1) = True Then
910| SCPI.CALCulate(Ch).FSIMulator.STATe = True
920| SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice = Dev(Ch - 1)
930|
SCPI.CALCulate(Ch).FSIMulator.BALun.TOPology.SBALanced.PPOR ts
```

```
= Tpl(Ch - 1)

940| SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(1).STATe = Tr
ue

950| SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(1).SBALanced.
DEFine = Trc(Ch - 1)

960| Else

970| SCPI.CALCulate(Ch).PARameter(1).DEFine = Trc(Ch - 1)

980| End If

990| SCPI.DISPlay.WINDow(Ch).TITLe.DATA = Ttl(Ch - 1)

1000| SCPI.DISPlay.WINDow(Ch).TITLe.STATe = True

1010| SCPI.INITiate(Ch).CONTinuous = True

1020| Next Ch

1030|

1040| SCPI.TRIGger.SEQuence.Source = "BUS"

1050| SCPI.SENSe.MULTiplexer(1).DISPlay.STATe = True

1060| SCPI.SENSe.MULTiplexer(1).STATe = True

1070| '

1080| ' Calibration

1090| '

1100| AnaPort = Array(1, 2)

1110| Res = ECal_Solt(1, 2, AnaPort, "A and T1")

1120| If Res <> 0 Then GoTo Prg_end

1130| AnaPort = Array(1, 3, 4)

1140| Res = ECal_Solt(2, 3, AnaPort, "A, R1+ and R1-")

1150| If Res <> 0 Then GoTo Prg_end
```

```
1160| AnaPort = Array(1, 2)
1170| Res = ECal_Solt(3, 2, AnaPort, "A and T2")
1180| If Res <> 0 Then GoTo Prg_end
1190| AnaPort = Array(1, 3)
1200| Res = ECal_Solt(4, 2, AnaPort, "A and R2+")
1210| If Res <> 0 Then GoTo Prg_end
1220| '
1230| ' Measurement
1240| '
1250| Meas_Start:
1260| MsgBox "Connect DUT.", vbOKOnly, "Measurement"
1270|
1280| SCPI.TRIGger.SEQuence.SINGle
1290| Dmy = SCPI.IEEE4882.OPC
1300|
1310| For Ch = 1 To 4
1320| SCPI.DISPlay.WINDow(Ch).TRACe(1).Y.SCALe.AUTO
1330| Next Ch
1340|
1350| Buff = MsgBox("Do you make another measurement?", vbYesNo)
1360| If Buff = vbYes Then GoTo Meas_Start
1370|
1380| Prg_end:
1390| End Sub
```

```
1400|

1410| Function ECal_Solt(Ch As Long, Solt As Long, AnaPort As Variant, TsetPort As String) As Long

1420| Dim Err_info As Variant

1430| Dim Buff As Long

1440|

1450| Ecal_start:

1460| SCPI.IEEE4882.CLS

1470|

1480| MsgBox "Connect " + TsetPort + " to ECal Module."

1490|

1500| On Error GoTo Ecal_err

1510|

1520| Select Case Solt

1530| Case 1

1540| SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT1 = AnaPort(0)

1550| Case 2

1560| SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT2 = AnaPort

1570| Case 3

1580| SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT3 = AnaPort

1590| Case 4

1600| SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT4 = AnaPort

1610| End Select

1620|
```

```
1630| ECal_Solt = 0
1640| GoTo Ecal_end
1650|
1660| Ecal_err:
1670| Err_info = SCPI.SYSTem.Error
1680| Buff = MsgBox("Error: " + Err_info(1), vbRetryCancel)
1690| If Buff = vbRetry Then
1700| Resume Ecal_start
1710| Else
1720| ECal_Solt = Err_info(0)
1730| Resume Ecal_end
1740| End If
1750|
1760| Ecal_end:
1770| End Function
```

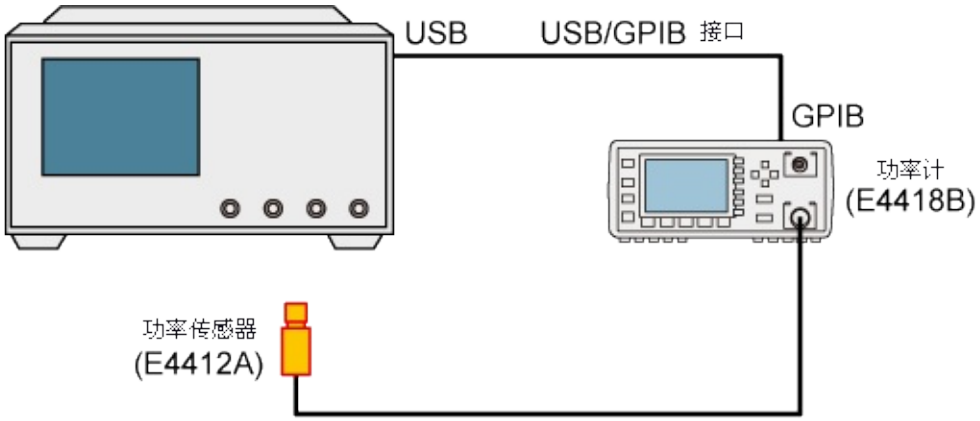
-
-
-

pow_cal.vba E4418BE4412A(VBA)VBA

mdlPowCal		
1 2		VISA

USB/GPIBE5071CE4418B(GPIB:13)(E4418B)(E4412A)
E5071C

E5071C



e5071c336

VBAGPIBE5071C **“Do you want to zero and calibrate the power sensor?”** **Yes** **No**

APOWER RF() **OK“** **Zeroing and calibrating the power sensor is complete”()**

E5071C1 **OK** **CORR_DATA**

(mdlPowCal)

100150

()(41)(-20+12 dBm)(-20 dBm)(-10 dBm)Swp_typeNop Pow_rangStart_pStop_pCw_freq

160170

(4)(10 dBm)Num_avgLimit

200

E5071C

230

GPIB(13)E5071C

260310

1Swp_typeNopPow_rangStart_pStop_pCw_freq

340370

Yes Control_PowerMeter9001340 **No**

410420

E5071C1 **OK**

450470

11Num_avg

500

E5071CErr

510590

Corr_data_ Limit_Test function_Limit_TestFalse( ) **Yes** **No**

610660

Yes() **No**

690790

CORR_DATA

Control_PowerMeter9001340

9901000

VISADefrm(12601300)

10301040

GPIB13E441812601300

10701090

VISA0

1100

POWER REF() **OK**

11101120

VISA10

11301190

VISARgst3251AND“”RsltRslt1

1220

VISA

12601300

VISA

Limit_Test13601530

14201470

FalseTrue

mdlPowCal

```
10| Sub Main()

20|

30| Dim Swp_type As String, File As String

40| Dim Start_p As Double, Stop_p As Double, Cw_freq As Double, Limi
t As Double

50| Dim Nop As Long, Pow_rang As Long, Num_avg As Long, Data_size
As Long, Buff As Long, Dmy As Long

60| Dim Corr_data As Variant, Err As Variant

70| Dim Verifier As Boolean

80| Dim FileNo As Integer, I As Integer

90|

100| Swp_type = "POW" 'Sweep type : POWER

110| Nop = 41 'Number of points : 41

120| Pow_rang = 0 'Power range : -20 to +12 dBm

130| Start_p = -20# 'Start power : -20 dBm

140| Stop_p = -10# 'Stop power : -10 dBm

150| Cw_freq = 1000000000# 'CW frequency : 1 GHz

160| Num_avg = 4 'Number of averaging : 4

170| Limit = 10# 'limit for corrected data : 10 dBm

180|

190| '''Presetting the ENA

200| SCPI.SYSTem.PRESet
```

```
210|
220| '''Setting GPIB address of the power meter to ENA
230| SCPI.SYSTem.COMMunicate.GPIB.PMETer.address = 13
240|
250| '''Setting measurement conditions
260| SCPI.SENSe(1).SWEep.TYPE = Swp_type
270| SCPI.SENSe(1).SWEep.POINts = Nop
280| SCPI.Source(1).POWer.ATTenuation.DATA = Pow_rang
290| SCPI.Source(1).POWer.STARt = Start_p
300| SCPI.Source(1).POWer.STOP = Stop_p
310| SCPI.SENSe(1).FREQuency.CW = Cw_freq
320|
330| '''Performing a calibration in the power meter
340| Buff = MsgBox("Do you perform zeroing and calibrating the power sensor?", vbYesNo, "Power meter calibration")
350| If Buff = vbYes Then
360| Control_PowerMeter
370| End If
380|
390| Meas_Start:
400| '''Connecting the power sensor to the port 1 in the ENA
410| SCPI.IEEE4882.CLS
420| MsgBox "Set the power sensor connected to the port 1 in the ENA.", vbOKOnly, "Power meter calibration"
430|
```

```
440| '''Performing power calibration measurement

450| SCPI.Source(1).POWer.PORT(1).CORRection.COLLect.AVERage.count = Num_avg

460| SCPI.Source(1).POWer.PORT(1).CORRection.COLLect.ACQuire = "ASEN "

470| Dmy = SCPI.IEEE4882.OPC

480|

490| '''Error handling at power meter calibration

500| Err = SCPI.SYSTem.Error

510| If Err(0) = 0 Then

520| Corr_data = SCPI.Source(1).POWer.PORT(1).CORRection.DATA

530| Verifier = Limit_Test(Nop, Limit, Corr_data)

540| If Verifier = False Then

550| Buff = MsgBox("Do you perform the power meter calibration me asurement again?", vbYesNo, "Power meter calibration")

560| If Buff = vbYes Then GoTo Meas_Start

570| If Buff = vbNo Then GoTo Prog_Stop

580| End If

590| MsgBox "Power meter calibration measurement is complete.", vbO KOnly, "Power meter calibration"

600|

610| Else

620| MsgBox "Error: " & Err(1)

630| Buff = MsgBox("Do you perform the power meter calibration meas urement again?", vbYesNo, "Power meter calibration")
```

```
640| If Buff = vbYes Then GoTo Meas_Start

650| If Buff = vbNo Then GoTo Prog_Stop

660| End If

670|

680| '''Installing the corrected data to a file

690| FileNo = FreeFile

700| File = "CORR_DATA"

710|

720| Open File For Output As FileNo

730|

740| For I = 0 To Nop - 1

750| Write #FileNo, Val(Corr_data(I))

760| Next I

770| Close #FileNo

780|

790| MsgBox "Installing the corrected data to the file is DONE.", vbO
KOnly, "Power meter calibration"

800|

810| GoTo Prog_End

820|

830| Prog_Stop:

840| MsgBox "Program Interruption", vbOKOnly, "Power meter calibratio
n"

850|

860| Prog_End:
```

```
870|
880| End Sub
890|
900| Private Sub Control_PowerMeter()
910|
920| Dim Status As Long 'VISA function status return
930| Dim Defrm As Long 'Session to default resource code
940| Dim E4418 As Long 'Session to power meter
950| Dim Rslt As Integer
960| Dim Rgst As String * 10
970|
980| '''Initializing the VISA system
990| Status = viOpenDefaultRM(Defrm)
1000| If (Status <> VI_SUCCESS) Then GoTo VisaErrorHandler
1010|
1020| '''Opening the session to the power meter
1030| Status = viOpen(Defrm, "GPIB0::13::INSTR", 0, 0, E4418)
1040| If (Status <> VI_SUCCESS) Then GoTo VisaErrorHandler
1050|
1060| '''Zeroing and calibratingthe power meter
1070| Status = viVPrintf(E4418, "SYST:PRES" & Chr$(10), 0)
1080| Status = viVPrintf(E4418, "*CLS" & Chr$(10), 0)
1090| Status = viVPrintf(E4418, "*ESE 1" & Chr$(10), 0)
1100| MsgBox "Set the power sensor connected to the POWER REF port
in the power meter.", vbOKOnly, "Power meter calibration"
```

```
1110| Status = viVPrintf(E4418, "CAL1:ALL" & Chr$(10), 0)

1120| Status = viVPrintf(E4418, "*OPC" & Chr$(10), 0)

1130| Do While Rslt = 0

1140| Status = viVPrintf(E4418, "*STB?" & Chr$(10), 0)

1150| Status = viVScanf(E4418, "%t", Rgst)

1160| Rslt = CInt(CInt(Rgst) And 32)

1170| Loop

1180|

1190| MsgBox "Zeroing and Calibrating the power sensor is complete.",
vbOKOnly, "Power meter calibration"

1200|

1210| "'Closing the resource manager session

1220| Call viClose(Defrm)

1230|

1240| GoTo Prog_End

1250|

1260| VisaErrorHandler:

1270| Dim VisaErr As String * 200

1280| Call viStatusDesc(Defrm, Status, VisaErr)

1290| MsgBox "Error : " & VisaErr, vbExclamation

1300| End

1310|

1320| Prog_End:

1330|

1340| End Sub
```

```
1350|

1360| Function Limit_Test(Nop As Long, Limit As Double, Corr_data As
Var iant) As Boolean

1370|

1380| Dim I As Integer

1390|

1400| For I = 0 To Nop - 1

1410|

1420| If Abs(Corr_data(I)) > Limit Then

1430| SCPI.Source(1).POWer.PORT(1).CORRection.STATe = False

1440| MsgBox "The corrected data is out of limit!", vbExclamation,
"Power meter calibration"

1450| Limit_Test = False

1460| Exit Function

1470| End If

1480|

1490| Next I

1500|

1510| Limit_Test = True

1520|

1530| End Function
```

PC

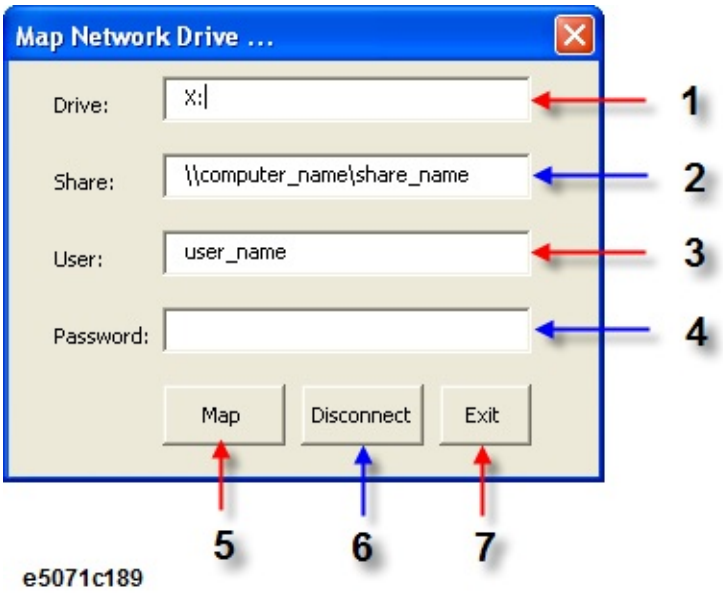
-
- VBA
- VBA

map_drive.vbaPCE5071CVBAVBA

frmMapDrive		
1		frmMapDrive

VBA

map_drive.vba



(1)(2)(3)(4) **Map** (5

Windows XP " PC

1. (1) (6)
2. **Exit** (7)

VBA

frmMapDrive

Sub CommandButton1_Click

Map IsDriveNameInUseMapDrive

Sub CommandButton2_Click

DisconnectDrive

Function IsDriveNameInUse

txtDrive.Text 1

Sub MapDrive

txtDrive.Text 1txtShare.Text 2txtUser.Text 3
txtPasswd.Text 4

Sub DisconnectDrive

txtDrive.Text 1

Sub CommandButton3_Click

Exit

PCfrmMapDrive

```
Private Sub CommandButton1_Click()

If Not IsDriveNameInUse Then

Call MapDrive

Else

MsgBox "Drive """ & txtDrive.Text & """ is Already used", vb Critical

End If

End Sub
```

```
Private Sub CommandButton2_Click()
Call DisconnectDrive
End Sub

Private Function IsDriveNameInUse() As Boolean
Set fso = CreateObject("Scripting.FileSystemObject")
IsDriveNameInUse = fso.DriveExists(txtDrive.Text)
End Function

Private Sub MapDrive()
Set network = CreateObject("wscript.network")
Call network.MapNetworkDrive(txtDrive.Text, txtShare.Text, vbFal se,
txtUser. Text, txtPasswd.Text)
End Sub

Private Sub DisconnectDrive()
Set network = CreateObject("wscript.network")
network.RemoveNetworkDrive txtDrive.Text
End Sub

Private Sub CommandButton3_Click()
Unload Me
End Sub
```

Complex.real.imag

```
Dim Num as Complex
Num.real=1.0
Num.imag=2.0
```

ComplexSet(x,y)	
ComplexPolar(x,y)	
ComplexSetArray(x)	
ComplexAdd(x,y)	
ComplexSub(x,y)	
ComplexMul(x,y)	
ComplexDiv(x,y)	
ComplexAbs(x)	
ComplexArg(x)	
ComplexNorm(x)	
ComplexConj(x)	
ComplexCos(x)	
ComplexCosh(x)	
ComplexSin(x)	
ComplexSinh(x)	

ComplexExp(x)	e^x.
ComplexLog(x)	
ComplexLog10(x)	
ComplexSqrt(x)	

```
:
:

Dim Dmy As Long
Dim s21_raw As Variant
Dim s31_raw As Variant
Dim s21_Comp As Complex
Dim s31_Comp As Complex
Dim trAce_ratio_comp As Complex
Dim trAce_ratio(401) As Double

SCPI.DISPlay.Split = "D1"
SCPI.DISPlay.WINDow(1).Split = "D12_34"
SCPI.CALCulate(1).PARameter.Count = 2
SCPI.CALCulate(1).PARameter(1).DEFine = "s21"
SCPI.CALCulate(1).PARameter(2).DEFine = "s31"
SCPI.SENSe(1).SWEep.POINts = 201

:
:
:

SCPI.TRIGger.SEQuence.Source = "bus"
```

```
SCPI.TRIGger.SEQuence.SINGle
Dmy = SCPI.IEEE4882.OPC

'''' Get corrected data array
SCPI.CALCulate(1).PARameter(1).SELect
s21_raw = SCPI.CALCulate(1).SELected.DATA.SDATa
SCPI.CALCulate(1).PARameter(2).SELect
s31_raw = SCPI.CALCulate(1).SELected.DATA.SDATa

For i = 0 To 200

'''' Copy corrected data array to the complex data array
'''' to take advantage of complex operation library
s21_Comp = ComplexSet(s21_raw(2 * i), s21_raw(2 * i + 1))
s31_Comp = ComplexSet(s31_raw(2 * i), s31_raw(2 * i + 1))

'''' Calculate the ratio of S31 and S21
'''' S31/S21
trAce_ratio_comp = ComplexDiv(s31_Comp, s21_Comp)

trAce_ratio(2 * i) = trAce_ratio_comp.real
trAce_ratio(2 * i + 1) = trAce_ratio_comp.imag

Next i
```

```
SCPI.CALCulate(1).PARameter.Count = 4

"" Write "S31/S21" data to corrected data array for the trace 3 (LogMag)
SCPI.CALCulate(1).PARameter(3).SELect
SCPI.CALCulate(1).SELected.Format = "MLOG"
SCPI.CALCulate(1).SELected.DATA.SDATa = trAce_ratio

"" Write "S31/S21" data to corrected data array for the trace 4 (Phase)
SCPI.CALCulate(1).PARameter(4).SELect
SCPI.CALCulate(1).SELected.Format = "PHASe"
SCPI.CALCulate(1).SELected.DATA.SDATa = trAce_ratio

:
:
```

ComplexAbs(*x*)

Result = ComplexAbs(*x*)

x (Complex)

(Double)

```
Dim a As Complex, b As Double
a = ComplexSet(1.5, 2.0)
b = ComplexAbs(a)
```

ComplexAdd(*x*,*y*)

Result = ComplexAdd(*x*,*y*)

xyx+y

x(Complex)

y(Complex)

(Complex)

```
Dim a As Complex, b As Complex, c As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSet(0.5, 3.5)
c = ComplexAdd(a, b)
```

ComplexArg(*x*)

Result = ComplexArg(*x*)

x(Complex)

(Double)

```
Dim a As Complex, b As Double, c As Double, pi As Double
a = ComplexSet(1.5, 2.0)
b = ComplexArg(a)
pi = 3.14159265
c = b * 180 / pi      ‘ radian -> degree
```

ComplexConj(*x*)

Result = ComplexConj(*x*)

x(Complex)

(Complex)

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexConj(a)
```

ComplexCos(*x*)

Result = ComplexCos(*x*)

$x($ $x)$

x(Complex)

(Complex)

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexCos(a)
```

ComplexCosh(*x*)

Result = ComplexCosh(*x*)

$x($ $x)$

x(Complex)

(Complex)

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexCosh(a)
```

ComplexDiv(*x*,*y*)

Result = ComplexDiv(*x*,*y*)

*xy*x/y

x(Complex)

y(Complex)

(Complex)

```
Dim a As Complex, b As Complex, c As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSet(0.5, 3.5)
c = ComplexDiv(a, b)
```

ComplexExp(*x*)

Result = ComplexExp(*x*)

e*x*.

x(Complex)

(Complex)

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexExp(a)
```

ComplexLog(*x*)

Result = ComplexLog(*x*)

$x(\log(x))$

x(Complex)

(Complex)

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexLog(a)
```

ComplexLog10(*x*)

Result = ComplexLog(*x*)

$x\log 10x$

x(Complex)

(Complex)

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexLog10(a)
```

ComplexMul(*x*,*y*)

Result = ComplexMul(*x*,*y*)

*xy*x\y

x(Complex)

y(Complex)

: (Complex)

```
Dim a As Complex, b As Complex, c As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSet(0.5, 3.5)
c = ComplexMul(a, b)
```

ComplexNorm(*x*)

Result = ComplexNorm(*x*)

x(Complex)

(Double)

```
Dim a As Complex, b As Double
a = ComplexSet(1.5, 2.0)
b = ComplexNorm(a)
```

ComplexPolar(*x*,*y*)

z = ComplexPolar(*x*,*y*)

zxy

x(Double)
y(Double)
z(Complex)

```
Dim a As Complex, pi As Double
pi = 3.14159265
a = ComplexPolar(2.5, 60 * pi / 180)
```

ComplexSet(*x*,*y*)

z = ComplexSet(*x*,*y*)

zxy *xyz.realz.imag*

x(Double)

y(Double)

z(Complex)

```
Dim a as Complex
a = ComplexSet(1.5, 2.0)
```

ComplexSetArray(*x*)

y = ComplexSetArray(*x*)

x $x2$ y

*x*VariantDouble

*y*Complex

```
Dim a as Variant, b as Complex
a = SCPI.CALCulate(1).SELected.DATA.SDATa
b = ComplexSetArray(a)
```

ComplexSin(*x*)

Result = ComplexSin(*x*)

$x(\sin(x))$

x(Complex)

(Complex)

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSin(a)
```

ComplexSinh(*x*)

Result = ComplexSinh(*x*)

$x(\sinh(x))$

x(Complex)

(Complex)

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSinh(a)
```

ComplexSqrt(*x*)

Result = ComplexSqrt(*x*)

x(Complex)

(Complex)

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSqrt(a)
```

ComplexSub(*x*,*y*)

Result = ComplexSub(*x*,*y*)

$xy(x - y)$

x(Complex)

y(Complex)

(Complex)

```
Dim a As Complex, b As Complex, c As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSet(0.5, 3.5)
c = ComplexSub(a, b)
```

-
-
-
-
-

VBA

COME5071C

1.
    -
    -
2.

COM

COM

- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STARt
- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STOP
- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STATe
- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.COUPle

COM

- SCPI.CALCulate(Ch).SELected.FUNCtion.PEXCursion
- SCPI.CALCulate(Ch).SELected.FUNCtion.PPOLarity

MaxPeakToPeak(Chan)	
MaxRightGap(Chan)	
MaxLeftGap(Chan)	
MaxGap(Chan)	
MaxEnvelopeGap(Chan)	22
GapMean(Chan)	
MaxRippleValue(Chan)	
MaxRipplePoint(Chan,Stim)	(*StimD*
Pole(Chan,D,LeftStim,LeftValue, RightStim,RightValue)	() *D*
FirstRightGap(Chan)	
FirstLeftGap(Chan)	
FirstRightInterval(Chan)	

FirstLeftInterval(Chan)	

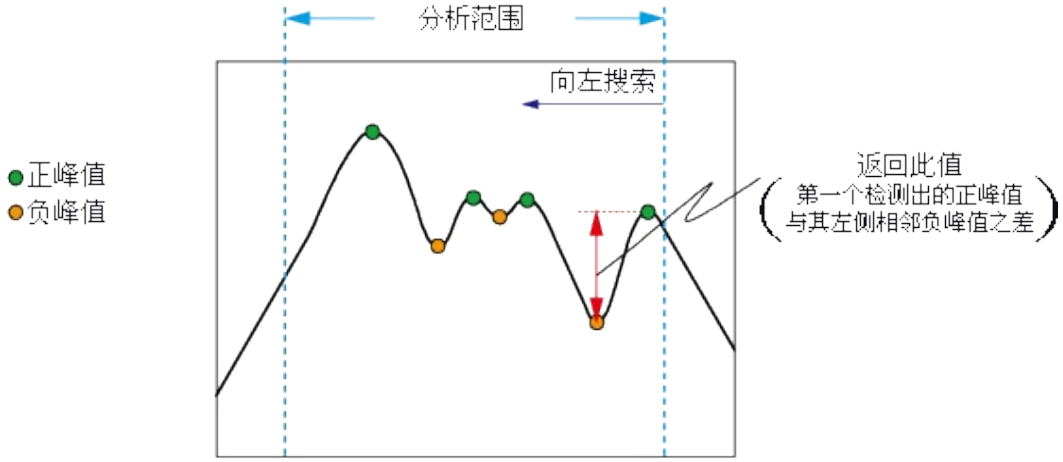
```
Sub Sample()

Dim Val As Double   (1)

SCPI.CALCulate(1).SELected.FUNCtion.PEXCursion = 1.5   (2)
SCPI.CALCulate(1).SELected.FUNCtion.PPOLarity = "BOTH"   (2)
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt = 935E6   (3)
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP = 960E6   (3)
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STATe = True   (3)
  ?span style="font-family: 'Times New Roman'; font-size: 7.0pt;">
  ?span style="font-family: 'Times New Roman'; font-size: 7.0pt;">
Val = MaxPeakToPeak(1)    (4)

End Sub
```

1. ValDouble
2. 1.5
3. 1935 MHz960 MHz
4. 1ValMaxPeakToPeak

FirstLeftGap(*Chan*)

Value = FirstLeftGap(*Chan*)

FirstLeftGap



e5071c431

	Chan
	(Integer)
	19

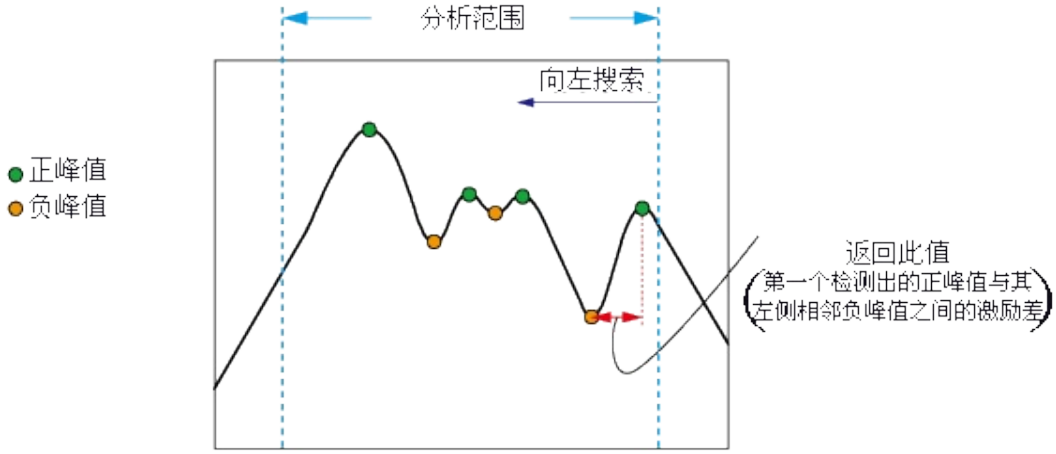
	Value
	(Double)
	0

```
Dim Value As Double

Value = FirstLeftGap(1)
MsgBox "First Left Gap =" & Value
```

FirstLeftInterval(*Chan*)

Value = FirstLeftInterval(*Chan*)

FirstLeftInterval



e5071c432

	Chan
	(Integer)
	19

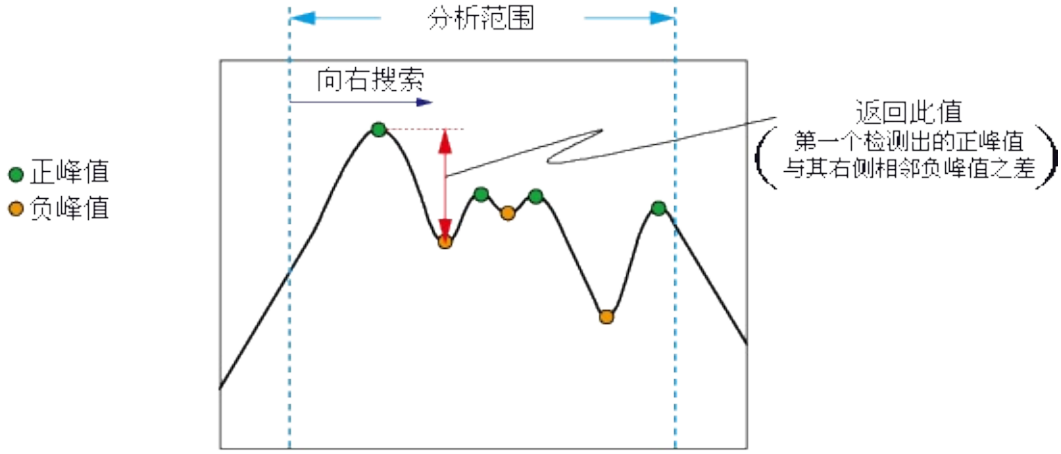
	Value
	(Double)
	0

```
Dim Value As Double

Value = FirstLeftInterval(1)
MsgBox "First Left Interval =" & Value
```

FirstRightGap(*Chan*)

Value = FirstRightGap(*Chan*)

FirstRightGap



e5071c434

	Chan
	(Integer)
	19

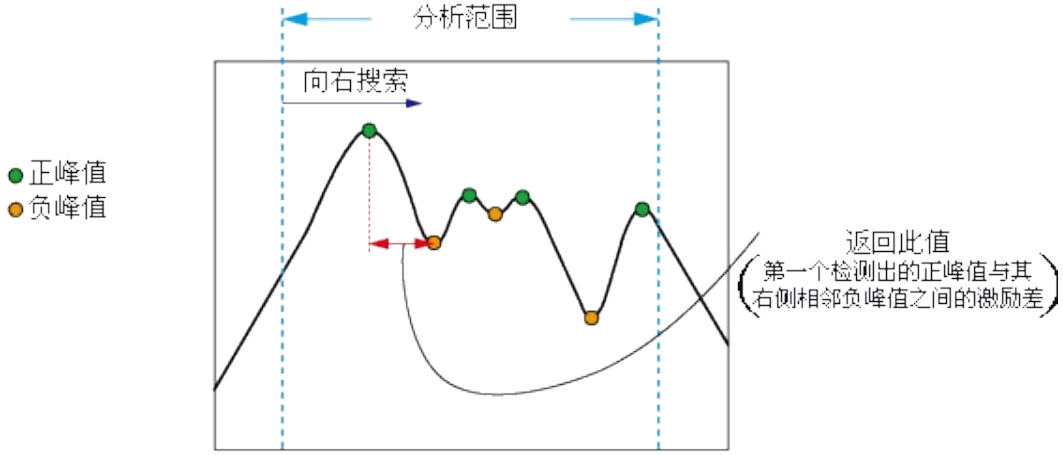
	Value
	(Double)
	0

```
Dim Value As Double
Value = FirstRightGap(1)
MsgBox "First Right Gap =" & Value
```

FirstRightInterval(*Chan*)

Value = FirstRightInterval(*Chan*)

FirstRightInterval



e5071c433

	Chan
	(Integer)
	19

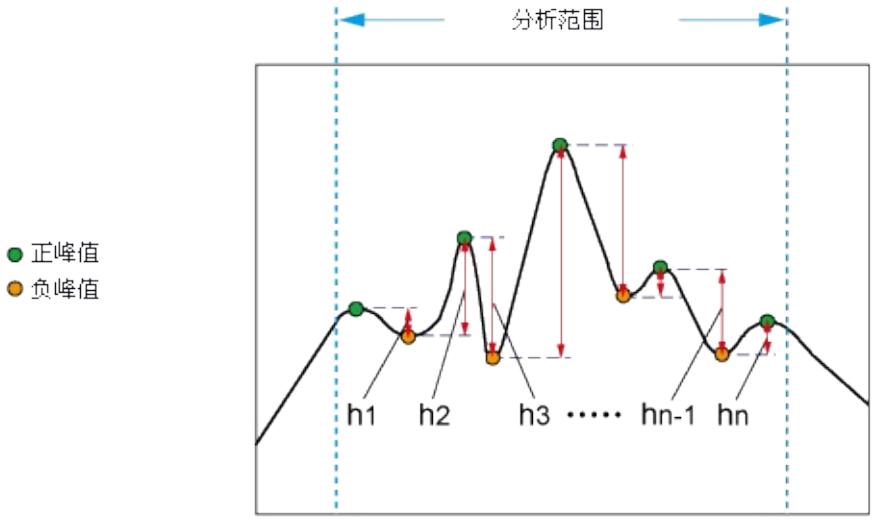
	Value
	(Double)
	0

```
Dim Value As Double

Value = FirstRightInterval(1)
MsgBox "First Right Interval =" & Value
```

GapMean(*Chan*)

Value = GapMean(*Chan*)

GapMean



返回如下算出的平均值：

$$\text{平均值} = \frac{h1 + h2 + h3 \cdots\cdots + hn\text{-}1 + hn}{n}$$

e5071c427

	Chan
	(Integer)
	19

	Value
	(Double)
	0

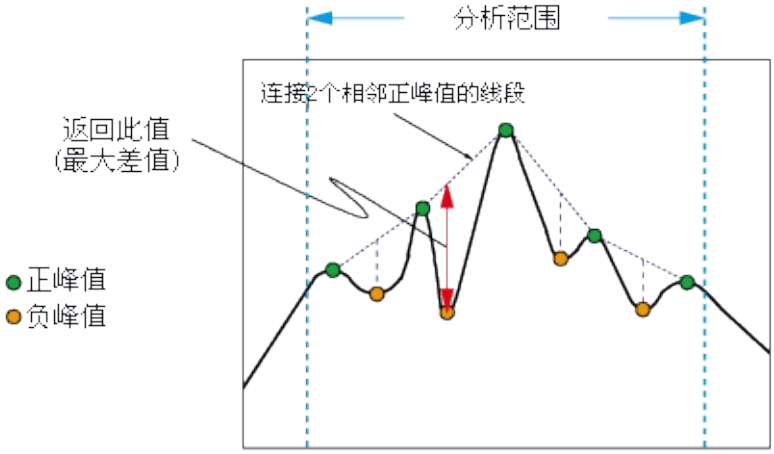
```
Dim Value As Double
Value = GapMean(1)
MsgBox "Gap Mean =" & Value
```

MaxEnvelopeGap(*Chan*)

Value = MaxEnvelopeGap(*Chan*)

2

MaxEnvelopeGap



e5071c426

	Chan
	(Integer)
	19

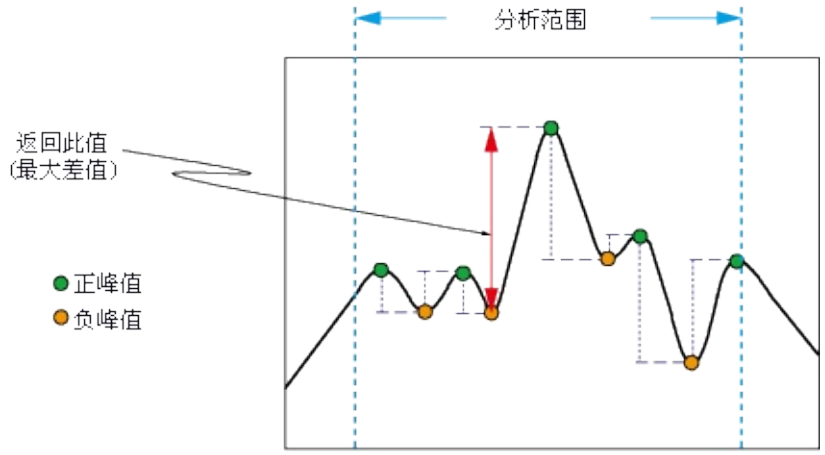
	Value
	2
	(Double)
	0

```
Dim Value As Double
Value = MaxEnvelopeGap(1)
MsgBox "Max Envelope Gap =" & Value
```

MaxGap(*Chan*)

Value = MaxGap(*Chan*)

MaxGap



e5071c425

	Chan
	(Integer)
	19

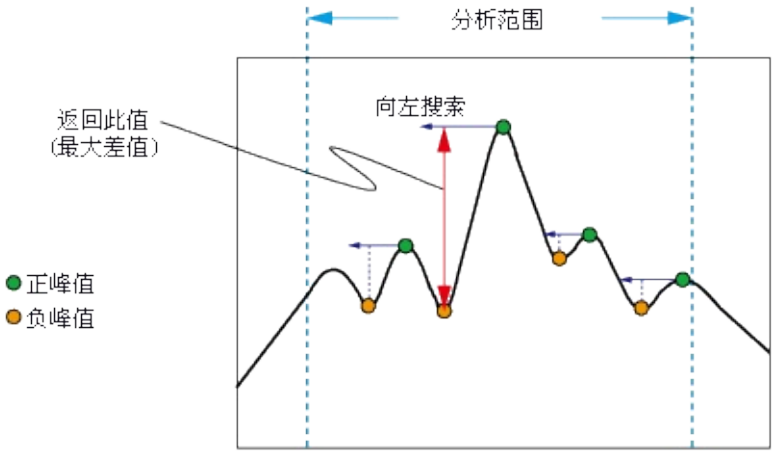
	Value
	(Double)
	0

```
Dim Value As Double
Value = MaxGap(1)
MsgBox "Max Gap =" & Value
```

MaxLeftGap(*Chan*)

Value = MaxLeftGap(*Chan*)

MaxLeftGap



e5071c424

	Chan
	(Integer)
	19

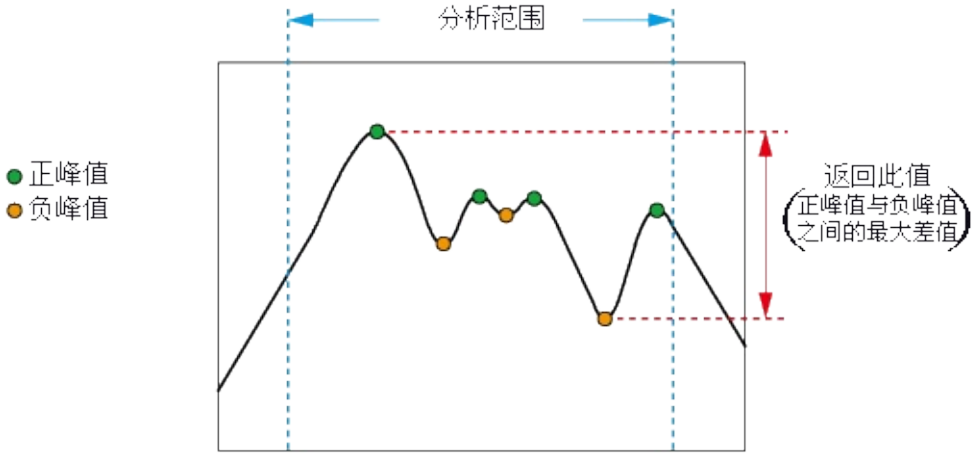
	Value
	(Double)
	0

```
Dim Value As Double
Value = MaxLeftGap(1)
MsgBox "Max Left Gap =" & Value
```

MaxPeakToPeak(*Chan*)

Value = MaxPeakToPeak(*Chan*)

MaxPeakToPeak



e5071c422

	Chan
	(Integer)
	19

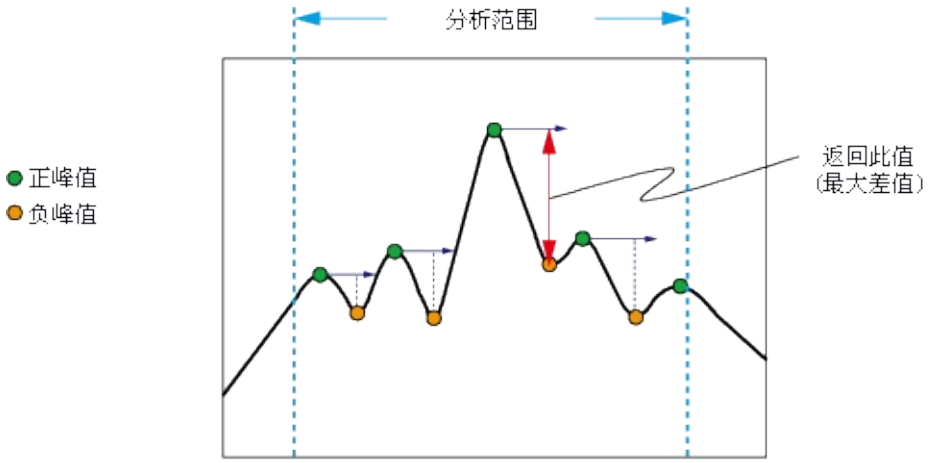
	Value
	(Double)
	0

```
Dim Value As Double
Value = MaxPeakToPeak(1)
MsgBox "Max Peak To Peak =" & Value
```

MaxRightGap(*Chan*)

Value = MaxRightGap(*chan*)

MaxRightGap



e5071c423

	Chan
	(Integer)
	19

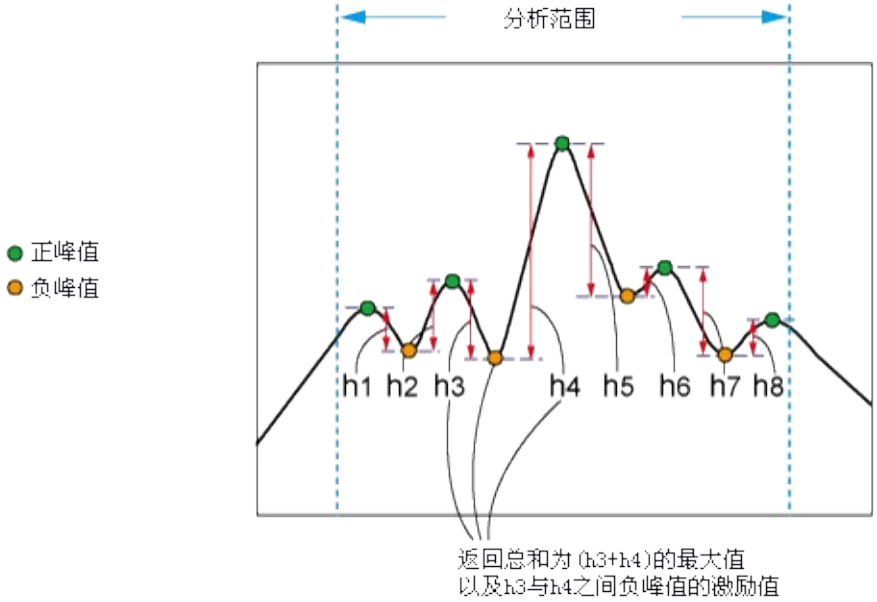
	Value
	(Double)
	0

```
Dim Value As Double

Value = MaxRightGap(1)
MsgBox "Max Right Gap =" & Value
```

MaxRipplePoint(*Chan*,*Stim*)

Value = MaxRipplePoint(*Chan,Stim*)

MaxRipplePoint



e5071c428

	Chan
	(Integer)
	19

	Value
	(Double)
	0

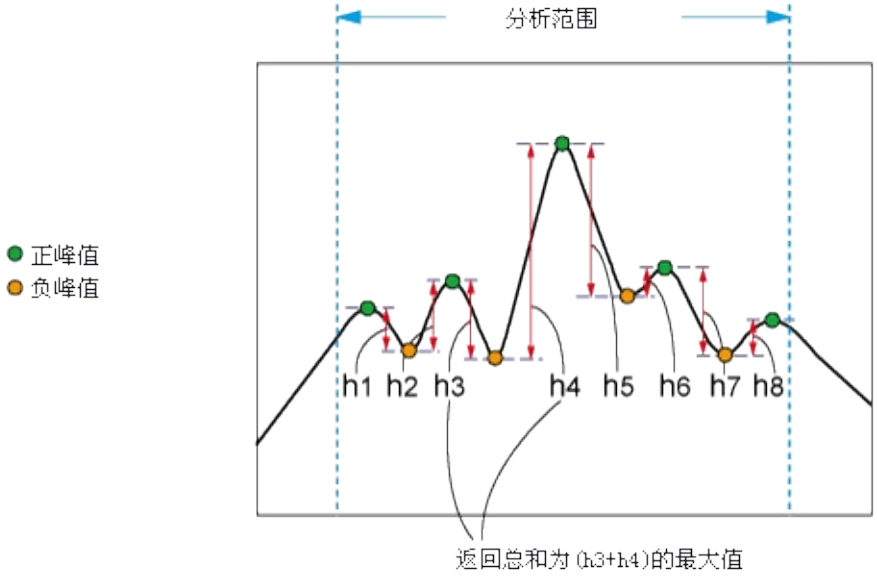
	Stim
	(Double)
	0

```
Dim Value As Double
Dim Stim As Double

Value = MaxRipplePoint(1, Stim)
MsgBox "Max Ripple Value =" & Value & " , Stimulus =" & Stim
```

MaxRippleValue(*Chan*)

Value = MaxRippleValue(*Chan*)

MaxRippleValue



e5071c429

	Chan
	(Integer)
	19

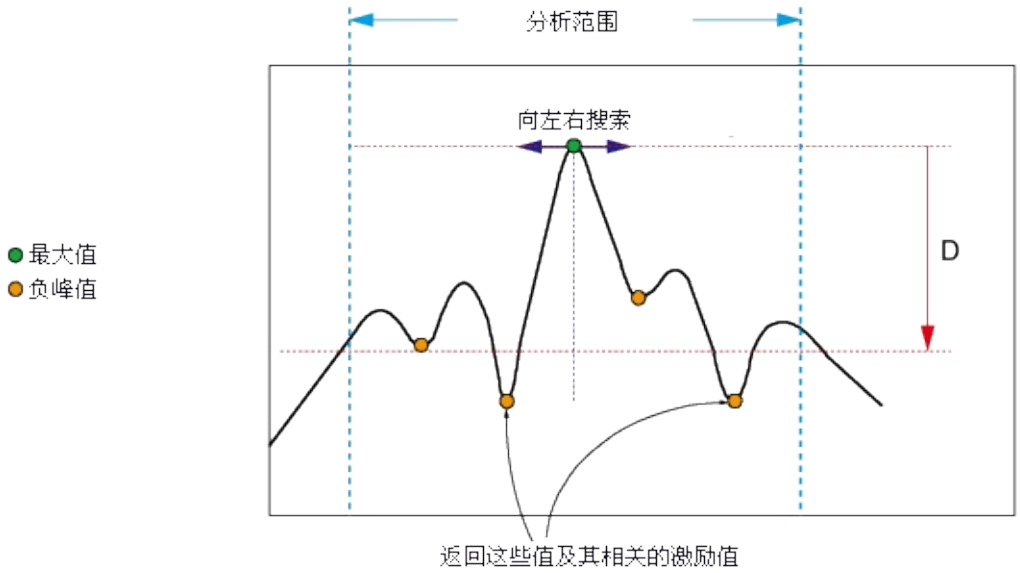
	Value
	(Double)
	0

```
Dim Value As Double
Value = MaxRippleValue(1)
MsgBox "Max Ripple Value =" & Value
```

Pole(*Chan***,***D***,***LeftStim***,***LeftValue***,***RightStim***,***RightValue***)**

Call Pole(*Chan*,*D*,*LeftStim*,*LeftValue*,*RightStim*,*RightValue*)

D *LeftValue* *LeftStimulus* *RightValue* *RightStimulus*



e5071c430

	Chan
	(Integer)
	19

	D
	(Double)

	LeftStim
	(Double) 0

	LeftValue
	(Double)
	0

	RightStim
	(Double) 0

	RightValue
	(Double) 0

```
Dim LeftStim As Double
Dim LeftValue As Double
Dim RightStim As Double
Dim RightValue As Double

Call Pole(1, 1, LeftStim, LeftValue, RightStim, RightValue)

MsgBox "Left Pole =" & LeftStim & ":" & LeftValue
MsgBox "Right Pole =" & RightStim & ":" & RightValue
```

-
- COM
-

E5071C COME5071CCOM/E5071CCOMCOM

E5071CCOM“ ”E5071CCOM“ ”E5071CE5071CCOM“ ”

E5071C VBAE5071CCOM/“”“ ”

3

“ ”	E5071C
	E5071C
“”	E5071C

COM

COM

String *Param*

E5071C COM5

Long		4	-21474836482147483 647
Double		8	1.79769313486232E+308 -4.94065645841247E-324 -1.79769313486232E+308 -4.94065645841247E-324
Boolean		2	COM SCPI
String		1/	20
Variant		16	

E5071C VBA

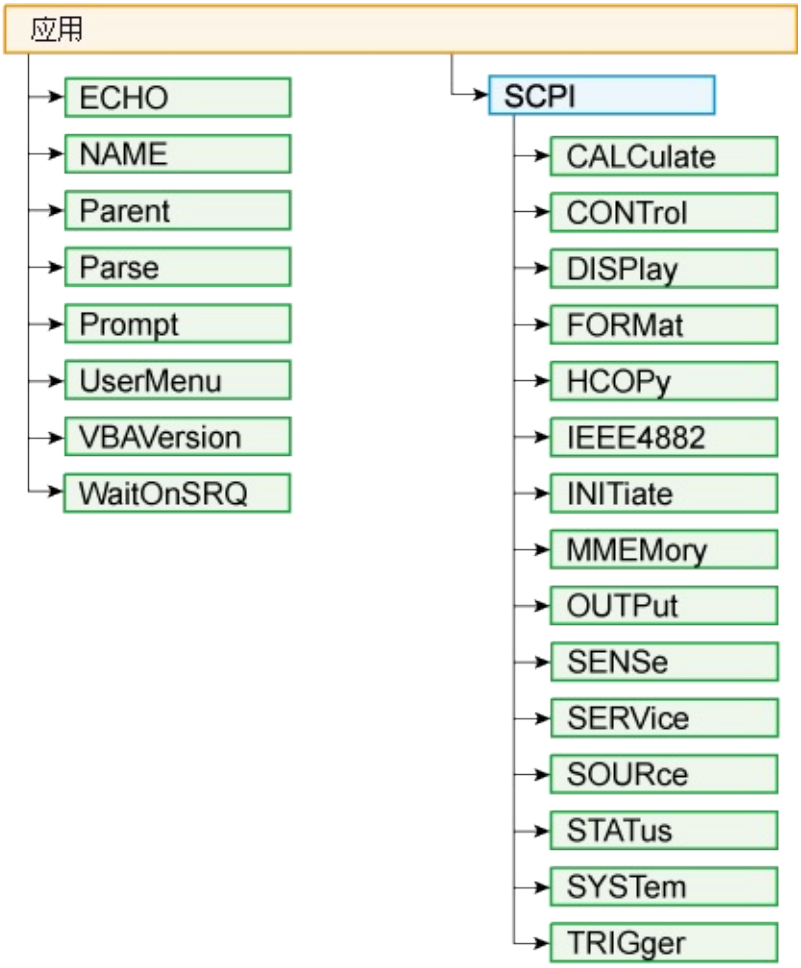
/

SCPI

SCPISCPI

COM

E5071CCOM



e5071c473

E5071C COMCOM7SCPISCPI

SCPI

SCPICOME5071CSCPI

SCPISCPI

- IEEESCPI.IEEE4882“*”
- .:
- T
- Y

SCPI	COM
OUTPUT 717;":SOUR1:POW -10"	SCPI.SOURce(1).POWer.L = -10
OUTPUT 717;":SENS1:CORR:COLL:METH:TYPE?" ENTER 717;A$	A = SCPI.SENSe(1).CORRecti
OUTPUT 717;"*CLS"	SCPI.IEEE4882.CLS

SCPI.ABORt

SCPI.ABORt

SCPI.ABORt

SCPI.INITiate(Ch).IMMediate

SCPI.INITiate(Ch).CONTinuous

Trigger > **Restart**

>

SCPI

:ABORt

```
10 OUTPUT 717;":ABORt"
```

ECHO

ECHO *V1,V2, ... ,V10*

ECHO *SCPI object*

echo

SCPI.DISPlay.ECHO.DATA

- 10
-

	V1,V2, **.....** *,V10*
	echo
	(Variant)

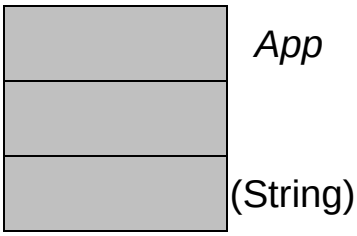
```
Dim Nop As Long
Dim i As Integer
Dim Fdata As Variant
Nop = SCPI.SENSe(1).SWEep.POINts
Fdata = SCPI.CALCulate(1).SELected.DATA.FDATa
ECHO "Test Results"
For i=1 to Nop
    ECHO i, Fdata(2*i-2), Fdata(2*i-1)
Next i

ECHO SCPI.SYSTem.ERRor
```

SCPI.DISPlay.ECHO.DATA

NAME

App = NAME

VBAE5071C


App
(String)

```
Dim Inst As String
Inst = NAME
ECHO Inst
```

Parse

Parse(*Scpi*)

Return = Parse(*Scpi?*)

E5071CSCPI

ParseSCPISCPICOM

	Scpi
	SCPI
	(String)

	Return
	SCPI
	(String)

```
Dim Start As String
Parse(":SENS1:FREQ:STAR 100E6")
Start = Parse(":SENS1:FREQ:STAR?")
```

```
Dim TtlLbl As String
Parse(":DISP:WIND1:TITL:DATA ""filter""")
TtlLbl = Parse(":DISP:WIND1:TITL:DATA?")
```

```
Dim Fmt As String
Parse(":CALC1:PAR2:SEL")
Parse(":CALC1:FORM SMIT")
Fmt = Parse(":CALC1:FORM?")
```

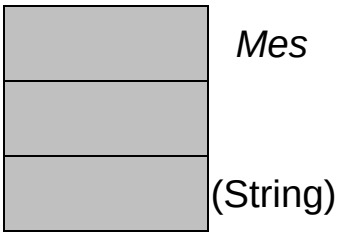
```
Dim BckLght As String
Parse(":SYST:BACK OFF")
BckLght = Parse(":SYST:BACK?")
```

Prompt

Prompt(*Mes*)

LCD **Macro Setup** > **Continue**

Macro Setup > **Continue**Visual Basic


Mes
(String)

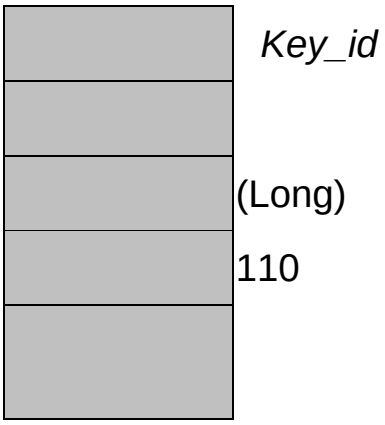
Prompt ("Connect DUT, and then press [Continue]")

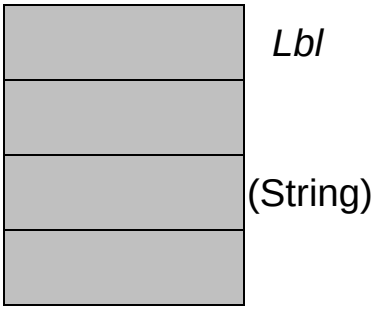
UserMenu.Item(*Key_id*).Caption

UserMenu.Item(*Key_id*).Caption = *Lbl*

Lbl = UserMenu.Item(*Key_id*).Caption

110 (*Key_id*)


Key_id
(Long)
110


Lbl
(String)

```
Dim KeyLbl As String
UserMenu.Item(1).Caption = "Meas"
KeyLbl = UserMenu.Item(1).Caption
```

UserMenu.Item(*Key_id*).Enabled

UserMenu.Item(*Key_id*).Enabled = *Status*

Status = UserMenu.Item(*Key_id*).Enabled

/110 *Key_id*

	Status
	/
	(Boolean)
	• -1 • 0
	-1

(*Key_id*) UserMenu.Item.Caption

```
Dim KeyEna As Boolean
UserMenu.Item(10).Enabled = False
KeyEna = UserMenu.Item(10).Enabled
```

UserMenu.Press

UserMenu.PRESet

UserMenu.PRESet

/

UserMenu.PRESet

UserMenu.Item.Caption
UserMenu.Item.Enabled

Macro Setup > Preset User Menu

UserMenu.Press(*Key_id***)**

UserMenu.Press(*Key_id*)

110 (*id*)

(*Key_id*) UserMenu.Item.Caption

UserMenu.Press(1)

UserMenu.Item.Enabled

Macro Setup > **User Menu** > **Button 1** to **Button 10**

UserMenu.Show

UserMenu.Show

UserMenu.Show

Macro Setup > User Menu

UserMenu_OnPress(ByVal *Key_id* As Long)

110 *Key_id*“”

(*Key_id*) UserMenu.Item.Caption

```
Private Sub UserMenu_OnPress(ByVal id As Long)
If id = 1 Then
MsgBox "Button 1 was pressed."
ElseIf id = 10 Then
MsgBox "Button 10 was pressed."
End If
End Sub
```

VBAVersion

Vers = VBAVersion

E5071CVBA

	Vers
	VBA
	(String)

```
Dim Version As String
Version = VBAVersion
ECHO Version
```

Visual Basic**About Microsoft Visual Basic...**

WaitOnSRQ

WaitOnSRQ *Status*, *Timeout*

RQS/MSS61

	Status
	RQS/MSS
	(Boolean)
	• -1 1 • 0 1

	Timeout
	(Long)
	02147483647
	-1
	ms

```
Dim Stat As Boolean
SCPI.IEEE4882.CLS
SCPI.STATus.OPERation.PTRansition = 0
SCPI.STATus.OPERation.NTRansition = 16
SCPI.STATus.OPERation.ENABle = 16
SCPI.IEEE4882.SRE = 128
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.INITiate(1).CONTinuous = True
SCPI.TRIGger.SEQuence.IMMediate
WaitOnSRQ Stat, 10000
If Stat = True Then
MsgBox "Done"
End If
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.BPOI

SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.BPORt(*Bpt*).IMAG
= *Value*

Value =
SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.BPORt(*Bpt*).IMAG

(*Ch*)12(*Bpt*)/

	Ch
	(Long)
	136
	1

	Bpt
	(Long)
	12
	1

	Value
	(Double)
	-1E+181E+18
	0
	ohm

```
Dim CImag As Double
SCPI.CALCulate(1).FSIMulator.BALun.CZConversion.BPORt(1).IMAGin
= 1E5
CImag =
SCPI.CALCulate(1).FSIMulator.BALun.CZConversion.BPORt(1).IMAGin
```

SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).REAl
SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).Z0.R
SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > **Fixture Simulator** > **Cmn ZConversion** > **Port1(bal) Imag|Port2(bal) Imag|Port3(bal) Imag**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:CZConversion:BPORt{[1]|2}:IMAGi <numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:CZConversion:BPORt{[1]|2}:IMAGi

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:CZC:BPOR1:IMAG 30E6"
20 OUTPUT 717;":CALC1:FSIM:BAL:CZC:BPOR1:IMAG?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.BPOI

SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.BPORt(*Bpt*).REAl = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.BPORt(*Bpt*).REAl

(*Ch*)12(*Bpt*)/

- SCPI.CALCulate(Ch).FSIM-ulator.BALun.CZConversion.BPORt(Bpt).Z0.R

	Value
	(Double)
	1E-31E7
	25
	ohm
	0.001

```
Dim CReal As Double
SCPI.CALCulate(1).FSIMulator.BALun.CZConversion.BPORt(1).REAL
= 30
CReal =
SCPI.CALCulate(1).FSIMulator.BALun.CZConversion.BPORt(1).REAL
```

SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).IMAG
SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).Z0.R
SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > **Fixture Simulator** > **Cmn ZConversion** > **Port1(bal) Real|Port2(bal) Real|Port3(bal) Real**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:CZConversion:BPORt{[1]|2}:REAL <numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:CZConversion:BPORt{[1]|2}:REAL?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:CZC:BPOR1:REAL 30E6"
20 OUTPUT 717;":CALC1:FSIM:BAL:CZC:BPOR1:REAL?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.BPOI

SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.BPORt(*Bpt*).Z0.R = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.BPORt(*Bpt*).Z0.R

(*Ch*)12(*Bpt*)/

- SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).IMAGinary

	Value
	(Double)
	1E-31E7
	25
	ohm
	0.001

```
Dim CZ0 As Double
SCPI.CALCulate(1).FSIMulator.BALun.CZConversion.BPORt(1).Z0.R =
30
CZ0 =
SCPI.CALCulate(1).FSIMulator.BALun.CZConversion.BPORt(1).Z0.R
```

SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).IMAG
SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).REAl
SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > **Fixture Simulator** > **Cmn ZConversion** > **Port1(bal) Real|Port2(bal) Real|Port3(bal) Real**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:CZConversion:BPORt{[1]|2}:Z0[:R]
<numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:CZConversion:BPORt{[1]|2}:Z0[:R]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:CZC:BPOR1:Z0 30"
20 OUTPUT 717;":CALC1:FSIM:BAL:CZC:BPOR1:Z0?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.STAT

SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.BALun.CZConversion.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim ComZcon As Boolean
SCPI.CALCulate(1).FSIMulator.BALun.CZConversion.STATe = True
ComZcon =
SCPI.CALCulate(1).FSIMulator.BALun.CZConversion.STATe
```

SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).IMAG
SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).REAl
SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).Z0.R
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > Fixture Simulator > Cmn ZConversion > Cmn ZConversion

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:CZConversion:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:BALun:CZConversion:STATe?

{1|0}<newline><^END>

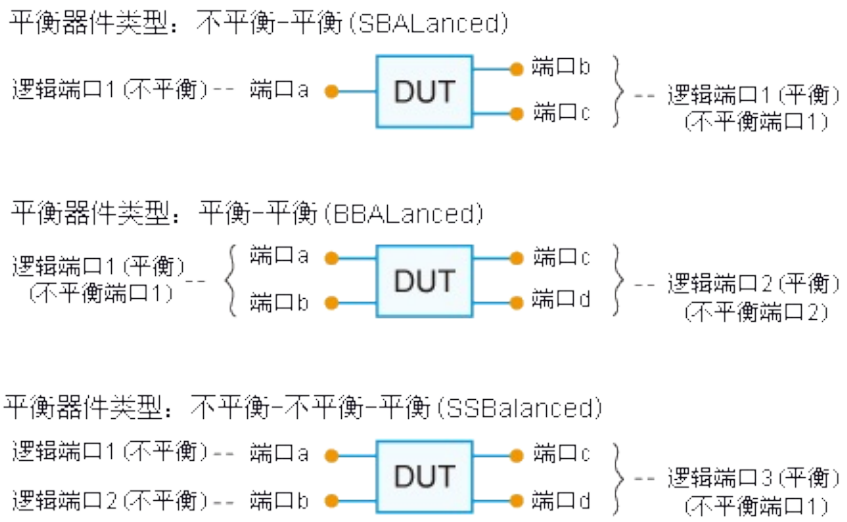
```
10 OUTPUT 717;":CALC1:FSIM:BAL:CZC:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:BAL:CZC:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DEVice

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DEVice = *Param*

Param = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DEVice

(Ch)



e5071c302

	Param
	(String)
	• “SBALanced”3 • “BBALanced”4 • 4
	“SBALanced”

```
Dim BalDev As String
SCPI.CALCulate(1).FSIMulator.BALun.DEVice = "bbal"
BalDev = SCPI.CALCulate(1).FSIMulator.BALun.DEVice
```

SCPI.CALCulate(Ch).FSIMulator.BALun.TOPology.BBALanced.PPORts
SCPI.CALCulate(Ch).FSIMulator.BALun.TOPology.SBALanced.PPORts
SCPI.CALCulate(Ch).FSIMulator.BALun.TOPology.SSBalanced.PPORts

SCPI

Analysis > Fixture Simulator > Topology > Device

Equivalent SCPI command

:CALCulate{[1]-36}:FSIMulator:BALun:DEVice {SBALanced|BBALanced|SSBalanced}

:CALCulate{[1]-36}:FSIMulator:BALun:DEVice?

{SBAL|BBAL|SSB}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DEV BBAL"
20 OUTPUT 717;":CALC1:FSIM:BAL:DEV?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.

BPORt(*Bpt*).PARameters.C

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).PARamet = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).PARamet

(*Ch*)12(*Bpt*)/LCPLPC
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit. BPORt(Bpt).TYPE
C

	Value
	C
	(Double)
	-1E181E18
	0
	F
	1E-18

```
Dim DmcC As Double
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).PARameters
= 12E-12
DmcC =
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).PARameters
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).TYPE
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:PARameters:C <numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:PARameters:C?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:C 12.3"
20 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:C?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*B*

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).PARamet = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).PARamet

(*Ch*)12(*Bpt*)/LCPLPC
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit. BPORt(Bpt).TYPE
G

	Value
	G
	(Double)
	-1E181E18
	0
	S
	1E-18

```
Dim DmcG As Double
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).PARameters
= 12E-12
DmcG =
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).PARameters
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).TYPE
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:PARameters:G <numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:PARameters:G?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:G 12.3"
20 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:G?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*B*

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).PARamet = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).PARamet

(*Ch*)12(*Bpt*)/LCPLPC
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit. BPORt(Bpt).TYPE
L

	Value
	L
	(Double)
	-1E181E18
	0
	H
	1E-18

```
Dim DmcL As Double
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).PARameters
= 12E-12
DmcL =
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).PARameters
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).TYPE
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:PARameters:L <numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:PARameters:L?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:L 12.3"
20 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:L?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*B*

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).PARamet = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).PARamet

(*Ch*)12(*Bpt*)/LCPLPC
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit. BPORt(Bpt).TYPE
R

	Value
	R
	(Double)
	-1E181E18
	0
	ohm
	1E-18

```
Dim DmcR As Double
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).PARameters
= 12E-12
DmcR =
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).PARameters
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).TYPE
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:PARameters:R <numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:PARameters:R?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:R 12.3"
20 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:PAR:R?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*B*

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).TYPE

(*Ch*)12(*Bpt*)

	Param
	(String)
	• “NONE” • “PLPC”LC • “USER”
	“NONE”
	2NONE

```
Dim CirType As String
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).TYPE =
"plpc"
CirType =
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).TYPE
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).USER.FIL
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.STATe

Analysis > **Fixture Simulator** > **Diff Matching** > **Select Circuit**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}[:TYPE] {NONE|PLPC|USER}

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}[:TYPE]?

{NONE|PLPC|USER}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1 PLPC"
20 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*B*

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).USER.FIL
= *File*

File =
SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.BPORt(*Bpt*).USER.FIL

(*Ch*)12(*Bpt*)2.s2p

“\”“/”

SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).TYPE

	File
	2.s2p
	(String)
	254
	“”

```
Dim DmcUser As String
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).USER.FILe
= "dmc.s2p"
DmcUser =
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).USER.FILe
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.BPORt(1).TYPE =
"user"
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).TYPE
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:USER:FILename <string>

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:BPORt{[1]|2}
:USER:FILename?

{string}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:USER:FIL
""Match_d.s2p""""
20 OUTPUT 717;":CALC1:FSIM:BAL:DMC:BPOR1:USER:FIL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.STATe

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DMCircuit.STATe

(Ch)/

Status

/

(Boolean)

-
-

```
Dim DifMch As Boolean
SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.STATe = True
DifMch = SCPI.CALCulate(1).FSIMulator.BALun.DMCircuit.STATe
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARamet
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).USER.FIL
SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).TYPE
SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:BALun:DMCircuit:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DMC:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:BAL:DMC:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.BPOI

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.BPORt(*Bpt*).IMAG
= *Value*

Value =
SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.BPORt(*Bpt*).IMAG

(*Ch*)12(*Bpt*)/

	Value
	(Double)
	-1E+181E+18
	0
	ohm

```
Dim DImag As Double
SCPI.CALCulate(1).FSIMulator.BALun.DZConversion.BPORt(1).IMAGin
= 200
DImag =
SCPI.CALCulate(1).FSIMulator.BALun.DZConversion.BPORt(1).IMAGin
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).REAI
SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).Z0.R
SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > **Fixture Simulator** > **Diff ZConversion** > **Port1(bal) Imag|Port2(bal) Imag|Port3(bal) Imag**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DZConversion
:BPORt{[1]|2}:IMAGinary <numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:DZConversion
:BPORt{[1]|2}:IMAGinary?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DZC:BPOR1:IMAG 300"
20 OUTPUT 717;":CALC1:FSIM:BAL:DZC:BPOR1:IMAG?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.BPOI

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.BPORt(*Bpt*).REAl = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.BPORt(*Bpt*).REAl

(*Ch*)12(*Bpt*)/

- “SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).Z0.R”

	Value
	(Double)
	1E-31E7
	100
	ohm
	0.001

```
Dim DReal As Double
SCPI.CALCulate(1).FSIMulator.BALun.DZConversion.BPORt(1).REAL
= 200
DReal =
SCPI.CALCulate(1).FSIMulator.BALun.DZConversion.BPORt(1).REAL
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).IMAG
SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).Z0.R
SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > **Fixture Simulator** > **Diff ZConversion** > **Port1(bal) Real|Port2(bal) Real|Port3(bal) Real**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DZConversion
:BPORt{[1]|2}:REAL <numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:DZConversion
:BPORt{[1]|2}:REAL?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DZC:BPOR1:REAL 300"
20 OUTPUT 717;":CALC1:FSIM:BAL:DZC:BPOR1:REAL?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.BPOI

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.BPORt(*Bpt*).Z0.R = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.BPORt(*Bpt*).Z0.R

(*Ch*)12(*Bpt*)/

- “SCPI.CALCulate(Ch).FSIMula-tor.BALun.DZConversion.BPORt(Bpt).IMAGinary”

	Value
	(Double)
	1E-31E7
	100
	ohm
	0.001

```
Dim DZ0 As Double
SCPI.CALCulate(1).FSIMulator.BALun.DZConversion.BPORt(1).Z0.R =
200
DZ0 =
SCPI.CALCulate(1).FSIMulator.BALun.DZConversion.BPORt(1).Z0.R
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).IMAG
SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).REAL
SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > **Fixture Simulator** > **Diff ZConversion** > **Port1(bal) Real|Port2(bal) Real|Port3(bal) Real**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DZConversion
:BPORt{[1]|2}:Z0[:R] <numeric>

:CALCulate{[1]-36}:FSIMulator:BALun:DZConversion
:BPORt{[1]|2}:Z0[:R]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DZC:BPOR1:Z0 300"
20 OUTPUT 717;":CALC1:FSIM:BAL:DZC:BPOR1:Z0?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.STAT

SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.BALun.DZConversion.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim DifZcon As Boolean
SCPI.CALCulate(1).FSIMulator.BALun.DZConversion.STATe = True
DifZcon = SCPI.CALCulate(1).FSIMulator.BALun.DZConversion.STATe
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).IMAG
SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).REAL
SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).Z0.R
SCPI.CALCulate(Ch).FSIMulator.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:DZConversion:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:BALun:DZConversion:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:DZC:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:BAL:DZC:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).BBA

SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).BBALanced.DEF = *Param*

Param = SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).BBALanced.DEF

“_____” (*Ch*)(*Tr*)/

	Param
	(String)
	• “SDD11”Specifies Sdd11 • “SDD21”Specifies Sdd21 • “SDD12”Specifies Sdd12 • “SDD22”Specifies Sdd22 • “SCD11”Specifies Scd11 • “SCD21”Specifies Scd21 • “SCD12”Specifies Scd12 • “SCD22”Specifies Scd22 • “SDC11”Specifies Sdc11 • “SDC21”Specifies Sdc21 • “SDC12”Specifies Sdc12 • “SDC22”Specifies Sdc22 • “SCC11”Specifies Scc11 • “SCC21”Specifies Scc21 • “SCC12”Specifies Scc12 • “SCC22”Specifies Scc22 • “IMB1”Specifies Imbalance1 • “IMB2”Specifies Imbalance2 • “CMRR”Specifies CMRR (Sdd21/Scc21)
	“SDD11”

```
Dim BbalPara As String
SCPI.CALCulate(1).FSIMulator.BALun.DEVice = "bbal"
SCPI.CALCulate(1).FSIMulator.BALun.PARameter(1).BBALanced.DEFi
= "sdd21"
BbalPara =
SCPI.CALCulate(1).FSIMulator.BALun.PARameter(1).BBALanced.DEFi
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice

Analysis > **Fixture Simulator|[Meas]** >
Sdd11|Sdd21|Sdd12|Sdd22|Scd11| Scd21|Scd12|Scd22|Sdc11|
Sdc21|Sdc12|Sdc22|Scc11|Scc21|Scc12|Scc22|
Imbalance1|Imbalance2|Sdd21/Scc21

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:PARameter{[1]-16}
:BBALanced[:DEFine]
{SDD11|SDD21|SDD12|SDD22|SCD11|SCD21|SCD12|SCD22|SDC11|SD
SCC21|SCC12|SCC22|IMB1|IMB2| CMRR}

:CALCulate{[1]-36}:FSIMulator:BALun:PARameter{[1]-16}
:BBALanced[:DEFine]?

{SDD11|SDD21|SDD12|SDD22|SCD11|SCD21|SCD12|SCD22|SDC11|SD
SCC11|SCC21|SCC12|SCC22|IMB1|IMB2|CMRR}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:PAR1:BBAL SDD21"
20 OUTPUT 717;":CALC1:FSIM:BAL:PAR1:BBAL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).SBAl

SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).SBALanced.DEF = *Param*

Param = SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).SBALanced.DEF

“_____” (*Ch*)(*Tr*)/

	Param
	(String)
	• “SSS11”Specifies Sss11 • “SDS21”Specifies Sds21 • “SSD12”Specifies Ssd12 • “SCS21”Specifies Scs21 • “SSC12”Specifies Ssc12 • “SDD22”Specifies Sdd22 • “SCD22”Specifies Scd22 • “SDC22”Specifies Sdc22 • “SCC22”Specifies Scc22 • “IMB”Specifies Imbalance • “CMRR”Specifies CMRR (Sds21/Scs21) • “CMRR2”Specifies CMRR2 (Ssd12/Ssc12)
	“SSS11”

```
Dim SbalPara As String
SCPI.CALCulate(1).FSIMulator.BALun.DEVice = "sbal"
SCPI.CALCulate(1).FSIMulator.BALun.PARameter(1).SBALanced.DEFi
= "scs21"
SbalPara =
SCPI.CALCulate(1).FSIMulator.BALun.PARameter(1).SBALanced.DEFi
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice

Analysis > **Fixture Simulator|[Meas]** >
Sss11|Sds21|Ssd12|Scs21|Ssc12|
Sdd22|Scd22|Sdc22|Scc22|Imbalance|Sds21/Scs21|Ssd12/Ssc12

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:PARameter{[1]-16}
:SBALanced[:DEFine] {SSS11|SDS21|
SSD12|SCS21|SSC12|SDD22|SCD22|SDC22|SCC22|IMB|CMRR|CMRR

:CALCulate{[1]-36}:FSIMulator:BALun:PARameter{[1]-16}
:SBALanced[:DEFine]?

{SSS11|SDS21|SSD12|SCS21|SSC12|SDD22|SCD22|SDC22|SCC22|IMI
<newline> <^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:PAR1:SBAL SDS21"
20 OUTPUT 717;":CALC1:FSIM:BAL:PAR1:SBAL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).SSBa

SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).SSBalanced.DEF = *Param*

Param = SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).SSBalanced.DEF

“_____” (*Ch*)(*Tr*)/

	Param
	(String)
	• “SSS11”Specifies Sss11 • “SSS21”Specifies Sss21 • “SSS12”Specifies Sss12 • “SSS22”Specifies Sss22 • “SDS31”Specifies Sds31 • “SDS32”Specifies Sds32 • “SSD13”Specifies Ssd13 • “SSD23”Specifies Ssd23 • “SCS31”Specifies Scs31 • “SCS32”Specifies Scs32 • “SSC13”Specifies Ssc13 • “SSC23”Specifies Ssc23 • “SDD33”Specifies Sdd33 • “SCD33”Specifies Scd33 • “SDC33”Specifies Sdc33 • “SCC33”Specifies Scc33 • “IMB1”Specifies Imbalance1 • “IMB2”Specifies Imbalance2 • “IMB3”Specifies Imbalance3 • “IMB4”Specifies Imbalance4 • “CMRR1”Specifies CMRR (Sds31/Scs31)

	• “CMRR2”Specifies CMRR (Sds32/Scs32)
	“SSS11”

```
Dim SsbPara As String
SCPI.CALCulate(1).FSIMulator.BALun.DEVice = "ssb"
SCPI.CALCulate(1).FSIMulator.BALun.PARameter(1).SSBalanced.DEFi
= "sds31"
SsbPara =
SCPI.CALCulate(1).FSIMulator.BALun.PARameter(1).SSBalanced.DEFi
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice

Analysis > **Fixture Simulator|[Meas]** >
Sss11|Sss21|Sss12|Sss22|Sds31|
Sds32|Ssd13|Ssd23|Scs31|Scs32|Ssc13|Ssc23|Sdd33|Scd33|Sdc33|S
Imbalance1|Imbalance2|Imbalance3|Imbalance4|Sds31/Scs31|Sds32/

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:PARameter{[1]-16}
:SSBalanced[:DEFine]
{SSS11|SSS21|SSS12|SSS22|SDS31|SDS32|SSD13|SSD23|SCS31|SCS
SDC33|SCC33|IMB1|IMB2|IMB3|IMB4|CMRR1|CMRR2}

:CALCulate{[1]-36}:FSIMulator:BALun:PARameter{[1]-16}
:SSBalanced[:DEFine]?

{SSS11|SSS21|SSS12|SSS22|SDS31|SDS32|SSD13|SSD23|SCS31|SCS
<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:PAR1:SSB SDS31"
20 OUTPUT 717;":CALC1:FSIM:BAL:PAR1:SSB?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).STATe

SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.BALun.PARameter(*Tr*).STATe

(*Ch*)(*Tr*)/——

	Status
	/——
	(Boolean)
	• —— • ——

```
Dim BalMode As Boolean
SCPI.CALCulate(1).FSIMulator.BALun.PARameter(1).STATe = True
BalMode =
SCPI.CALCulate(1).FSIMulator.BALun.PARameter(1).STATe
```

SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > Fixture Simulator > BalUn

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:PARameter{[1]-16}:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:BALun:PARameter{[1]-16}:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:PAR1:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:BAL:PAR1:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.BBALanc

SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.BBALanced.PPORts = *Ports*

Ports = SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.BBALanced.PPORts

“——” *(Ch)*

SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice BBAL“——”

	Ports
	4 • *0* a • *1* b • *2* c • *3* d 0
	(Variant)
	14
	01 /12 /23 /34
	1
	2

```
Dim BbalPort As Variant
SCPI.CALCulate(1).FSIMulator.BALun.DEVice = "bbal"
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.BBALanced.PPORts
= Array(3,4,1,2)
BbalPort =
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.BBALanced.PPORts
```

```
Dim BbalPort(3) As Variant
Dim Ref As Variant
BbalPort(0) = 3
BbalPort(1) = 4
BbalPort(2) = 1
BbalPort(3) = 2
SCPI.CALCulate(1).FSIMulator.BALun.DEVice = "bbal"
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.BBALanced.PPORts
= BbalPort
Ref =
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.BBALanced.PPORts
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice

Analysis > **Fixture Simulator** > **Topology** > **Port1(bal)**

Analysis > **Fixture Simulator** > **Topology** > **Port2(bal)**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:TOPology:BBALanced [:PPORts] <numeric 1>, <numeric 2>,<numeric 3>,<numeric 4>

:CALCulate{[1]-36}:FSIMulator:BALun:TOPology:BBALanced [:PPORts]?

{numeric 1},{numeric 2},{numeric 3},{numeric 4}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:TOP:BBAL 1,2,3,4"
20 OUTPUT 717;":CALC1:FSIM:BAL:TOP:BBAL?"
30 ENTER 717;A,B,C,D
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.PROPerty

SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.PROPerty.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.PROPerty.STATe

—— *(Ch)*/

	Status
	/
	(Boolean)
	• •

```
Dim TopProp As Boolean
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.PROPerty.STATe =
True
TopProp =
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.PROPerty.STATe
```

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:TOPology
:PROPerty:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:BALun:TOPology
:PROPerty:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:TOP:PROP:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:BAL:TOP:PROP:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.SBALanc

SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.SBALanced.PPORts = *Ports*

Ports = SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.SBALanced.PPORts

“——” *(Ch)*

SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice SBAL“——”

	Ports
	3 • *0* a • *1* b • *2* c 0
	(Variant)
	14
	01 /12 /23
	1
	2

```
Dim SbalPort As Variant
SCPI.CALCulate(1).FSIMulator.BALun.DEVice = "sbal"
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.SBALanced.PPORts
= Array(1,3,4)
SbalPort =
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.BBALanced.PPORts
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice

Analysis > **Fixture Simulator** > **Topology** > **Port1(se)**

Analysis > **Fixture Simulator** > **Topology** > **Port2(bal)**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:TOPology:SBALanced [:PPORts] <numeric 1>, <numeric 2>,<numeric 3>

:CALCulate{[1]-36}:FSIMulator:BALun:TOPology:SBALanced [:PPORts]?

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:TOP:SBAL 1,2,3"
20 OUTPUT 717;":CALC1:FSIM:BAL:TOP:SBAL?"
30 ENTER 717;A,B,C
```

SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.SSBalanc

SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.SSBalanced.PPORts = *Ports*

Ports = SCPI.CALCulate(*Ch*).FSIMulator.BALun.TOPology.SSBalanced.PPORts

“————” (*Ch*)/
SCPI.CALCulate(Ch).FSIMulator.BALun.DEViceSBAL“————”

	Ports
	4 • *0* a • *1* b • *2* c • *3* d 0
	(Variant)
	14
	01 /12 /23 /34
	1
	2

```
Dim SsbPort As Variant
SCPI.CALCulate(1).FSIMulator.BALun.DEVice = "ssb"
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.SSBalanced.PPORts
= Array(1,4,2,3)
SsbPort =
SCPI.CALCulate(1).FSIMulator.BALun.TOPology.SSBalanced.PPORts
```

SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice

Analysis > **Fixture Simulator** > **Topology** > **Port1(se)**
Analysis > **Fixture Simulator** > **Topology** > **Port2(se)**
Analysis > **Fixture Simulator** > **Topology** > **Port3(bal)**

SCPI

:CALCulate{[1]-36}:FSIMulator:BALun:TOPology:SSBalanced [:PPORts] <numeric 1>, <numeric 2>,<numeric 3>,<numeric 4>

:CALCulate{[1]-36}:FSIMulator:BALun:TOPology:SSBalanced [:PPORts]?

{numeric 1},{numeric 2},{numeric 3},{numeric 4}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:BAL:TOP:SSB 1,2,3,4"
20 OUTPUT 717;":CALC1:FSIM:BAL:TOP:SSB?"
30 ENTER 717;A,B,C,D
```

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.NETWork(*Nwk*).FILe

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.NETWork(*Nwk*).FILename = *File*

File = SCPI.CALCulate(*Ch*).FSIMulator.EMBed.NETWork(*Nwk*).FILename

(*Ch*)12(*Nwk*)4//4“.s4p”

“\”“/”

3.50

	Nwk
	(Long)
	12
	1

	File
	4/4.s4p
	(String)
	254
	“”
	NONE SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).TYPE /

```
Dim Emb As String
SCPI.CALCulate(1).FSIMulator.EMBed.NETWork(1).FILename =
"network.s4p"
Emb = SCPI.CALCulate(1).FSIMulator.EMBed.NETWork(1).FILename
```

SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).TYPE
SCPI.CALCulate(Ch).FSIMulator.EMBed.STATe

Analysis > **Fixture Simulator** > **De-Embedding S4P** > **Topology** > **User File (nwk1)|User File (nwk2)**

SCPI

:CALCulate{[1]-36}:FSIMulator:EMBed:NETWork{[1]|2}: FILename <string>

:CALCulate{[1]-36}:FSIMulator:EMBed:NETWork{[1]|2}: FILename?

{string}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:EMB:NETW1:FIL ""Network.s4p"""
20 OUTPUT 717;":CALC1:FSIM:EMB:NETW1:FIL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.NETWork(*Nwk*).TYP

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.NETWork(*Nwk*).TYPE = *File*

File = SCPI.CALCulate(*Ch*).FSIMulator.EMBed.NETWork(*Nwk*).TYPE

/12 *(Nwk)* *(Ch)*4/

3.50

	Param
	(String)
	• “NONE” • “EMBed” • “DEEMbed”
	“NONE”
	/ SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).FILename4/NONE

```
Dim EmbType As String
SCPI.CALCulate(1).FSIMulator.EMBed.NETWork(1).FILename =
"network.s4p"
SCPI.CALCulate(1).FSIMulator.EMBed.NETWork(1).TYPE = "deem"
EmbType = SCPI.CALCulate(1).FSIMulator.EMBed.NETWork(1).TYPE
```

SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).FILename
SCPI.CALCulate(Ch).FSIMulator.EMBed.STATe

Analysis > **Fixture Simulator** > **De-Embedding S4P** > **Topology** > **Type (nwk1)|Type (nwk2)** > **None|Embed|De-Embed**

SCPI

:CALCulate{[1]-36}:FSIMulator:EMBed:NETWork{[1]|2}:TYPE {NONE|EMBed|DEEMbed}

:CALCulate{[1]-36}:FSIMulator:EMBed:NETWork{[1]|2}:TYPE?

{NONE|EMB|DEEM}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:EMB:NETW1:TYPE DEEM"
20 OUTPUT 717;":CALC1:FSIM:EMB:NETW1:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.STATe

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.EMBed.STATe

(Ch)/4/

	Status
	/4/
	(Boolean)
	• 4/ • 4/
	0

```
Dim Emb As Boolean
SCPI.CALCulate(1).FSIMulator.EMBed.STATe = True
Emb = SCPI.CALCulate(1).FSIMulator.EMBed.STATe
```

SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).FILename
SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).TYPE
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > Fixture Simulator > De-Embedding S4P > De-Embedding S4P

SCPI

:CALCulate{[1]-36}:FSIMulator:EMBed:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:EMBed:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:EMB:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:EMB:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TOPology.A.PORTs

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TOPology.A.PORTs = *Ports*

Ports = SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TOPology.A.PORTs

A/ (*Ch*)4/

	Ports
	2 • *0* a • *1* b 0
	(Variant)
	14
	01 /12
	1

```
Dim EnbPort As Variant
SCPI.CALCulate(1).FSIMulator.ENBed.TYPE = "a"
SCPI.CALCulate(1).FSIMulator.ENBed.TOPology.A.PORTs = Array(2,1)
EnbPort = SCPI.CALCulate(1).FSIMulator.ENBed.TOPology.A.PORTs
```

SCPI.CALCulate(Ch).FSIMulator.EMBed.TYPE

Analysis > **Fixture Simulator** > **De-Embedding S4P** > **Topology** > **Ports** >
1-2|1-3|1-4|2-1|2-3|2-4|3-1|3-2|3-4|4-1|4-2|4-3

SCPI

:CALCulate{[1]-36}:FSIMulator:EMBed:TOPology:A:PORTs <value 1>, <value 2>

:CALCulate{[1]-36}:FSIMulator:EMBed:TOPology:A:PORTs?

{value 1},{value 2}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:EMB:TOP:A:PORT 1,2"
20 OUTPUT 717;":CALC1:FSIM:EMB:TOP:A:PORT?"
30 ENTER 717;A,B
```

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TOPology.B.PORTs

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TOPology.B.PORTs = *Ports*

Ports = SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TOPology.B.PORTs

B/ (*Ch*)4/

	Ports
	3 • *0* a • *1* b • *2* c 0
	(Variant)
	14
	01 /12 /23
	1

```
Dim EnbPort As Variant
SCPI.CALCulate(1).FSIMulator.ENBed.TYPE = "b"
SCPI.CALCulate(1).FSIMulator.ENBed.TOPology.B.PORTs =
Array(1,3,2)
EnbPort = SCPI.CALCulate(1).FSIMulator.ENBed.TOPology.B.PORTs
```

SCPI.CALCulate(Ch).FSIMulator.EMBed.TYPE

Analysis > **Fixture Simulator** > **De-Embedding S4P** > **Topology** > **Ports** >
1-2-3|1-2-4|1-3-2|1-3-4|1-4-2|1-4-3|2-1-3|2-1-4|2-3-1|2-3-4|2-4-1|2-4-3| 3-1-2|3-1-4|3-2-1|3-2-4|3-4-1|3-4-2|4-1-2|4-1-3|4-2-1|4-2-3|4-3-1|4-3-2

SCPI

:CALCulate{[1]-36}:FSIMulator:EMBed:TOPology:B:PORTs <value 1>, <value 2>,<value 3>

:CALCulate{[1]-36}:FSIMulator:EMBed:TOPology:B:PORTs?

{value 1},{value 2},{value 3}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:EMB:TOP:B:PORT 1,2,3"
20 OUTPUT 717;":CALC1:FSIM:EMB:TOP:B:PORT?"
30 ENTER 717;A,B,C
```

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TOPology.C.PORTs

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TOPology.C.PORTs = *Ports*
Ports = SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TOPology.C.PORTs

C/ (*Ch*)4/

	Ports
	4 • *0* a • *1* b • *2* c • *3* d 0
	(Variant)
	14
	01 /12 /23 /34
	1

```
Dim EnbPort As Variant
SCPI.CALCulate(1).FSIMulator.ENBed.TYPE = "c"
SCPI.CALCulate(1).FSIMulator.ENBed.TOPology.C.PORTs =
Array(1,4,2,3)
EnbPort = SCPI.CALCulate(1).FSIMulator.ENBed.TOPology.C.PORTs
```

SCPI.CALCulate(Ch).FSIMulator.EMBed.TYPE

Analysis > **Fixture Simulator** > **De-Embedding S4P** > **Topology** > **Ports** >
1-2-3-4|1-2-4-3|1-3-2-4|1-3-4-2|1-4-2-3|1-4-3-2|2-1-3-4|2-1-4-3|2-3-1-4|2-3-4-1| 2-4-1-3|2-4-3-1|3-1-2-4|3-1-4-2|3-2-1-4|3-2-4-1|3-4-1-2|3-4-2-1|4-1-2-3|4-1-3-2| 4-2-1-3|4-2-3-1|4-3-1-2|4-3-2-1

SCPI

:CALCulate{[1]-36}:FSIMulator:EMBed:TOPology:C:PORTs <value 1>,<value 2>,<value 3>,<value 4>

:CALCulate{[1]-36}:FSIMulator:EMBed:TOPology:C:PORTs?

{value 1},{value 2},{value 3},{value 4}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:EMB:TOP:C:PORT 1,2,3,4"
20 OUTPUT 717;":CALC1:FSIM:EMB:TOP:C:PORT?"
30 ENTER 717;A,B,C,D
```

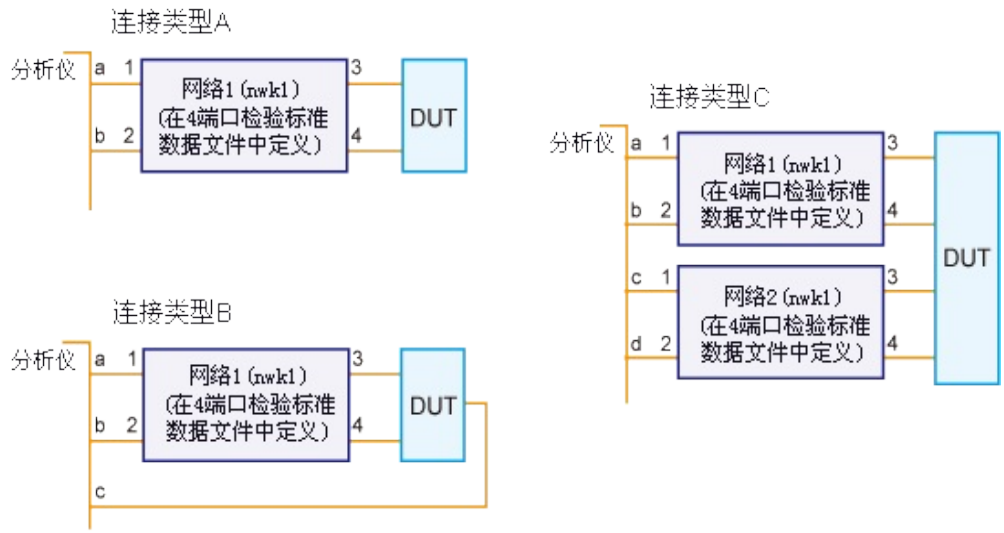
SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TYPE

SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TYPE = *File*

File = SCPI.CALCulate(*Ch*).FSIMulator.EMBed.TYPE

3.50

Connection Type



e5071c313

	Param
	(String)
	• “A”A • “B”B • “C”C
	“A”

```
Dim EmbType As String
SCPI.CALCulate(1).FSIMulator.EMBed.TYPE = "b"
EmbType = SCPI.CALCulate(1).FSIMulator.EMBed.TYPE
```

SCPI.CALCulate(Ch).FSIMulator.EMBed.TOPology.A.PORTs
SCPI.CALCulate(Ch).FSIMulator.EMBed.TOPology.B.PORTs
SCPI.CALCulate(Ch).FSIMulator.EMBed.TOPology.C.PORTs

Analysis > **Fixture Simulator** > **De-Embedding S4P** > **Topology** > **Select Topology** > **A|B|C**

SCPI

:CALCulate{[1]-36}:FSIMulator:EMBed:TYPE {A|B|C}

:CALCulate{[1]-36}:FSIMulator:EMBed:TYPE?

{A|B|C}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:EMB:TYPE A"
20 OUTPUT 717;":CALC1:FSIM:EMB:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.DEEMbed.PORT(*P*

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.DEEMbed.PORT(*Pt*).TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).FSIMulator.SENDed.DEEMbed.PORT(*Pt*).TYPE

(*Ch*)14(*Pt*)/

	Pt
	(Long)
	14
	1

	Param
	(String)
	• “NONE” • “USER”
	“NONE”
	2NONE

```
Dim DeemType As String
SCPI.CALCulate(1).FSIMulator.SENDed.DEEMbed.PORT(1).USER.FIL
= "network.s2p"
SCPI.CALCulate(1).FSIMulator.SENDed.DEEMbed.PORT(1).TYPE =
"user"
DeemType =
SCPI.CALCulate(1).FSIMulator.SENDed.DEEMbed.PORT(1).TYPE
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.PORT(Pt).USER.FIl
SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.STATe
SCPI.SENSe(Ch).CORRection.COLLect.ECAL.ORIentation.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:DEEMbed:PORT{[1]|2|3|4}[:TYPE] {NONE|USER}

:CALCulate{[1]-36}:FSIMulator:SENDed:DEEMbed:PORT{[1]|2|3|4}[:TYPE]?

{NONE|USER}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:DEEM:PORT1 USER"
20 OUTPUT 717;":CALC1:FSIM:SEND:DEEM:PORT1?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.DEEMbed.PORT(*P*

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.DEEMbed.PORT(*Pt*).USER.FIl
= *File*

File =
SCPI.CALCulate(*Ch*).FSIMulator.SENDed.DEEMbed.PORT(*Pt*).USER.FIl

(*Ch*)14(*Pt*)2“.s2p”

“\”“/”

SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.PORT(Pt).TYPE

	File
	2.s2p
	(String)
	254
	“”

```
Dim DeemUser As String
SCPI.CALCulate(1).FSIMulator.SENDed.DEEMbed.PORT(1).USER.FIL
= "network.s2p"
DeemUser =
SCPI.CALCulate(1).FSIMulator.SENDed.DEEMbed.PORT(1).USER.FIL
SCPI.CALCulate(1).FSIMulator.SENDed.DEEMbed.PORT(1).TYPE =
"user"
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.PORT(Pt).TYPE
SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:DEEMbed:PORT{[1]|2|3|4}:USER:FILename <string>

:CALCulate{[1]-36}:FSIMulator:SENDed:DEEMbed:PORT{[1]|2|3|4}:USER:FILename?

{string}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:DEEM:PORT1:USER:FIL
""Network.s2p""""
20 OUTPUT 717;":CALC1:FSIM:SEND:DEEM:PORT1:USER:FIL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.DEEMbed.STATe

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.DEEMbed.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.SENDed.DEEMbed.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •
	0

```
Dim Deemb As Boolean
SCPI.CALCulate(1).FSIMulator.SENDed.DEEMbed.STATe = True
Deemb = SCPI.CALCulate(1).FSIMulator.SENDed.DEEMbed.STATe
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.PORT(Pt).USER.FIl
SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.PORT(Pt).TYPE
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > Fixture Simulator > De-Embedding > De-Embedding

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:DEEMbed:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:SENDed:DEEMbed:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:DEEM:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:SEND:DEEM:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*P*

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).PARamet
= *Value*

Value =
SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).PARamet

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit. PORT(Pt).TYPEC
(*Ch*)14(*Pt*)

	Value
	C
	(Double)
	-1E181E18
	0
	F
	1E-18

```
Dim PmcC As Double
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).PARamete
= 12E-12
PmcC =
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).PARamete
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).TYPE
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:PARa
<numeric>

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:PARa

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:PAR:C 12.3"
20 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:PAR:C?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*P*

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).PARamet
= *Value*

Value =
SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).PARamet

/ SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit. PORT(Pt).TYPE
G (*Ch*)14(*Pt*)

	Value
	G
	(Double)
	-1E181E18
	0
	S
	1E-18

```
Dim PmcG As Double
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).PARamete
= 12E-12
PmcG =
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).PARamete
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).TYPE
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:PARa
<numeric>

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:PARa

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:PAR:G 12.3"
20 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:PAR:G?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*P*

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).PARamet
= *Value*

Value =
SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).PARamet

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit. PORT(Pt).TYPEL
(*Ch*)14(*Pt*)

	Value
	L
	(Double)
	-1E181E18
	0
	H
	1E-18

```
Dim PmcL As Double
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).PARamete
= 12E-12
PmcL =
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).PARamete
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).TYPE
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:PARa
<numeric>

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:PARa

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:PAR:L 12.3"
20 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:PAR:L?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*P*

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).PARamete
= *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).
PARameters.R

/ SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit. PORT(Pt).TYPE
R (*Ch*)14(*Pt*)

	Value
	R
	(Double)
	-1E181E18
	0
	ohm
	1E-18

```
Dim PmcR As Double
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).PARamete
= 12E-12
PmcR =
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).PARamete
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).TYPE
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:PARa
<numeric>

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:PARa

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:PAR:R 12.3"
20 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:PAR:R?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*P*

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).TYPE

*(Ch)*14*(Pt)*

	Param
	(String)
	• “NONE” • “SLPC”LC • “PCSL”CL • “PLSC”LC • “SCPL”CL • “PLPC”LC • “USER”
	“NONE”
	2NONE

```
Dim CirType As String
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).TYPE =
"slpc"
CirType =
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).TYPE
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).USER.FIL
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.STATe

Analysis > **Fixture Simulator** > **Port Matching** > **Select Circuit**

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}[:TYPE] {NONE| SLPC|PCSL|PLSC|SCPL|PLPC|USER}

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}[:TYPE]?

{NONE|SLPC|PCSL|PLSC|SCPL|PLPC|USER}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1 SLPC"
20 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*P*

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).USER.FIL
= *File*

File =
SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.PORT(*Pt*).USER.FIL

(*Ch*)14(*Pt*)2

“\”“/”

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).TYPE

	File
	2.s2p
	(String)
	254
	“”

```
Dim PmcUser As String
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).USER.FILe
= "match.s2p"
PmcUser =
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).USER.FILe
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.PORT(1).TYPE =
"user"
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).TYPE
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:USER <string>

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:PORT{[1]|2|3|4}:USER

{string}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:USER:FIL
""Match.s2p""""
20 OUTPUT 717;":CALC1:FSIM:SEND:PMC:PORT1:USER:FIL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.STATe

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.SENDed.PMCircuit.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim Pmcir As Boolean
SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.STATe = True
Pmcir = SCPI.CALCulate(1).FSIMulator.SENDed.PMCircuit.STATe
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).TYPE
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARamet
SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).USER.FIL
SCPI.CALCulate(Ch).FSIMulator.STATe

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:SENDed:PMCircuit:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:PMC:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:SEND:PMC:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.POR

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.PORT(*Pt*).IMAG = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.PORT(*Pt*).IMAG

(*Ch*)14(*Pt*)/

	Value
	(Double)
	-1E+181E+18
	0
	ohm

```
Dim ZImag As Double
SCPI.CALCulate(1).FSIMulator.SENDed.ZCONversion.PORT(1).IMAGir
= -9.2E10
ZImag =
SCPI.CALCulate(1).FSIMulator.SENDed.ZCONversion.PORT(1).IMAGir
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).REAL
SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).Z0.R
SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > Fixture Simulator > Port ZConversion > Port1 Z0 Imag|Port2 Z0 Imag|Port3 Z0 Imag|Port4 Z0 Imag

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:ZCONversion
:PORT{[1]|2|3|4}:IMAGinary <numeric>

:CALCulate{[1]-36}:FSIMulator:SENDed:ZCONversion
:PORT{[1]|2|3|4}:IMAGinary?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:ZCON:PORT1:IMAG 75"
20 OUTPUT 717;":CALC1:FSIM:SEND:ZCON:PORT1:IMAG?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.POR

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.PORT(*Pt*).REAL = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.PORT(*Pt*).REAL

(*Ch*)14(*Pt*)/

- “SCPI.CALCulate(Ch).FSIM-ulator.SENDed.ZCONversion.PORT(Pt).Z0.R”

	Value
	(Double)
	0.0011E7
	50
	ohm
	0.001

```
Dim ZReal As Double
SCPI.CALCulate(1).FSIMulator.SENDed.ZCONversion.PORT(1).REAL
= 3.7E5
ZReal =
SCPI.CALCulate(1).FSIMulator.SENDed.ZCONversion.PORT(1).REAL
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).IMAG
SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).Z0.R
SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > **Fixture Simulator** > **Port ZConversion** > **Port1 Z0 Real|Port2 Z0 Real|Port3 Z0 Real|Port4 Z0 Real**

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:ZCONversion
:PORT{[1]|2|3|4}:REAL <numeric>

:CALCulate{[1]-36}:FSIMulator:SENDed:ZCONversion
:PORT{[1]|2|3|4}:REAL?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:ZCON:PORT1:REAL 75"
20 OUTPUT 717;":CALC1:FSIM:SEND:ZCON:PORT1:REAL?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.POR

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.PORT(*Pt*).Z0.R = *Value*

Value = SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.PORT(*Pt*).Z0.R

(*Ch*)14(*Pt*)/

SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).IMAGinary

	Value
	(Double)
	0.0011E7
	50
	ohm
	0.001

```
Dim ZconR As Double
SCPI.CALCulate(1).FSIMulator.SENDed.ZCONversion.PORT(1).Z0.R =
75
ZconR =
SCPI.CALCulate(1).FSIMulator.SENDed.ZCONversion.PORT(1).Z0.R
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).IMAG
SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).REAL
SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.STATe
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > **Fixture Simulator** > **Port ZConversion** > **Port1 Z0 Real|Port2 Z0 Real|Port3 Z0 Real|Port4 Z0 Real**

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:ZCONversion
:PORT{[1]|2|3|4}:Z0[:R] <numeric>

:CALCulate{[1]-36}:FSIMulator:SENDed:ZCONversion
:PORT{[1]|2|3|4}:Z0[:R]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:ZCON:PORT1:Z0 75"
20 OUTPUT 717;":CALC1:FSIM:SEND:ZCON:PORT1:Z0?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.STAT

SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.SENDed.ZCONversion.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim Zcon As Boolean
SCPI.CALCulate(1).FSIMulator.SENDed.ZCONversion.STATe = True
Zcon = SCPI.CALCulate(1).FSIMulator.SENDed.ZCONversion.STATe
```

SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).IMAG
SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).REAL
SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).Z0.R
SCPI.CALCulate(Ch).FSIMulator.STATe

Analysis > Fixture Simulator > Port ZConversion > Port ZConversion

SCPI

:CALCulate{[1]-36}:FSIMulator:SENDed:ZCONversion:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:SENDed:ZCONversion:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:SEND:ZCON:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:SEND:ZCON:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).FSIMulator.STATe

SCPI.CALCulate(*Ch*).FSIMulator.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).FSIMulator.STATe

/ (*Ch*)

	Status
	/
	(Boolean)
	• •

```
Dim FxtSim As Boolean
SCPI.CALCulate(1).FSIMulator.STATe = True
FxtSim = SCPI.CALCulate(1).FSIMulator.STATe
```

SCPI

:CALCulate{[1]-36}:FSIMulator:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}:FSIMulator:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FSIM:STAT ON"
20 OUTPUT 717;":CALC1:FSIM:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).PARameter(*Tr*).DEFine

SCPI.CALCulate(*Ch*).PARameter(*Tr*).DEFine = *Param*

Param = SCPI.CALCulate(*Ch*).PARameter(*Tr*).DEFine

(*Ch*)/ (*Tr*)

	Param
	(String)
	• “S<XY>” x=14 Y=14 • A|B|C|D • R<X> (X=1-4) • AUX1AUX2
	“S11”

```
Dim MeasPara As String
SCPI.CALCulate(1).PARameter(1).DEFine = "s21"
MeasPara = SCPI.CALCulate(1).PARameter(1).DEFine
```

Meas > **S<XY>** {X=1-4;Y=1-4}|

Meas > **Absolute** > **R<X>** {X=1-4}|**A**|**B**|**C**|**D**|

Meas > **Aux Input 1**|**Aux Input 2**

SCPI

:CALCulate{[1]-36}:PARameter{[1]-16}:DEFine
{S11|S21|S31|S41|S12|S22|S32|S42|S13|S23|S33|S43|S14|S24|S34|S44
1|R2|R3|R4|Aux1|AUX2}

:CALCulate{[1]-36}:PARameter{[1]|-16}:DEFine?

{S11|S21|S31|S41|S12|S22|S32|S42|S13|S23|S33|S43|S14|S24|S34|S44
<newline><^END>

```
10 OUTPUT 717;":CALC1:PAR1:DEF S21"
20 OUTPUT 717;":CALC1:PAR1:DEF?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).PARameter(*Tr*).SELect

SCPI.CALCulate(*Ch*).PARameter(*Tr*).SELect

/ (*Ch*)(*Tr*)

	Tr
	(Long)
	116
	1

SCPI.CALCulate(2).PARameter(2).SELect

SCPI.CALCulate(Ch).SELected.CORRection.EDELay.MEDium
SCPI.CALCulate(Ch).SELected.CORRection.EDELay.TIME
SCPI.CALCulate(Ch).SELected.CORRection.EDELay.WGCutoff
SCPI.DISPlay.WINDow(Ch).ACTivate
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

Trace Prev / Trace Next

SCPI

:CALCulate{[1]-36}:PARameter{[1]-16}:SELect

```
10 OUTPUT 717;":CALC1:PAR1:SEL"
```

SCPI.CALCulate(*Ch*).PARameter(*Tr*).SPORt

SCPI.CALCulate(*Ch*).PARameter(*Tr*).SPORt = *Value*

Value = SCPI.CALCulate(*Ch*).PARameter(*Tr*).SPORt

(*Ch*)(*Tr*)/AUX

	Value
	(Long)
	14
	1
	“SCPI.CALCulate(Ch).PARameter(Tr).DEFine”/AUX

```
Dim Sport As Long
SCPI.CALCulate(1).PARameter(1).DEFine = "B"
SCPI.CALCulate(1).PARameter(1).SPORt = 4
Sport = SCPI.CALCulate(1).PARameter(1).SPORt
```

SCPI.CALCulate(Ch).PARameter(Tr).DEFine

Meas > **Absolute** > **A(x)** ...**D(x)**...**R1(x)** ...**R4(x)** (x: 1 to 4)

Meas > **AUX Input 1** or **AUX Input 2** > **Sweep Port**

SCPI

:CALCulate{[1]-36}:PARameter{[1]-16}:SPORt <numeric>

:CALCulate{[1]-36}:PARameter{[1]-16}:SPORt?

{value}<newline><^END>

```
10 OUTPUT 717;":CALC1:PAR1:DEF B"
20 OUTPUT 717;":CALC1:PAR1:SPOR 4"
30 OUTPUT 717;":CALC1:PAR1:SPOR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).PARameter.COUNt

SCPI.CALCulate(*Ch*).PARameter.COUNt = *Value*

Value = SCPI.CALCulate(*Ch*).PARameter.COUNt

/ (*Ch*)

	Value
	(Long)
	/
	1

```
Dim TraceNum As Long
SCPI.CALCulate(1).PARameter.COUNt = 4
TraceNum = SCPI.CALCulate(1).PARameter.COUNt
```

Display > Num of Traces

SCPI

:CALCulate{[1]-36}:PARameter:COUNt <numeric>

:CALCulate{[1]-36}:PARameter:COUNt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:PAR:COUN 4"
20 OUTPUT 717;":CALC1:PAR:COUN?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.BLIMit.DB

```
SCPI.CALCulate(Ch).SELected.BLIMit.DB = Value
Value = SCPI.CALCulate(Ch).SELected.BLIMit.DB
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect/

	Value
	N dB
	(Double)
	05E8
	0
	dB

(*Ch*)

```
Dim BLimDB As Double
SCPI.CALCulate(1).SELected.BLIMit.DB = 3
BLimDB = SCPI.CALCulate(1).SELected.BLIMit.DB
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

Analysis > Bandwidth Limit > N dB Points

SCPI

:CALCulate{[1]-36}[:SELected]:BLIMit:DB <numeric>

:CALCulate{[1]-36}[:SELected]:BLIMit:DB?

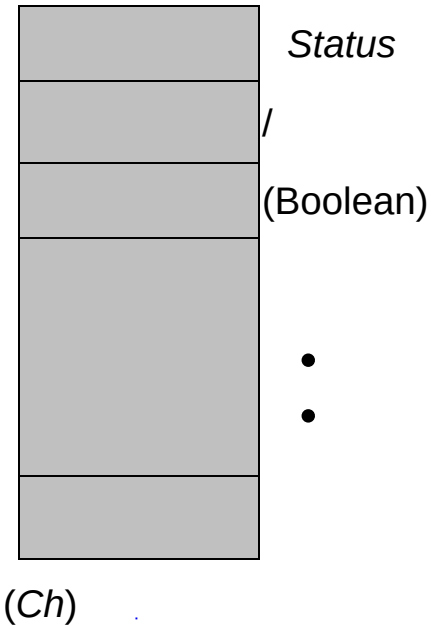
{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:BLIM:DB 3"
20 OUTPUT 717;":CALC1:BLIM:DB?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.MARKer

```
SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.MARKer = Status
Status = SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.MARKer
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect/


Status
/
(Boolean)
(*Ch*)

```
Dim BLimMk As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.BLIMit.DISPlay.MARKer = True
BLimMk = SCPI.CALCulate(1).SELected.BLIMit.DISPlay.MARKer
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.BLIMit.STATe
SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.VALue

Analysis > Bandwidth Limit > BW Marker

SCPI

:CALCulate{[1]-36}[:SELected]:BLIMit:DISPlay:MARKer {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:BLIMit:DISPlay:MARKer?

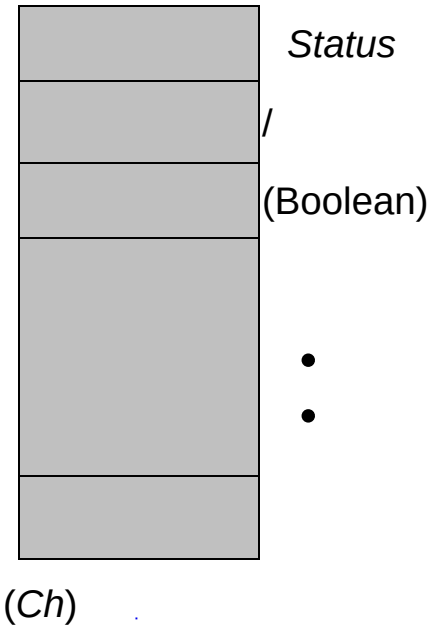
{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:BLIM:DISP:MARK ON"
20 OUTPUT 717;":CALC1:BLIM:DISP:MARK?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.VALue

```
SCPI.CALCulate(CH).SELected.BLIMit.DISPlay.VALue = Status
Status = SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.VALue
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect/


Status
/
(Boolean)
(*Ch*)

```
Dim BLimVal As Boolean
SCPI.CALCulate(1).PARameter(1).SELectSCPI.CALCulate(1).SELectec
= True
BLimVal = SCPI.CALCulate(1).SELected.BLIMit.DISPlay.VALue
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.BLIMit.STATe
SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.MARKer

Analysis > Bandwidth Limit > BW Display

SCPI

:CALCulate{[1]-36}[:SELected]:BLIMit:DISPlay:VALue {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:BLIMit:DISPlay:VALue?

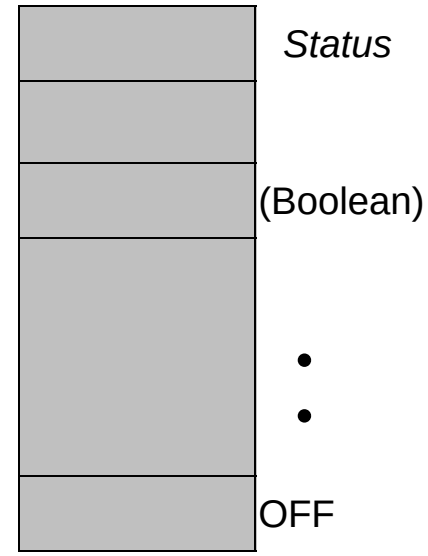
{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:BLIM:DISP:VAL ON"
20 OUTPUT 717;":CALC1:BLIM:DISP:VAL?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.BLIMit.FAIL

Status = SCPI.CALCulate(Ch).SELected.BLIMit.FAIL

SCPI.CALCulate(Ch).PARameter(Tr).SELect



(*Ch*)

```
Dim Result As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.BLIMit.STATe = True
Result = SCPI.CALCulate(1).SELected.BLIMit.FAIL
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

SCPI

:CALCulate{[1]-36}[:SELected]:BLIMit:FAIL?

{1|0}<newline><^END>

1	
0	

OFF0

```
10 OUTPUT 717;":CALC1:BLIM:FAIL?"
20 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.BLIMit.MAXimum

```
SCPI.CALCulate(Ch).SELected.BLIMit.MAXimum = Value
Value = SCPI.CALCulate(Ch).SELected.BLIMit.MAXimum
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect/

	Value
	(Double)
	01E12
	3E5
	HzdBm

(*Ch*)116136

```
Dim BLimMax As Double
SCPI.CALCulate(1).SELected.BLIMit.MAXimum = 1E9
BLimMax = SCPI.CALCulate(1).SELected.BLIMit.MAXimum
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.BLIMit.STATe
SCPI.CALCulate(Ch).SELected.BLIMit.MINimum

Analysis > Bandwidth Limit > Max Bandwidth

SCPI

:CALCulate{[1]-36}[:SELected]:BLIMit:MAXimum <numeric>

:CALCulate{[1]-36}[:SELected]:BLIMit:MAXimum?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:BLIM:MAX 3E5"
20 OUTPUT 717;":CALC1:BLIM:MAX?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.BLIMit.MINimum

```
SCPI.CALCulate(Ch).SELected.BLIMit.MINimum = Value
Value = SCPI.CALCulate(Ch).SELected.BLIMit.MINimum
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect/

	Value
	(Double)
	01E12
	1E4
	HzdBm

(*Ch*)

```
Dim BLimMin As Double
SCPI.CALCulate(1).SELected.BLIMit.MINimum = 1E6
BLimMin = SCPI.CALCulate(1).SELected.BLIMit.MINimum
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.BLIMit.STATe
SCPI.CALCulate(Ch).SELected.BLIMit.MAXimum

Analysis > Bandwidth Limit > Min Bandwidth

SCPI

:CALCulate{[1]-36}[:SELected]:BLIMit:MINimum <numeric>

:CALCulate{[1]-36}[:SELected]:BLIMit:MINimum?

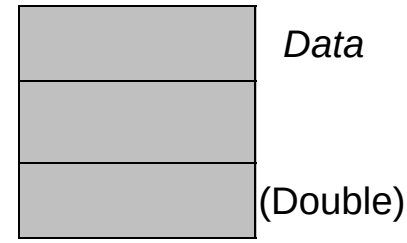
{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:BLIM:MIN 1E4"
20 OUTPUT 717;":CALC1:BLIM:MIN?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.BLIMit.REPort.DATA

```
Data = SCPI.CALCulate(Ch).SELected.BLIMit.REPort.DATA
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect



(*Ch*)

```
Dim BWData As Double
SCPI.CALCulate(1).PARameter(1).SELect
BWData = SCPI.CALCulate(1).SELected.BLIMit.REPort.DATA
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

SCPI

:CALCulate{[1]-36}[:SELected]:BLIMit:REPort[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:BLIM:REP?"
20 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

```
SCPI.CALCulate(Ch).SELected.BLIMit.STATe = Status
Status = SCPI.CALCulate(Ch).SELected.BLIMit.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect/

	Status
	/
	(Boolean)
	• •

(*Ch*) .

```
Dim BLimTest As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.BLIMit.STATe = True
BLimTest = SCPI.CALCulate(1).SELected.BLIMit.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.BLIMit.DB
SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.MARKer
SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.VALue
SCPI.CALCulate(Ch).SELected.BLIMit.FAIL
SCPI.CALCulate(Ch).SELected.BLIMit.MAXimum
SCPI.CALCulate(Ch).SELected.BLIMit.MINimum
SCPI.CALCulate(Ch).SELected.BLIMit.REPort.DATA

Analysis > Bandwidth Limit > BW Test

SCPI

:CALCulate{[1]-36}[:SELected]:BLIMit[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:BLIMit[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:BLIM ON"
20 OUTPUT 717;":CALC1:BLIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.CONVersion.FUNCtion

SCPI.CALCulate(*Ch*).SELected.CONVersion.FUNCtion = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.CONVersion.FUNCtion

/ (Ch)

	Param
	(String)
	• “ZREFlection” • “ZTRansmit” • “YREFlection” • “YTRansmit” • “INVersion”S • “ZTSHunt” • “YTSHunt” • “CONJugation”
	ZREFlection

```
Dim Func As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.CONVersion.FUNCtion = "ztr"
Func = SCPI.CALCulate(1).SELected.CONVersion.FUNCtion
```

SCPI.CALCulate(Ch).SELected.CONVersion.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

Analysis > **Conversion** >
Z:Reflection|Z:Transmission|Y:Reflection|Y:Transmission|1/S|
Z:Trans-Shunt|Y:Trans-Shunt|Conjugation

SCPI

:CALCulate{[1]-36}[:SELected]:CONVersion:FUNCtion {ZREFlection|
ZTRansmit|YREFlection|YTRansmit|INVersion|ZTSHunt|YTSHunt|CONJu

:CALCulate{[1]-36}[:SELected]:CONVersion:FUNCtion?

{ZREF|ZTR|YREF|YTR|INV|ZTSH|YTSH|CONJ}<newline><^END>

```
10 OUTPUT 717;":CALC1:CONV:FUNC ZTR"
20 OUTPUT 717;":CALC1:CONV:FUNC?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.CONVersion.STATe

SCPI.CALCulate(*Ch*).SELected.CONVersion.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.CONVersion.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim Conv As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.CONVersion.STATe = True
Conv = SCPI.CALCulate(1).SELected.CONVersion.STATe
```

SCPI.CALCulate(Ch).SELected.CONVersion.FUNCtion

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > Conversion > Conversion

SCPI

:CALCulate{[1]-36}[:SELected]:CONVersion[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:CONVersion[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:CONV ON"
20 OUTPUT 717;":CALC1:CONV?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.MEDiu

SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.MEDium = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.MEDium

(*Ch*)

	Param
	(String)
	• “COAXial” • “WAVeguide”
	“COAXial”

```
Dim EdelMed As String
SCPI.CALCulate(1).SELected.CORRection.EDELay.MEDium =
"WAVeguide"
EdelMed =
SCPI.CALCulate(1).SELected.CORRection.EDELay.MEDium
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.CORRection.EDELay.TIME
SCPI.CALCulate(Ch).SELected.CORRection.EDELay.WGCutoff

Scale > **Electrical Delay** > **Media**

SCPI

:CALCulate{[1]-36}[:SELected]:CORRection:EDELay:MEDium {COAXial|WAVeguide}

:CALCulate{[1]-36}[:SELected]:CORRection:EDELay:MEDium?

{COAXial|WAVeguide}<newline><^END>

```
10 OUTPUT 717;":CALC1:CORR:EDEL:MED WAV"
20 OUTPUT 717;":CALC1:CORR:EDEL:MED?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.TIME

SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.TIME = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.TIME

/136 *(Ch)*

	Value
	(Double)
	-1010
	0
	s

```
Dim Edel As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.CORRection.EDELay.TIME = 0.2
Edel = SCPI.CALCulate(1).SELected.CORRection.EDELay.TIME
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.CORRection.EDELay.MEDium
SCPI.CALCulate(Ch).SELected.CORRection.EDELay.WGCutoff

Scale > Electrical Delay

SCPI

:CALCulate{[1]-36}[:SELected]:CORRection:EDELay:TIME <numeric>

:CALCulate{[1]-36}[:SELected]:CORRection:EDELay:TIME?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:CORR:EDEL:TIME 0.2"
20 OUTPUT 717;":CALC1:CORR:EDEL:TIME?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.WGCut

SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.WGCutoff = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.WGCutoff

/136 *(Ch)*

	Value
	(Double)
	9E33E9230430 9E38.5E9280480 1E53E9235435 1E58.5E9285485
	9E3230430280480 1E5235435285485
	s

```
Dim EdelWgc As Double
SCPI.CALCulate(1).SELected.CORRection.EDELay.WGCutoff = 1E9
Edel = SCPI.CALCulate(1).SELected.CORRection.EDELay.WGcutoff
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.CORRection.EDELay.MEDium
SCPI.CALCulate(Ch).SELected.CORRection.EDELay.TIME

SCPI

:CALCulate{[1]-36}[:SELected]:CORRection:EDELay:WGCutoff <numeric>

:CALCulate{[1]-36}[:SELected]:CORRection:EDELay:WGCutoff?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:CORR:EDEL:WGC 1E9"
20 OUTPUT 717;":CALC1:CORR:EDEL:WGC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.CORRection.OFFSet.PHASe

SCPI.CALCulate(*Ch*).SELected.CORRection.OFFSet.PHASe = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.CORRection.OFFSet.PHASe

/ (*Ch*)

	Value
	(Double)
	-360360
	0
	°

```
Dim Offset As Double
SCPI.CALCulate(2).PARameter(1).SELect
SCPI.CALCulate(2).SELected.CORRection.OFFSet.PHASe = 2.5
Offset = SCPI.CALCulate(2).SELected.CORRection.OFFSet.PHASe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Scale > Phase Offset

SCPI

:CALCulate{[1]-36}[:SELected]:CORRection:OFFSet:PHASe <numeric>

:CALCulate{[1]-36}[:SELected]:CORRection:OFFSet:PHASe?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:CORR:OFFS:PHAS 2.5"
20 OUTPUT 717;":CALC1:CORR:OFFS:PHAS?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.DATA.FDATa

SCPI.CALCulate(*Ch*).SELected.DATA.FDATa = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.DATA.FDATa

(*Ch*)/

SCPI.CALCulate(Ch).SELected.FORMat

“1.#QNB”

	Data
	NOP*2n1NOP • *n*2-2* n • *n*2-1* nSmith0 0
	(Variant)
	NOP*2

```
Dim FmtData As Variant
SCPI.SENSe(1).SWEep.POINts = 201
SCPI.CALCulate(1).PARameter(1).SELect
FmtData = SCPI.CALCulate(1).SELected.DATA.FDATa
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.DATA.FDATa = FmtData
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.SENSe(Ch).SWEep.POINts
SCPI.CALCulate(Ch).SELected.FORMat
SCPI.CALCulate(Ch).SELected.DATA.FMEMory
SCPI.CALCulate(Ch).SELected.DATA.SDATa

SCPI

:CALCulate{[1]-36}[:SELected]:DATA:FDATa <numeric1>,?nbsp;,
<numeric NOP*2>

:CALCulate{[1]-36}[:SELected]:DATA:FDATa?

{numeric 1},?nbsp;,{numeric NOP*2}<newline><^END>

```
10 DIM A(1:201,1:2)
20 OUTPUT 717;":CALC1:DATA:FDAT?"
30 ENTER 717;A(*)
```

SCPI.CALCulate(*Ch*).SELected.DATA.FMEMory

SCPI.CALCulate(*Ch*).SELected.DATA.FMEMory = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.DATA.FMEMory

(*Ch*)/

SCPI.CALCulate(Ch).SELected.FORMat

“1.#QNB”

	Data
	NOP*2n1NOP • *n*2-2* n • *n*2-1* nSmith0 0
	(Variant)
	NOP\ 2

```
Dim FmtMem As Variant
SCPI.SENSe(1).SWEep.POINts = 201
SCPI.CALCulate(1).PARameter(1).SELect
FmtMem = SCPI.CALCulate(1).SELected.DATA.FMEMory
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.DATA.FMEMory = FmtMem
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.SENSe(Ch).SWEep.POINts
SCPI.CALCulate(Ch).SELected.FORMat
SCPI.CALCulate(Ch).SELected.DATA.FDATa
SCPI.CALCulate(Ch).SELected.DATA.SMEMory

SCPI

:CALCulate{[1]-36}[:SELected]:DATA:FMEMory <numeric 1>,?nbsp;,
<numeric NOP*2>

:CALCulate{[1]-36}[:SELected]:DATA:FMEMory?

{numeric 1},?nbsp;,{numeric NOP*2}<newline><^END>

```
10 DIM A(1:201,1:2)
20 OUTPUT 717;":CALC1:DATA:FMEM?"
30 ENTER 717;A(*)
```

SCPI.CALCulate(*Ch*).SELected.DATA.SDATa

SCPI.CALCulate(*Ch*).SELected.DATA.SDATa = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.DATA.SDATa

(Ch)/

“1.#QNB”

	Data
	NOP*2n1NOP • *n*2-2* n • *n*2-1* n 0
	(Variant)
	NOP*2

```
Dim CorData As Variant
SCPI.SENSe(1).SWEep.POINts = 201
CorData = SCPI.CALCulate(1).SELected.DATA.SDATa
SCPI.SENSe(2).SWEep.POINts = 201
SCPI.CALCulate(2).SELected.DATA.SDATa = CorData
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.SENSe(Ch).SWEep.POINts
SCPI.CALCulate(Ch).SELected.DATA.SMEMory
SCPI.CALCulate(Ch).SELected.DATA.FDATa

SCPI

:CALCulate{[1]-36}[:SELected]:DATA:SDATa <numeric 1>,?nbsp;,
<numeric NOP*2>

:CALCulate{[1]-36}[:SELected]:DATA:SDATa?

<numeric 1>,?nbsp;,<numeric NOP*2><^END>

```
10 DIM A(1:201,1:2)
20 OUTPUT 717;":CALC1:DATA:SDAT?"
30 ENTER 717;A(*)
```

SCPI.CALCulate(*Ch*).SELected.DATA.SMEMory

SCPI.CALCulate(*Ch*).SELected.DATA.SMEMory = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.DATA.SMEMory

(Ch)/

“1.#QNB”

	Data
	NOPx2n1NOP • *nx2-2* n • *nx2-1* n 0
	(Variant)
	NOP\ 2

```
Dim CorMem As Variant
SCPI.SENSe(1).SWEep.POINts = 201
CorMem = SCPI.CALCulate(1).SELected.DATA.SMEMory
SCPI.SENSe(2).SWEep.POINts = 201
SCPI.CALCulate(1).SELected.DATA.SMEMory = CorMem
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.SENSe(Ch).SWEep.POINts
SCPI.CALCulate(Ch).SELected.DATA.SDATa
SCPI.CALCulate(Ch).SELected.DATA.FMEMory

SCPI

:CALCulate{[1]-36}[:SELected]:DATA:SMEMory <numeric 1>,?nbsp;,
<numeric NOP*2>

:CALCulate{[1]-36}[:SELected]:DATA:SMEMory?

<numeric 1>,?nbsp;,<numeric NOP*2><^END>

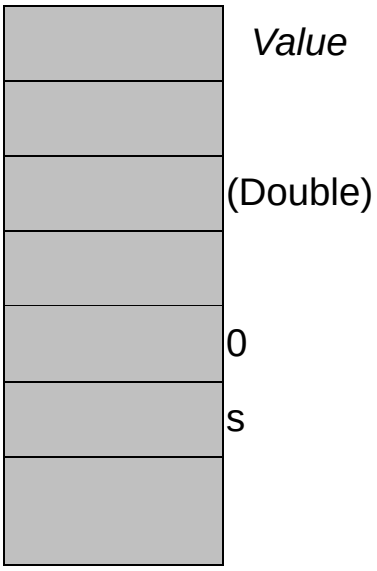
```
10 DIM A(1:201,1:2)
20 OUTPUT 717;":CALC1:DATA:SMEM?"
30 ENTER 717;A(*)
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.CENTer

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.CENTer = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.CENTer

(Ch)/


Value
(Double)
0
s

```
Dim FilCent As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.CENTer = 1E-8
FilCent = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.CENTer
```

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.SPAN
SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Gating** > **Center**

SCPI

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:CENTer <numeric>
:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:CENTer?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FILT:TIME:CENT 1E-8"
20 OUTPUT 717;":CALC1:FILT:TIME:CENT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME. SHAPe

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SHAPe = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SHAPe

(Ch)/

	Param
	(String)
	• “MAXimum” • “WIDE” • “NORMal” • “MINimum”
	“NORMal”

```
Dim FilShape As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SHAPe = "wide"
FilShape = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SHAPe
```

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.TYPE
SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Gating** > **Shape** > **Maximum|Wide|Normal|Minimum**

SCPI

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:SHAPe {MAXimum|WIDE|NORMal|MINimum}

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:SHAPe?

{MAX|WIDE|NORM|MIN}<newline><^END>

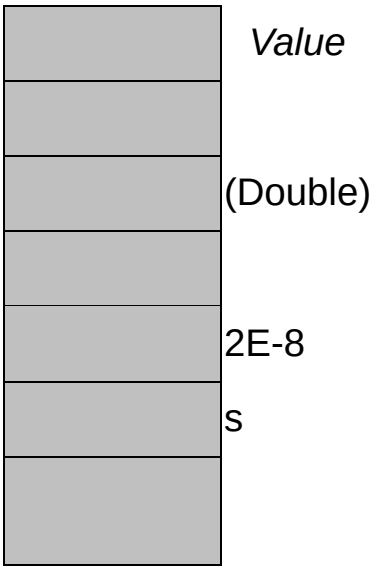
```
10 OUTPUT 717;":CALC1:FILT:TIME:SHAP WIDE"
20 OUTPUT 717;":CALC1:FILT:TIME:SHAP?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME. SPAN

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SPAN = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SPAN

(Ch)/


Value
(Double)
2E-8
s

```
Dim FilStar As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SPAN = 1E-8
FilStar = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SPAN
```

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.CENTer
SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > Gating > Span

SCPI

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:SPAN <numeric>

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:SPAN?

{numeric}<newline><^END>

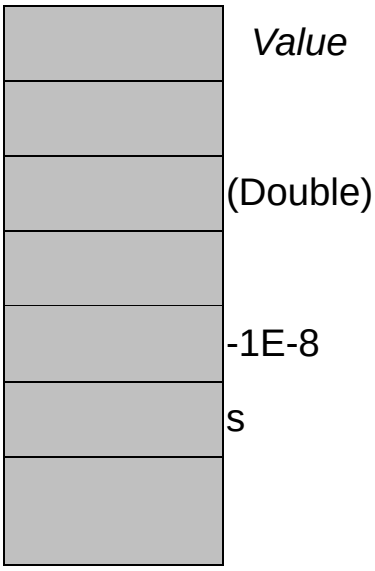
```
10 OUTPUT 717;":CALC1:FILT:TIME:SPAN 1E-8"
20 OUTPUT 717;":CALC1:FILT:TIME:SPAN?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME. STARt

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STARt

(Ch)/


Value
(Double)
-1E-8
s

```
Dim FilCent As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STARt = 0
FilCent = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STARt
```

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STOP
SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Gating** > **Start**

SCPI

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:STARt <numeric>

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:STARt?

{numeric}<newline><^END>

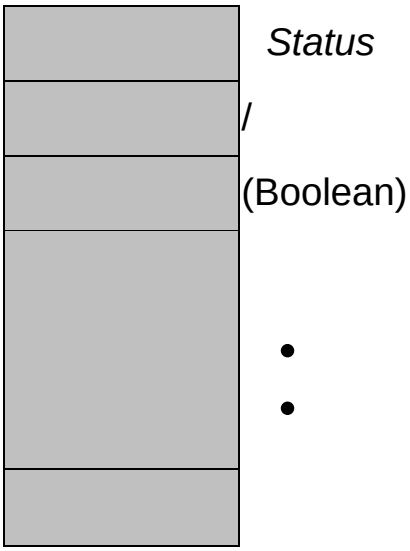
```
10 OUTPUT 717;":CALC1:FILT:TIME:STAR 0"
20 OUTPUT 717;":CALC1:FILT:TIME:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME. STATe

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STATe

(*Ch*)/
33


Status
/
(Boolean)

```
Dim Gating As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STATe = True
Gating = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.SENSe(Ch).SWEep.TYPE
SCPI.SENSe(Ch).SWEep.POINts

Analysis > Gating > Gating

SCPI

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:STATe?

{1|0}<newline><^END>

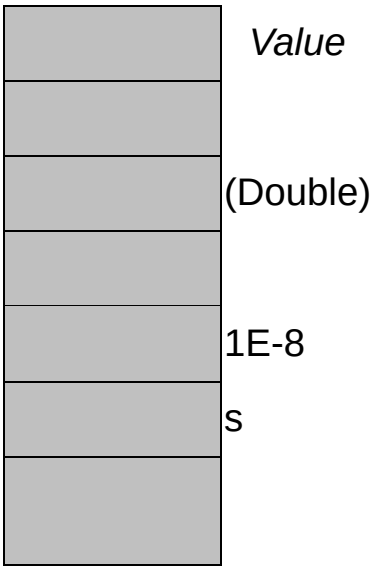
```
10 OUTPUT 717;":CALC1:FILT:TIME:STAT ON"
20 OUTPUT 717;":CALC1:FILT:TIME:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME. STOP

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STOP

(Ch)/


Value
(Double)
1E-8
s

```
Dim FilStop As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STOP = 2E-8
FilStop = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STOP
```

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STARt
SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Gating** > **Stop**

SCPI

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:STOP <numeric>

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME:STOP?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FILT:TIME:STOP 2E-8"
20 OUTPUT 717;":CALC1:FILT:TIME:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME. TYPE

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.TYPE

(Ch)/

	Param
	(String)
	• “BPASs” • “NOTCh”"
	“BPASs”

```
Dim FilType As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SHAPe = "notc"
FilType = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SHAPe
```

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.SHAPe
SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > Gating > Type

SCPI

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME[:TYPE] {BPASs|NOTCh}

:CALCulate{[1]-36}[:SELected]:FILTer[:GATE]:TIME[:TYPE]?

{BPAS|NOTC}<newline><^END>

```
10 OUTPUT 717;":CALC1:FILT:TIME NOTC"
20 OUTPUT 717;":CALC1:FILT:TIME?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FORMat

SCPI.CALCulate(*Ch*).SELected.FORMat = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FORMat

/ (*Ch*)

	Param
	(String)
	• “MLOGarithmic” • “PHASe” • “GDELay” • “SLINear”Smith/ • “SLOGarithmic”Smith/ • “SCOMplex”Smith/ • “SMITh”SmithR+jX • “SADMittance”SmithG+jB • “PLINear”Lin/ • “PLOGarithmic”/ • “POLar”/ • “MLINear” • “SWR”SWR • “REAL” • “IMAGinary” • “UPHase” • “PPHase”
	“MLOGarithmic”

```
Dim Fmt As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FORMat = "smit"
Fmt = SCPI.CALCulate(1).SELected.FORMat
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Format > **Log Mag|Phase|Group Delay|Lin Mag|SWR|Real|Imaginary|Expand Phase| Positive Phase**

Format > **Smith** > **Lin/Phase|Log/Phase|Real/Imag|R+jX|G+jB**

Format > **Polor** > **Lin/Phase|Log/Phase|Real/Imag**

SCPI

:CALCulate{[1]-36}[:SELected]:FORMat {MLOGarithmic|PHASe|GDELay|
SLINear|SLOGarithmic|SCOMplex|SMITh|SADMittance|PLINear|PLOGar
IMAGinary|UPHase|PPHase}

:CALCulate{[1]-36}[:SELected]:FORMat?

{MLOG|PHAS|GDEL|SLIN|SLOG|SCOM|SMIT|SADM|PLIN|PLOG|POL|M
REAL|IMAG|UPH|PPH}<newline><^END>

```
10 OUTPUT 717;":CALC1:FORM SLIN"
20 OUTPUT 717;":CALC1:FORM?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DATA

Data = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DATA

(*Ch*)SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

	Data
	Nx2 SCPI.CALCulate(Ch).SELected.FUNCtion.POINtsN n1N • *n*2-2* n • *n*2-1* n 0 0
	(Variant)

```
Dim AnaData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "mean"
SCPI.CALCulate(1).SELected.FUNCtion.EXECute
AnaData = SCPI.CALCulate(1).SELected.FUNCtion.DATA
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute
SCPI.CALCulate(Ch).SELected.FUNCtion.POINts

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:DATA?

<numeric 1>,?nbsp;,<numeric N*2><^END>

```
10 OUTPUT 717;":CALC1:FUNC:POIN?"
20 ENTER 717;A
30 REDIM B(1:2*A)
40 OUTPUT 717;":CALC1:FUNC:DATA?"
50 ENTER 717;B(*)
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.COUPle

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.COUPle = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.COUPle

(*Ch*)SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

	Status
	/
	(Boolean)
	• •

```
Dim TrCpl As Boolean
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.COUPle = False
TrCpl = SCPI.CALCulate(1).SELected.FUNCtion.DOMain.COUPle
```

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:DOMain:COUPle {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:FUNCtion:DOMain:COUPle?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FUNC:DOM:COUP OFF"
20 OUTPUT 717;":CALC1:FUNC:DOM:COUP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STARt

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STARt

(*Ch*)/ SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

	Value
	(Double)
	0
	HzdBms

```
Dim AnaStar As Double
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt = 1.5E9
AnaStar = SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt
```

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STOP
SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STATe
SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.COUPle
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:DOMain:STARt <numeric>

:CALCulate{[1]-36}[:SELected]:FUNCtion:DOMain:STARt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FUNC:DOM:STAR 1.7E9"
20 OUTPUT 717;":CALC1:FUNC:DOM:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STATe

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STATe

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute (*Ch*)/

	Status
	(Boolean)
	• •

```
Dim AnaRnge As Boolean
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt = 1.5E9
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP = 1.8E9
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STATe = True
AnaRnge = SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STATe
```

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STARt
SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STOP
SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.COUPle
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:DOMain[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:FUNCtion:DOMain[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:FUNC:DOM ON"
20 OUTPUT 717;":CALC1:FUNC:DOM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STOP

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STOP

(*Ch*)/ SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

	Value
	(Double)
	0
	HzdBm s

```
Dim AnaStop As Double
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP = 1.8E9
AnaStop = SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP
```

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STARt
SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STATe
SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.COUPle
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:DOMain:STOP <numeric>

:CALCulate{[1]-36}[:SELected]:FUNCtion:DOMain:STOP?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FUNC:DOM:STOP 1.8E9"
20 OUTPUT 717;":CALC1:FUNC:DOM:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.EXECute

SCPI.CALCulate(*Ch*).SELected.FUNCtion.EXECute

(*Ch*)SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.EXECute

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STATe

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:EXECute

```
10 OUTPUT 717;":CALC1:FUNC:EXEC"
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.PEXCursion

SCPI.CALCulate(*Ch*).SELected.FUNCtion.PEXCursion = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.PEXCursion

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute (*Ch*)/

	Value
	(Double)
	05E8
	3
	• MLOGdB • PHASUPHPPH° • GDELs •

```
Dim PeakExc As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "peak"
SCPI.CALCulate(1).SELected.FUNCtion.PEXCursion = 1.5
PeakExc = SCPI.CALCulate(1).SELected.FUNCtion.PEXCursion
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.FUNCtion.PPOLarity
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:PEXCursion <numeric>

:CALCulate{[1]-36}[:SELected]:FUNCtion:PEXCursion?

{numeric}<newline><^END>

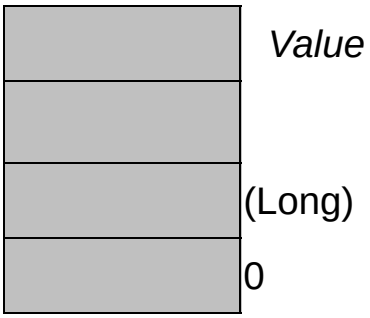
```
10 OUTPUT 717;":CALC1:FUNC:PEXC 0.2"
20 OUTPUT 717;":CALC1:FUNC:PEXC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.POINts

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.POINts

(*Ch*)SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

1


Value
(Long)
0

```
Dim AnaPoin As Long
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "ape"
SCPI.CALCulate(1).SELected.FUNCtion.EXECute
AnaPoin = SCPI.CALCulate(1).SELected.FUNCtion.POINts
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute
SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:POINts?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FUNC:POIN?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.PPOLarity

SCPI.CALCulate(*Ch*).SELected.FUNCtion.PPOLarity = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FUNCtion.PPOLarity

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute (*Ch*)/

	Param
	(String)
	• “POSitive” • “NEGative” • “BOTH”
	“POSitive”

```
Dim PeakPol As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "peak"
SCPI.CALCulate(1).SELected.FUNCtion.PPOLarity = "both"
PeakPol = SCPI.CALCulate(1).SELected.FUNCtion.PPOLarity
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.FUNCtion.PEXCursion
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:PPOLarity {POSitive|NEGative|BOTH}

:CALCulate{[1]-36}[:SELected]:FUNCtion:PPOLarity?

{POS|NEG|BOTH}<newline><^END>

```
10 OUTPUT 717;":CALC1:FUNC:PPOL BOTH"
20 OUTPUT 717;":CALC1:FUNC:PPOL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TARGet

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TARGet = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.TARGet

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute (*Ch*)/

Value
(Double)
-5E85E8
0
• MLOGdB • PHASUPHPPH° • GDELs •

```
Dim TargVal As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "atar"
SCPI.CALCulate(1).SELected.FUNCtion.TARGet = -12.5
TargVal = SCPI.CALCulate(1).SELected.FUNCtion.TARGet
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.FUNCtion.TTRansition
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:TARGet <numeric>

:CALCulate{[1]-36}[:SELected]:FUNCtion:TARGet?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:FUNC:TARG -12.5"
20 OUTPUT 717;":CALC1:FUNC:TARG?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TTRansition

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TTRansition = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FUNCtion.TTRansition

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute (*Ch*)/

	Param
	(String)
	• “POSitive” • “NEGative” • “BOTH”
	“BOTH”

```
Dim TargTran As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "atar"
SCPI.CALCulate(1).SELected.FUNCtion.TTRansition = "pos"
TargTran = SCPI.CALCulate(1).SELected.FUNCtion.TTRansition
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.FUNCtion.TARGet
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:TTRansition {POSitive|NEGative|BOTH}

:CALCulate{[1]-36}[:SELected]:FUNCtion:TTRansition?

{POS|NEG|BOTH}<newline><^END>

```
10 OUTPUT 717;":CALC1:FUNC:TTR NEG"
20 OUTPUT 717;":CALC1:FUNC:TTR?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TYPE

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FUNCtion.TYPE

(Ch)/

	Param
	(String)
	• “PTPeak” • “STDEV” • “MEAN” • “MAXimum” • “MINimum” • “PEAK” • “APEak” • “ATARget”
	“PTPeak”

```
Dim AnaType As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "atar"
AnaType = SCPI.CALCulate(1).SELected.FUNCtion.TYPE
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.FUNCtion.PEXCursion
SCPI.CALCulate(Ch).SELected.FUNCtion.PPOLarity
SCPI.CALCulate(Ch).SELected.FUNCtion.TARGet
SCPI.CALCulate(Ch).SELected.FUNCtion.TTRansition
SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:FUNCtion:TYPE {PTPeak| STDEV|MEAN| MAXimum|MINimum|PEAK|APEak|ATARget}

:CALCulate{[1]-36}[:SELected]:FUNCtion:TYPE?

{PTP|STDEV|MEAN|MAX|MIN|PEAK|APE|ATAR}<newline><^END>

```
10 OUTPUT 717;":CALC1:FUNC:TYPE PEAK"
20 OUTPUT 717;":CALC1:FUNC:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.LIMit.DATA

SCPI.CALCulate(*Ch*).SELected.LIMit.DATA = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.LIMit.DATA

(Ch)/

	Data
	1 + Num*5n1Num • 0010000 • *n*5-4* n 02 0: 1: 2: • *n*5-3* n// • *n*5-2* n// • *n*5-1* n • *n*5* n 0
	(Variant)
	1+Num*5 *n*5-4* 012 *n*5-3* *n*5-2* *n*5-1* *n85*

```
Dim LimData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.DATA = Array(1,1,1e6,1e9,0,0)
LimData = SCPI.CALCulate(1).SELected.LIMit.DATA
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.DATA = Array(0) 'Clear Limit Table
```

```
Dim LimData(5) As Variant
Dim Ref As Variant
LimData(0) = 1
LimData(1) = 1
LimData(2) = 1e6
LimData(3) = 1e9
LimData(4) = 0
LimData(5) = 0
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.DATA = LimData
Ref = SCPI.CALCulate(1).SELected.LIMit.DATA
```

```
Dim LimData(0) As Variant
LimData(0) = 0
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.DATA = LimData 'Clear Limit Table
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.LIMit.STATe
SCPI.CALCulate(Ch).SELected.LIMit.DISPlay.STATe

Analysis > Limit Test > Edit Limit Line

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit:DATA <numeric 1>, ... ,<numeric 1+(N*5)>

:CALCulate{[1]-36}[:SELected]:LIMit:DATA?

{numeric 1}, ... ,{numeric 1+(N*5)}<newline><^END>

```
10 DIM B(1:2,1:5)
20 OUTPUT 717;":CALC1:LIM:DATA 2,1,1E9,3E9,0,0,2,1E9,3E9,-3,-3"
30 OUTPUT 717;":CALC1:LIM:DATA?"
40 ENTER 717;A,B(*)
```

```
10 OUTPUT 717;":CALC1:LIM:DATA 0" ! Clear Limit Table
```

SCPI.CALCulate(*Ch*).SELected.LIMit.DISPlay.STATe

SCPI.CALCulate(*Ch*).SELected.LIMit.DISPlay.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.LIMit.DISPlay.STATe

(Ch)/

	Status
	(Boolean)
	• •

```
Dim LimDisp As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit = True
LimDisp = SCPI.CALCulate(1).SELected.LIMit.DISPlay.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.STATe

Analysis > Limit Test > Limit Line

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit:DISPlay[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:LIMit:DISPlay[:STATe]?

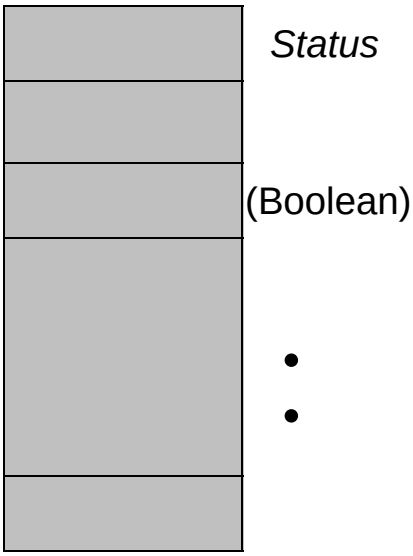
{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:LIM:DISP ON"
20 OUTPUT 717;":CALC1:LIM:DISP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.LIMit.FAIL

Status = SCPI.CALCulate(*Ch*).SELected.LIMit.FAIL

(*Ch*)


Status
(Boolean)

```
Dim Result As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.STATe = True
Result = SCPI.CALCulate(1).SELected.LIMit.FAIL
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.STATe

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit:FAIL?

{1|0}<newline><^END>

1	
0	

0

```
10 OUTPUT 717;":CALC1:LIM:FAIL?"
20 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude

```
SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude = Value
Value = SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude
```

/ SCPI.CALCulate(Ch).PARameter(Tr).SELect

	Value
	(Double)
	-5E85E8
	0
	dB

(*Ch*)

```
Dim LimOffset As Double
SCPI.CALCulate(1).SELected.LIMit.OFFSet.AMPLitude = -10
LimOffset = SCPI.CALCulate(1).SELected.LIMit.OFFSet.AMPLitude
```

SCPI.CALCulate(Ch).SELected.LIMit.STATe
SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.MARKer
SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.STIMulus

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit:OFFSet:AMPLitude <numeric>

:CALCulate{[1]-36}[:SELected]:LIMit:OFFSet:AMPLitude?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:LIM:OFFS:AMPL -10"
20 OUTPUT 717;":CALC1:LIM:OFFS:AMPL?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.MARKer

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.MARKer

SCPI.CALCulate(Ch).PARameter(Tr).SELect

(*Ch*)

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.OFFSet.MARKer

SCPI.CALCulate(Ch).SELected.LIMit.STATe
SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude
SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.STIMulus

Analysis > Limit Test > Limit Line Offsets > Marker -> Amplitude Offset

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit:OFFSet:MARKer

```
10 OUTPUT 717;":CALC1:LIM:OFFS:MARK"
```

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.STIMulus

```
SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.STIMulus = Value
Value = SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.STIMulus
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect/

	Value
	(Double)
	-1E121E12
	0
	HzdBm

(*Ch*)

```
Dim LimOffset As Double
SCPI.CALCulate(1).SELected.LIMit.OFFSet.STIMulus = 1E9
LimOffset = SCPI.CALCulate(1).SELected.LIMit.OFFSet.STIMulus
```

SCPI.CALCulate(Ch).SELected.LIMit.STATe
SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude
SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.MARKer

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit:OFFSet:STIMulus <numeric>

:CALCulate{[1]-36}[:SELected]:LIMit:OFFSet:STIMulus?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:LIM:OFFS:STIM 5E3"
20 OUTPUT 717;":CALC1:LIM:OFFS:STIM?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.LIMit.REPort.ALL

```
Data = SCPI.CALCulate(Ch).SELected.LIMit.REPort.ALL
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

	Data
	NOPx4n1NOP • *nx4-3* • *nx4-2* -11 -1: 0: 1: • *nx4-1* 0 • *nx4* 0 0
	(Variant)

(*Ch*)

```
Dim LimData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
LimData = SCPI.CALCulate(1).SELected.LIMit.REPort.ALL
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.LIMit.STATe
SCPI.CALCulate(Ch).SELected.LIMit.REPort.DATA
SCPI.CALCulate(Ch).SELected.LIMit.REPort.POINts

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit:REPort:ALL?

```
10 OUTPUT 717;":SENS1:SWE:POIN?"
20 ENTER 717;A
30 REDIM B(1:4*A)
40 OUTPUT 717;":CALC1:LIM:REP:ALL?"
50 ENTER 717;B(*)
```

SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.DATA

Data = SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.DATA

(*Ch*)

Data

SCPI.CALCulate(Ch).SELected.LIMit.REPort.POINts

(Variant)

```
Dim FailData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.STATe = True
FailData = SCPI.CALCulate(1).SELected.LIMit.REPort.DATA
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.LIMit.REPort.POINts
SCPI.CALCulate(Ch).SELected.LIMit.STATe

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit:REPort[:DATA]?

{numeric 1},?/span> ,{numeric N}<newline><^END>

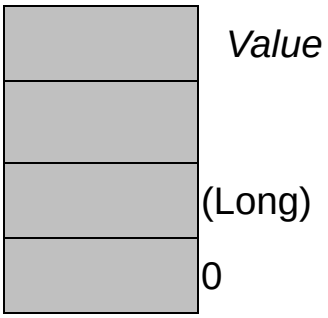
Where N is the number of the measurement points that failed (can be read out with the :CALC{1-36}:LIM:REP:POIN? command).

```
10 OUTPUT 717;":CALC1:LIM:REP:POIN?"
20 ENTER 717;A
30 REDIM B(1:A)
40 OUTPUT 717;":CALC1:LIM:REP?"
50 ENTER 717;B(*)
```

SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.POINts

Value = SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.POINts

(*Ch*)


Value
(Long)
0

```
Dim FailPoin As Long
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.STATe = True
FailPoin = SCPI.CALCulate(1).SELected.LIMit.REPort.POINts
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.STATe

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit:REPort:POINts?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:LIM:REP:POIN?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.LIMit.STATe

SCPI.CALCulate(*Ch*).SELected.LIMit.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.LIMit.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim LimTest As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.STATe = True
LimTest = SCPI.CALCulate(1).SELected.LIMit.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.LIMit.DISPlay.STATe
SCPI.DISPlay.FSIGn

Analysis > Limit Test > Limit Test

SCPI

:CALCulate{[1]-36}[:SELected]:LIMit[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:LIMit[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:LIM ON"
20 OUTPUT 717;":CALC1:LIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).ACTivate

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).ACTivate

(*Ch*)19(*Mk*)(*Mk*:10)

ON

	Mk
	(Long)
	110 10

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.DISPlay.WINDow(Ch).ACTivate

Marker > **Marker 1|Marker 2|Marker 3|Marker 4|Ref Marker**

Marker > **More Markers** > **Marker 5|Marker 6|Marker 7|Marker 8|Marker 9**

SCPI

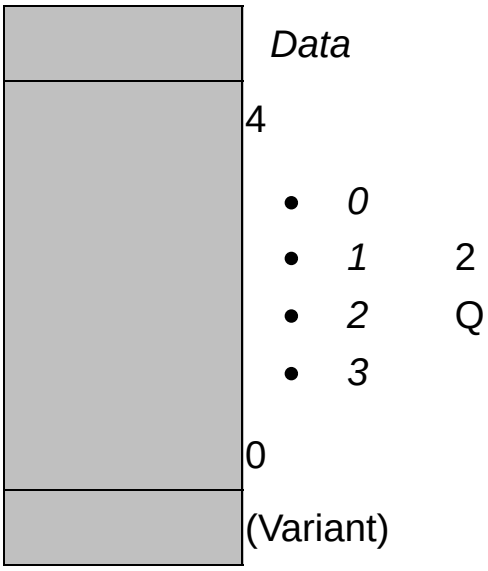
:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:ACTivate

```
10 OUTPUT 717;":CALC1:MARK1:ACT"
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.DATA

Data = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.DATA

(*Ch*)19(*Mk*)(*Mk*:10)


Data
4
0
1 2
2 Q
3
0
(Variant)

```
Dim BandData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
BandData = SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.DATA
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer.BWIDth.STATe
SCPI.CALCulate(Ch).SELected.MARKer(Mk).BWIDth.THReshold

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:BWIDth:DATA?

{numeric1},{numeric2},{numeric3},{numeric4},<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:BWID:DATA?"
20 ENTER 717;A,B,C,D
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.THResl

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.THReshold = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.THReshold

(*Ch*)/19 (*Mk*)(*Mk*:10)

	Value
	(Double)
	-5E85E8
	-3
	• MLOGdB • PHASUPHPPH° • GDELs •

```
Dim BandVal As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.THReshold = -6
BandVal =
SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.THReshold
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer.BWIDth.STATe

Marker Search > Bandwidth Value

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:BWIDth: THReshold <numeric>

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:BWIDth: THReshold?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:BWID:THR 6"
20 OUTPUT 717;":CALC1:MARK1:BWID:THR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).DISCrete

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).DISCrete = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).DISCrete

(*Mk*)/19 (*Ch*)(*Mk*:10)

	Status
	/
	(Boolean)
	• •

```
Dim MkrDsc As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).DISCrete = True
MkrDsc = SCPI.CALCulate(1).SELected.MARKer(1).DISCrete
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Marker Fctn > Discrete

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:DISCrete {ON|OFF|1|0}
:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:DISCrete?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:DISC OFF"
20 OUTPUT 717;":CALC1:MARK1:DISC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.EXEC

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.EXECute

(*Ch*)19(*Mk*)(*Mk*:10)

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion. TYPE

```
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE =
"maximum"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.EXECute
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STATe

Marker Search > **Max|Min**

Marker Search > **Peak** > **Search Peak|Search Left|Search Right**

Marker Search > **Target** > **Search Target|Search Left|Search Right**

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion:EXECute

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:EXEC"
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PEXC

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PEXCursion = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PEXCursion

19(*Mk*)(*Mk*:10) (*Ch*)/

	Value
	(Double)
	05E8
	3
	• MLOGdB • PHASUPHPPH° • GDELs •

```
Dim PeakExc As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "peak"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.PEXCursion = 0.2
PeakExc =
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.PEXCursion
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PPOLarity

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: PEXCursion <numeric>

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: PEXCursion?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:PEXC 0.2"
20 OUTPUT 717;":CALC1:MARK1:FUNC:PEXC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PPOI

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PPOLarity = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PPOLarity

(*Ch*)/19 (*Mk*)(*Mk*:10)

	Param
	(String)
	• “POSitive” • “NEGative” • “BOTH”
	“POSitive”

```
Dim PeakPol As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "peak"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.PPOLarity = "both"
PeakPol =
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.PPOLarity
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PEXCursion

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: PPOLarity {POSitive|NEGative|BOTH}

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: PPOLarity?

{POS|NEG|BOTH}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:PPOL NEG"
20 OUTPUT 717;":CALC1:MARK1:FUNC:PPOL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TARG

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TARGet = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TARGet

Description

(*Ch*)/19 (*Mk*)(*Mk*:10)

	Value
	(Double)
	-5E85E8
	0
	• MLOGdB • PHASUPHPPH° • GDELs •

```
Dim TargVal As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TARGet = -12.5
TargVal = SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TARGet
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TTRansition

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: TARGet <numeric>

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: TARGet?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:TARG -12.5"
20 OUTPUT 717;":CALC1:MARK1:FUNC:TARG?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TRAC

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TRACking = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TRACking

(*Ch*)/19 (*Mk*)(*Mk*:10)

	Status
	/
	(Boolean)
	• •

```
Dim SrchTrac As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "targ"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TRACking = True
SrchTrac =
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TRACking
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: TRACking {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: TRACking?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:TRAC ON"
20 OUTPUT 717;":CALC1:MARK1:FUNC:TRAC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TTRa

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TTRansition = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TTRansition

(*Ch*)19(*Mk*)(*Mk*:10).

	Param
	(String)
	• “POSitive” • “NEGative” • “BOTH”
	“BOTH”

```
Dim TargTran As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "targ"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TTRansition =
"neg"
TargTran =
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TTRansition
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TARGet

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: TTRansition {POSitive| NEGative|BOTH}

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion: TTRansition?

{POS|NEG|BOTH}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:TTR NEG"
20 OUTPUT 717;":CALC1:MARK1:FUNC:TTR?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TYPE

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TYPE

(*Ch*)19(*Mk*)(*Mk*:10)

	Param
	(String)
	• “MAXimum” • “MINimum” • “PEAK” • “LPEak” • “RPEak” • “TARGet” • “LTARget” • “RTARget”
	“MAXimum”

```
Dim SrchType As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "targ"
SrchType = SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PEXCursion
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PPOLarity
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TARGet
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TTRansition
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

Marker Search > **Max|Min**

Marker Search > **Peak** > **Search Peak|Search Left|Search Right**

Marker Search > **Target** > **Search Target|Search Left|Search Right**

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion
:TYPE {MAXimum|
MINimum|PEAK|LPEak|RPEak|TARGet|LTARget|RTARget}

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:FUNCtion
:TYPE?

{MAX|MIN|PEAK|LPE|RPE|TARG|LTAR|RTAR}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:TYPE PEAK"
20 OUTPUT 717;":CALC1:MARK1:FUNC:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).SET

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).SET = *Param*

(*Ch*)19(*Mk*)(*Mk*:10)(*Param*)

	Param
	(String)
	• “STARt” • “STOP” • “CENTer” • “RLEVel” • “DELay”20%

```
Dim MkrTo As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).SET = "cent"
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer.REFerence.STATe

Marker Fctn > **Marker -> Start|Marker -> Stop|Marker -> Center|Marker -> Reference|Marker -> Delay**

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:SET {STARt|STOP|CENTer|RLEVel|DELay}

```
10 OUTPUT 717;":CALC1:MARK1:SET CENT"
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).STATe

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).STATe

(*Ch*)/19 (*Mk*)Mk10

	Status
	/19
	(Boolean)
	• •

```
Dim Mkr As Boolean
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.MARKer(10).STATe = True
Mkr = SCPI.CALCulate(1).SELected.MARKer(10).STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Marker > **Marker 1|Marker 2|Marker 3|Marker 4|Ref Marker**

Marker > **More Markers** > **Marker 5|Marker 6|Marker 7|Marker 8|Marker 9**

ON

Marker > **Clear Marker Menu** > **Marker 1|Marker 2|Marker 3|Marker 4|Marker 5|**
Marker 6|Marker 7|Marker 8|Marker 9|Ref Marker

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1 ON"
20 OUTPUT 717;":CALC1:MARK1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).X

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).X = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).X

(*Ch*)19(*Mk*)(*Ch*:10)

	Value
	(Double)
	HzdBms

```
Dim MkrX As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).X = 1E9
MkrX = SCPI.CALCulate(1).SELected.MARKer(1).X
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer.REFerence.STATe
SCPI.CALCulate(Ch).SELected.MARKer(Mk).Y

Marker > **Marker 1|Marker 2|Marker 3|Marker 4|Ref Marker**

Marker > **More Markers** > **Marker 5|Marker 6|Marker 7|Marker 8|Marker 9**

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:X <numeric>

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:X?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK1:X 1E9"
20 OUTPUT 717;":CALC1:MARK1:X?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).Y

Data = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).Y

(*Ch*)19(*Mk*)(*Mk*:10)

ON SCPI.CALCulate(Ch).SELected.MARKer.REFerence.STATe“”

	Data
	2 • *0* • *1* Smith0 0
	(Variant)

```
Dim MkrY As Variant
SCPI.CALCulate(1).PARameter(1).SELect
MkrY = SCPI.CALCulate(1).SELected.MARKer(1).Y
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer.REFerence.STATe
SCPI.CALCulate(Ch).SELected.MARKer(Mk).X

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer{[1]-10}:Y?

{numeric 1},{numeric 2}<newline><^END>

{numeric 1}	
{numeric 2}	Smith0

```
10 OUTPUT 717;":CALC1:MARK1:Y?"
30 ENTER 717;A,B
```

SCPI.CALCulate(*Ch*).SELected.MARKer.BWIDth.STATe

SCPI.CALCulate(*Ch*).SELected.MARKer.BWIDth.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.BWIDth.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim BandSrch As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer.BWIDth.STATe = True
BandSrch = SCPI.CALCulate(1).SELected.MARKer.BWIDth.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.MARKer(Mk).BWIDth.DATA
SCPI.CALCulate(Ch).SELected.MARKer(Mk).BWIDth.THReshold

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer:BWIDth[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:MARKer:BWIDth[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK:BWID ON"
20 OUTPUT 717;":CALC1:MARK:BWID?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.COUPle

SCPI.CALCulate(*Ch*).SELected.MARKer.COUPle = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.COUPle

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim MkrCpl As Boolean
SCPI.CALCulate(1).SELected.MARKer.COUPle = False
MkrCpl = SCPI.CALCulate(1).SELected.MARKer.COUPle
```

Marker Fctn > Couple

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer:COUPle {ON|OFF|1|0}
:CALCulate{[1]-36}[:SELected]:MARKer:COUPle?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK:COUP OFF"
20 OUTPUT 717;":CALC1:MARK:COUP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.C

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.COUPle = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.COUPle

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim TrCpl As Boolean
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.COUPle =
False
TrCpl =
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.COUPle
```

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer:FUNCtion:DOMain:COUPle {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:MARKer:FUNCtion:DOMain:COUPle?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK:FUNC:DOM:COUP OFF"
20 OUTPUT 717;":CALC1:MARK:FUNC:DOM:COUP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.S

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STARt

(Ch)/

	Value
	(Double)
	0
	HzdBm s

```
Dim SchStar As Double
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STARt =
1.7E9
SchStar =
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STARt
```

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STOP
SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STATe
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer:FUNCtion:DOMain:STARt <numeric>

:CALCulate{[1]-36}[:SELected]:MARKer:FUNCtion:DOMain:STARt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK:FUNC:DOM:STAR 1.7E9"
20 OUTPUT 717;":CALC1:MARK:FUNC:DOM:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.S

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STATe

(*Ch*)/

	Status
	(Boolean)
	• •

```
Dim SchRnge As Boolean
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STARt =
1.5E9
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STOP =
1.8E9
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STATe =
True
SchRnge =
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STATe
```

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STARt
SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STOP
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer:FUNCtion:DOMain[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:MARKer:FUNCtion:DOMain[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK:FUNC:DOM ON"
20 OUTPUT 717;":CALC1:MARK:FUNC:DOM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.S

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STOP

(Ch)/

	Value
	(Double)
	0
	HzdBms

```
Dim SchStop As Double
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STOP =
1.8E9
SchStop =
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STOP
```

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STARt
SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STATe
SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer:FUNCtion:DOMain:STOP <numeric>

:CALCulate{[1]-36}[:SELected]:MARKer:FUNCtion:DOMain:STOP?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK:FUNC:DOM:STOP 1.8E9"
20 OUTPUT 717;":CALC1:MARK:FUNC:DOM:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.REFerence.STATe

SCPI.CALCulate(*Ch*).SELected.MARKer.REFerence.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.REFerence.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim RefMode As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer.REFerence.STATe = True
RefMode = SCPI.CALCulate(1).SELected.MARKer.REFerence.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Marker > Ref Marker Mode

SCPI

:CALCulate{[1]-36}[:SELected]:MARKer:REFerence[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:MARKer:REFerence[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:MARK:REF ON"
20 OUTPUT 717;":CALC1:MARK:REF?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MATH.FUNCtion

SCPI.CALCulate(*Ch*).SELected.MATH.FUNCtion = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MATH.FUNCtion

(Ch)/

	Param
	(String)
	• “NORMal” • “DIVide” */ Mem* • “MULTiply” *x Mem* • “SUBTract” *- Mem* • “ADD” *+ Mem* *Mem*
	“NORMal”

```
Dim MathFunc As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MATH.FUNCtion = "div"
MathFunc = SCPI.CALCulate(1).SELected.MATH.FUNCtion
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Display > **Data Math** > **OFF|Data / Mem|Data * Mem|Data - Mem|Data + Mem**

SCPI

:CALCulate{[1]-36}[:SELected]:MATH:FUNCtion {NORMal|SUBTract|DIVide|ADD|MULTiply}

:CALCulate{[1]-36}[:SELected]:MATH:FUNCtion?

{NORM|DIV|MULT|SUBT|ADD}<newline><^END>

```
10 OUTPUT 717;":CALC1:MATH:FUNC DIV"
20 OUTPUT 717;":CALC1:MATH:FUNC?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MATH.MEMorize

SCPI.CALCulate(*Ch*).SELected.MATH.MEMorize

(*Ch*)

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MATH.MEMorize

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Display > **Data** - **> Mem**

SCPI

:CALCulate{[1]-36}[:SELected]:MATH:MEMorize

```
10 OUTPUT 717;":CALC1:MATH:MEM"
```

SCPI.CALCulate(*Ch*).SELected.MIXer.XAXis

SCPI.CALCulate(*Ch*).SELected.MIXer.XAXis = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MIXer.XAXis

(Ch)/X

	Param
	X
	(String)
	• “NORMal” • “RFPLo” *RF* + *LO* • “RFMLo” *RF* - *LO* • “LOMRf” *LO* - *RF*
	“NORMal”

```
Dim Xaxis As String
SCPI.SENSe.OFFSet.STATe = False
SCPI.CALCulate(1).SELected.MIXer.XAXis = "RFPLo"
Xaxis = SCPI.CALCulate(1).SELected.MIXer.XAXis
```

SCPI.SENSe(Ch).OFFSet.STATe

SCPI.CALCulate(Ch).SELected.OFFset.XAXis

Sweep Setup > **Frequency Offset** > **X-Axis** > **Normal|RF+LO|RF-LO|LO-RF**

SCPI

:CALCulate{[1]-36}[:SELected]:MIXer:XAXis {NORMal|RFPLo|RFMLo|LOMRf}

:CALCulate{[1]-36}[:SELected]:MIXer:XAXis?

{NORM|RFPL|RFML|LOMR}<newline><^END>

```
10 OUTPUT 717;":CALC1:MIX:XAX RFPL"
20 OUTPUT 717;":CALC1:MIX:XAX?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MSTatistics.DATA

Data = SCPI.CALCulate(*Ch*).SELected.MSTatistics.DATA

(*Ch*)

	Data
	3 • *0* • *1* • *2* 0
	(Variant)

```
Dim MstData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
MstData = SCPI.CALCulate(1).SELected.MSTatistics.DATA
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MSTatistics.STATe

SCPI

:CALCulate{[1]-36}[:SELected]:MSTatistics:DATA?

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

{numeric 1}

{numeric 2}

{numeric 3}

```
10 OUTPUT 717;":CALC1:MST:DATA?"
20 ENTER 717;A,B,C
```

SCPI.CALCulate(*Ch*).SELected.MSTatistics.STATe

SCPI.CALCulate(*Ch*).SELected.MSTatistics.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MSTatistics.STATe

/ (*Ch*)

	Status
	/
	(Boolean)
	• •

```
Dim Mst As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MSTatistics.STATe = True
Mst = SCPI.CALCulate(1).SELected.MSTatistics.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MSTatistics.DATA

Marker Fctn > Statistics

SCPI

:CALCulate{[1]-36}[:SELected]:MSTatistics[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:MSTatistics[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:MST ON"
20 OUTPUT 717;":CALC1:MST?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.OFFset.XAXis

SCPI.CALCulate(*Ch*).SELected.OFFSet.XAXis = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.OFFSet.XAXis

(Ch)/X

	Param
	X
	(String)
	• “BASE” • “STIMulus” • “RESPonse”
	“RESPonse”

```
Dim Xaxis As String
SCPI.SENSe.OFFSet.STATe = True
SCPI.CALCulate(1).SELected.OFFSet.XAXis = "STIMulus"
Xaxis = SCPI.CALCulate(1).SELected.OFFSet.XAXis
```

SCPI.SENSe(Ch).OFFSet.STATe

SCPI.CALCulate(Ch).SELected.MIXer.XAXis

Sweep Setup > **Frequency Offset** > **X-Axis** > **Base|Stimulus|Response**

SCPI

:CALCulate{[1]-36}[:SELected]:OFFSet:XAXis {BASE|STIMulus|RESPonse}

:CALCulate{[1]-36}[:SELected]:OFFSet:XAXis?

{BASE|STIMulus|RESPonse}<newline><^END>

```
10 OUTPUT 717;":CALC1:OFFS:XAX RESP"
20 OUTPUT 717;":CALC1:OFFS:XAX?"
30 ENTER 717;A$
```

SCPI.CALCulate(Ch).SELected.RLIMit.DATA

```
SCPI.CALCulate(Ch).SELected.RLIMit.DATA = Data
Data = SCPI.CALCulate(Ch).SELected.RLIMit.DATA
```

(*Ch*)/ SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.FORMat.DATA

	Status
	1 + Num\ 4n1Num • *0* 0120 *0* • *nx4-3* n 01 0: 1: • *nx4-2* n// • *nx4-1* n// • *nx4* ndB 0
	(Variant)
	1+Num*4 *n \ 4-3* 01 *n* 4-2 n * 4-1*

1

```
Dim RLimData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.DATA = Array(1,1,1E6,1E9,0)
RLimData = SCPI.CALCulate(1).SELected.RLIMit.DATA

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.DATA = Array(0) ''' Clear Ripple
Limit Table
```

2

```
Dim RlimData(5) As Variant
Dim Ref As Variant
RLimData(0) = 1
RLimData(1) = 1
RLimData(2) = 1e6
RLimData(3) = 1e9
RLimData(4) = 0
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.DATA = RLimData
Ref = SCPI.CALCulate(1).SELected.RLIMit.DATA
```

```
Dim RLimData(0) as Variant
RLimData(0) = 0
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.DATA = RLimData ''' Clear Ripple
Limit Table
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

Analysis > **Ripple Limit** > **Edit Ripple Limit** > **Add**

SCPI

:CALCulate{[1]-36}[:SELected]:RLIMit:DATA <numeric 1>,?/span> ,
<numeric 1+(N\ 4)>

:CALCulate{[1]-36}[:SELected]:RLIMit:DATA?

{numeric 1},?/span> ,{numeric 1+(N\ 4)}<newline><^END>

```
10 DIM B(1:2,1:4)
20 OUTPUT 717;":CALC1:RLIM:DATA 2,1,1E9,3E9,3,1,5E9,7E9,3"
30 OUTPUT 717;":CALC1:RLIM:DATA?"
40 ENTER 717;A,B(*)
```

10 OUTPUT 717;":CALC1:RLIM:DATA 0" ! Clear Ripple Limit Table

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE

```
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE = Status
Status = SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE
```

(*Ch*)/ SCPI.CALCulate(Ch).PARameter(Tr).SELect

	Status
	/
	(Boolean)
	• •

```
Dim RLimDisp As Boolean
SCPI.CALCulate(1).SELected.RLIMit.DISPlay.LINE = True
RLimDisp = SCPI.CALCulate(1).SELected.RLIMit.DISPLay.LINE
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.RLIMit.STATe
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue

Analysis > Ripple Limit > Ripple Limit

SCPI

:CALCulate{[1]-36}[:SELected]:RLIMit:DISPlay:LINE {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:RLIMit:DISPlay:LINE?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:RLIM:DISP:LINE ON"
20 OUTPUT 717;":CALC1:RLIM:DISP:LINE?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect

```
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect = Value
Value = SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect
```

(*Ch*)/ SCPI.CALCulate(Ch).PARameter(Tr).SELect

	Value
	(Long)
	112
	1

```
Dim RBand As Long
SCPI.CALCulate(1).SELected.RLIMit.DISPlay.SELect = 2
RBand = SCPI.CALCulate(1).SELected.RLIMit.DISPlay.SELect
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.RLIMit.STATe
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue

Analysis > Ripple Limit > Ripple Band

SCPI

:CALCulate{[1]-36}[:SELected]:RLIMit:DISPlay:SELect <numeric>

:CALCulate{[1]-36}[:SELected]:RLIMit:DISPlay:SELect?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:RLIM:DISP:SEL 5"
20 OUTPUT 717;":CALC1:RLIM:DISP:SEL?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue

```
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue = Param
Param = SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue
```

(*Ch*)/ SCPI.CALCulate(Ch).PARameter(Tr).SELect

	Param
	(String)
	• “OFF” • “ABSolute” • “MARgin”
	“OFF”

```
Dim RDisp As String
SCPI.CALCulate(1).SELected.RLIMit.DISPlay.VALue = "ABSolute"
RDisp = SCPI.CALCulate(1).SELected.RLIMit.DISPlay.VALue
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.RLIMit.STATe
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect

Analysis > **Ripple Limit** > **Ripple Value > OFF|Absolute|Margin**

SCPI

:CALCulate{[1]-36}[:SELected]:RLIMit:DISPlay:VALue {OFF|ABS|MAR}

:CALCulate{[1]-36}[:SELected]:RLIMit:DISPlay:VALue?

{OFF|ABS|MAR}<newline><^END>

```
10 OUTPUT 717;":CALC1:RLIM:DISP:VAL ABS"
20 OUTPUT 717;":CALC1:RLIM:DISP:VAL?"
30 ENTER 717;A$
```

SCPI.CALCulate(Ch).SELected.RLIMit.FAIL

Status = SCPI.CALCulate(Ch).SELected.RLIMit.FAIL

(*Ch*) SCPI.CALCulate(Ch).PARameter(Tr).SELect

	Status
	(Boolean)
	• •
	OFF

```
Dim Result As Boolean
Result = SCPI.CALCulate(1).SELected.RLIMit.FAIL
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

SCPI

:CALCulate{[1]-36}[:SELected]:RLIMit:FAIL?

```
10 OUTPUT 717;":CALC1:RLIM:FAIL?"
20 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.RLIMit.REPort.DATA

Data = SCPI.CALCulate(Ch).SELected.RLIMit.REPort.DATA

(*Ch*) SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.FORMat.DATA

	Status
	1 + Num\ 3n112 • *0* • *nx3-2* • *nx3-1* • *nx3* 0: 1: 0
	(Variant)

```
Dim RData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
RData = SCPI.CALCulate(1).SELected.RLIMit.REPort.DATA
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

SCPI

:CALCulate{[1]-36}[:SELected]:RLIMit:REPort[:DATA]?

{numeric 1},?/span> ,{numeric 1+N\ 3}<newline><^END>

<numeric 1>	112
<numeric 1+ (n\ 3)-2>	
<numeric 1+ (n\ 3)-1>	
<numeric 1+(n\ 3)>	0: 1:

N<numeric 1>n112

```
10 DIM B(1:2,1:3)
20 OUTPUT 717;":CALC1:RLIM:REP?"
30 ENTER 717;A,B(*)
```

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

```
SCPI.CALCulate(Ch).SELected.RLIMit.STATe = Status
Status = SCPI.CALCulate(Ch).SELected.RLIMit.STATe
```

/Ch SCPI.CALCulate(Ch).PARameter(Tr).SELect

	Status
	/
	(Boolean)
	• •

```
Dim RLimTest As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.STATe = True
RLimTest = SCPI.CALCulate(1).SELected.RLIMit.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.RLIMit.DATA
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect
SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue
SCPI.CALCulate(Ch).SELected.RLIMit.FAIL
SCPI.CALCulate(Ch).SELected.RLIMit.REPort.DATA

SCPI

:CALCulate{[1]-36}[:SELected]:RLIMit[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:RLIMit[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:RLIM ON"
20 OUTPUT 717;":CALC1:RLIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.SMOothing.APERture

SCPI.CALCulate(*Ch*).SELected.SMOothing.APERture = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.SMOothing.APERture

/ (*Ch*)

	Value
	(Double)
	0.0525
	1.5

```
Dim SmoAper As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.SMOothing.APERture = 2.5
SmoAper = SCPI.CALCulate(1).SELected.SMOothing.APERture
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.SMOothing.STATe

Avg > Smo Aperture

SCPI

:CALCulate{[1]-36}[:SELected]:SMOothing:APERture <numeric>

:CALCulate{[1]-36}[:SELected]:SMOothing:APERture?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:SMO:APER 2.5"
20 OUTPUT 717;":CALC1:SMO:APER?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.SMOothing.STATe

SCPI.CALCulate(*Ch*).SELected.SMOothing.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.SMOothing.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •
	0

```
Dim Smo As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.SMOothing.STATe = True
Smo = SCPI.CALCulate(1).SELected.SMOothing.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.SMOothing.APERture

Avg > Smoothing

SCPI

:CALCulate{[1]-36}[:SELected]:SMOothing[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:SMOothing[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:SMO:STAT ON"
20 OUTPUT 717;":CALC1:SMO:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.CENTer

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.CENTer = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.CENTer

(Ch)/

	Value
	(Double)
	0
	s

```
Dim Cent As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.CENTer = 1E-8
Cent = SCPI.CALCulate(1).SELected.TRANsform.TIME.CENTer
```

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.SPAN
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Center**

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:CENTer <numeric>

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:CENTer?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:TRAN:TIME:CENT 1E-8"
20 OUTPUT 717;":CALC1:TRAN:TIME:CENT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.IMPulse.W

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.IMPulse.WIDTh = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.IMPulse.WIDTh

(*Ch*)/Kayser Bessel

	Value
	(Double)
	s

```
Dim ImpWid As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.IMPulse.WIDTh = 1E-
10
ImpWid =
SCPI.CALCulate(1).SELected.TRANsform.TIME.IMPulse.WIDTh
```

SCPI.CALCulate(Ch).SELected.TRANsform.TIME. KBESsel
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STEP.RTIMe
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Center**

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:IMPulse:WIDTh <numeric>

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:IMPulse:WIDTh?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:TRAN:TIME:IMP:WIDT 1E-10"
20 OUTPUT 717;":CALC1:TRAN:TIME:IMP:WIDT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.KBESsel

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.KBESsel = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.KBESsel

(*Ch*)/Kayser Bessel

	Value
	b
	(Double)
	013
	6

```
Dim Beta As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.KBESsel = 3
Beta = SCPI.CALCulate(1).SELected.TRANsform.TIME.KBESsel
```

SCPI.CALCulate(Ch).SELected.TRANsform.TIME. IMPulse.WIDTh
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STEP.RTIMe
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Center**

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:KBESsel <numeric>

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:KBESsel?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:TRAN:TIME:KBES 3"
20 OUTPUT 717;":CALC1:TRAN:TIME:KBES?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.LPFReque

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.LPFRequency = *Value*

(*Ch*)

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.LPFRequency

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.TYPE
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Set Freq Low pass**

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:LPFRequency

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.SPAN

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.SPAN = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.SPAN

(*Ch*)

	Value
	(Double)
	2E-8
	s

```
Dim Span As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.SPAN = 1E-8
Cent = SCPI.CALCulate(1).SELected.TRANsform.TIME.SPAN
```

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.CENTer
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Center**

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:SPAN <numeric>

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:SPAN?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:TRAN:TIME:SPAN 1E-8"
20 OUTPUT 717;":CALC1:TRAN:TIME:SPAN?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STARt

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STARt

(Ch)/

	Value
	(Double)
	-1E-8
	s

```
Dim Star As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STARt = 0
Star = SCPI.CALCulate(1).SELected.TRANsform.TIME.STARt
```

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STOP
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Start**

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STARt <numeric>

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STARt?

{numeric}<newline><^END>

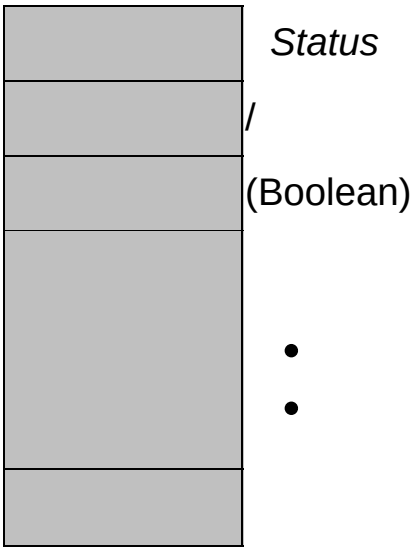
```
10 OUTPUT 717;":CALC1:TRAN:TIME:STAR 0"
20 OUTPUT 717;":CALC1:TRAN:TIME:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STATe

(*Ch*)/
333


Status
/
(Boolean)

```
Dim Trans As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STATe = True
Trans = SCPI.CALCulate(1).SELected.TRANsform.TIME.STATe
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.SENSe(Ch).SWEep.TYPE
SCPI.SENSe(Ch).SWEep.POINts

Analysis > Transform > Transform

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STATe {ON|OFF|1|0}

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CALC1:TRAN:TIME:STAT ON"
20 OUTPUT 717;":CALC1:TRAN:TIME:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STEP.RTIM

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STEP.RTIMe = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STEP.RTIMe

/Kayser Bessel (*Ch*)

	Value
	(Double)
	s

```
Dim RTime As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STEP.RTIMe = 1E-10
RTime = SCPI.CALCulate(1).SELected.TRANsform.TIME.STEP.RTIMe
```

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.IMPulse.WIDTh
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.KBESsel
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Center**

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STEP:RTIMe <numeric>

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STEP:RTIMe?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:TRAN:TIME:STEP:RTIM 1E-10"
20 OUTPUT 717;":CALC1:TRAN:TIME:STEP:RTIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STIMulus

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STIMulus = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STIMulus

(Ch)/

	Param
	(String)
	• “IMPulse” • “STEP”
	“IMPulse”

```
Dim StimType As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STIMulus = "step"
StimType = SCPI.CALCulate(1).SELected.TRANsform.TIME.STIMulus
```

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.TYPE
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Type** > **Bandpass|Lowpass Step|Lowpass Imp.**

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STIMulus {IMPulse|STEP}

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STIMulus?

{IMP|STEP}<newline><^END>

```
10 OUTPUT 717;":CALC1:TRAN:TIME LPAS"
20 OUTPUT 717;":CALC1:TRAN:TIME:STIM STEP"
30 OUTPUT 717;":CALC1:TRAN:TIME:STIM?"
40 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STOP

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STOP

(Ch)/

	Value
	(Double)
	1E-8
	s

```
Dim Span As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STOP = 2E-8
Cent = SCPI.CALCulate(1).SELected.TRANsform.TIME.STOP
```

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STARt
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Stop**

SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STOP <numeric>

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME:STOP?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CALC1:TRAN:TIME:STOP 2E-8"
20 OUTPUT 717;":CALC1:TRAN:TIME:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.TYPE

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.TYPE

(Ch)/

	Param
	(String)
	• “BPASs” • “LPASs”
	“BPASs”

```
Dim Typ As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.SHAPe = "lpas"
Typ = SCPI.CALCulate(1).SELected.TRANsform.TIME.SHAPe
```

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STIMulus
SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
SCPI.CALCulate(Ch).PARameter(Tr).SELect

Analysis > **Transform** > **Type** > **Bandpass|Lowpass Step|Lowpass Imp.**

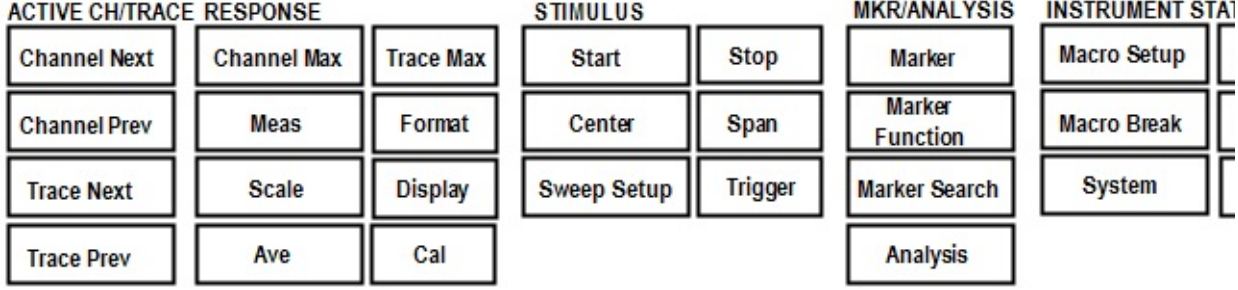
SCPI

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME[:TYPE] {BPASs|LPASs}

:CALCulate{[1]-36}[:SELected]:TRANsform:TIME[:TYPE]?

{BPAS|LPAS}<newline><^END>

```
10 OUTPUT 717;":CALC1:TRAN:TIME LPAS"
20 OUTPUT 717;":CALC1:TRAN:TIME?"
30 ENTER 717;A$
```



		COM	SCPI
Start	Start	SCPI.SENSe(Ch).FREQuency.STARt	:SENS{1-36}:FREQ:STAR
Stop	Stop	SCPI.SENSe(Ch).FREQuency.STOP	:SENS{1-36}:FREQ:STOP
Center	Center	SCPI.SENSe(Ch).FREQuency.CENTer	:SENS{1-36}:FREQ:CENT
Span	Span	SCPI.SENSe(Ch).FREQuency.SPAN	:SENS{1-36}:FREQ:SPAN

COM| SCPI

Analysis			COM
Fixture Simulator			
Fixture Simulator			SCPI.CALCulate(Ch).FSIMulator.STATe
Topology	Device	SE-Bal	SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice SCPI.CALCulate(C
Topology	Device	Bal-Bal	SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice SCPI.CALCulate(Ch).FSIMulator.BALun.TOPology.BBALanced.PPC
Topology	Device	SE-SE-Bal	SCPI.CALCulate(Ch).FSIMulator.BALun.DEVice SCPI.CALCulate(Ch).FSIMulator.BALun.TOPology.SSBalanced.PPC
Topology	Port x (se|bal)		See Analysis > Fixture Simulator > Topology > Device
Topology	Property		SCPI.CALCulate(Ch).FSIMulator.BALun.TOPology.PROPerty.STATe
BalUn			SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).STATe
Balun OFF ALL Trace			None
Balun ON ALL Trace			None
Measurement			See Meas key
Port Matching	Port Matching		SCPI.CALCulate(Ch).FSIMulator.STATe
Port Matching	Select Port		SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).TYP
Port Matching	Select Circuit		SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).TYP
Port Matching	C	SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARameters.C	
Port Matching	G	SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARameters.G	
Port Matching	L	SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARameters.L	
Port Matching	R	SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).PARameters.R	
Port Matching	User File	SCPI.CALCulate(Ch).FSIMulator.SENDed.PMCircuit.PORT(Pt).USER.FILename	
Port ZConversion	Port Z Conversion	SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.STATe	
Port ZConversion	Port x Z0 Real	SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).Z0.R SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).REAL	
Port ZConversion	Port x Z0 Imag	SCPI.CALCulate(Ch).FSIMulator.SENDed.ZCONversion.PORT(Pt).IMAGinary	

De-Embedding	De-Embedding	SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.STATe
De-Embedding	Select Port	SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.PORT(Pt).TYPE
De-Embedding	Select Type	SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.PORT(Pt).TYPE
De-Embedding	User File	SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.PORT(Pt).USER.FILename
Diff Matching	Diff. Matching	SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.STATe
Diff Matching	Select Bal Port	SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).TYPE
Diff Matching	Select Circuit	SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).TYPE
Diff Matching	C	SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARameters.C
Diff Matching	G	SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARameters.G
Diff Matching	L	SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARameters.L
Diff Matching	R	SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).PARameters.R
Diff Matching	User File	SCPI.CALCulate(Ch).FSIMulator.BALun.DMCircuit.BPORt(Bpt).USER.FILename
Diff Zconversion	Diff Z Conversion	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion. STATe
Diff Zconversion	Port1 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).Z0.R
Diff Zconversion	Port1 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).REAL
Diff Zconversion	Port1 (bal) Imag	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).IMAGinary
Diff Zconversion	Port2 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).Z0.R
Diff Zconversion	Port2 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).REAL
Diff Zconversion	Port2 (bal) Imag	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).IMAGinary
Diff Zconversion	Port3 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).Z0.R
Diff Zconversion	Port3 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).REAL

Diff Zconversion	Port3 (bal) Imag	SCPI.CALCulate(Ch).FSIMulator.BALun.DZConversion.BPORt(Bpt).IMAGinary	
Cmn Zconversion	Cmn Z Conversion	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion. STATe	
Cmn Zconversion	Port1 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).Z0.R	
Cmn Zconversion	Port1 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).REAL	
Cmn Zconversion	Port1 (bal) Imag	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).IMAGinary	
Cmn Zconversion	Port2 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).Z0.R	
Cmn Zconversion	Port2 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).REAL	
Cmn Zconversion	Port2 (bal) Imag	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).IMAGinary	
Cmn Zconversion	Port3 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).Z0.R	
Cmn Zconversion	Port3 (bal) Real	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).REAL	
Cmn Zconversion	Port3 (bal) Imag	SCPI.CALCulate(Ch).FSIMulator.BALun.CZConversion.BPORt(Bpt).IMAGinary	
De-Embedding S4P	De-Embedding S4P		SCPI.CALCulate(Ch).FSIMulator.EMBed.STATe
De-Embedding S4P	Topology	Select Topology	SCPI.CALCulate(Ch).FSIMulator.EMBed.TYPE
De-Embedding S4P	Topology	Ports	SCPI.CALCulate(Ch).FSIMulator.EMBed.TOPology.A.PORTs
De-Embedding S4P	Topology	Ports	SCPI.CALCulate(Ch).FSIMulator.EMBed.TOPology.B.PORTs
De-Embedding S4P	Topology	Type (nwk1)	SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).TYPE
De-Embedding S4P	Topology	User File (nwk1)	SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).FILename
De-Embedding S4P	Topology	Type (nwk2)	SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).TYPE
De-Embedding S4P	Topology	User File (nwk2)	SCPI.CALCulate(Ch).FSIMulator.EMBed.NETWork(Nwk).FILename
Gating			
Gating			SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe
Start			SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STARt

Stop			SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STOP
Center			SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.CENTer
Span			SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.SPAN
Type			SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.TYPE
Shape	Maximum		SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.SHAPe
Shape	Wide		SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.SHAPe
Shape	Normal		SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.SHAPe
Shape	Minimum		SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.SHAPe
Transform			
Transform			SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe
Start			SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STARt
Stop			SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STOP
Center			SCPI.CALCulate(Ch).SELected.TRANsform.TIME.CENTer
Span			SCPI.CALCulate(Ch).SELected.TRANsform.TIME.SPAN
Set Freq Low Pass			SCPI.CALCulate(Ch).SELected.TRANsform.TIME.LPFRequency
Type	Bandpass		SCPI.CALCulate(Ch).SELected.TRANsform.TIME.TYPE
Type	Lowpass Step		SCPI.CALCulate(Ch).SELected.TRANsform.TIME.TYPE SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STIMulus
Type	Lowpass Imp		SCPI.CALCulate(Ch).SELected.TRANsform.TIME.TYPE SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STIMulus
Window	Maximum		SCPI.CALCulate(Ch).SELected.TRANsform.TIME.KBESsel
Window	Normal		SCPI.CALCulate(Ch).SELected.TRANsform.TIME.KBESsel
Window	Minimum		SCPI.CALCulate(Ch).SELected.TRANsform.TIME.KBESsel
Window	Impulse Width		SCPI.CALCulate(Ch).SELected.TRANsform.TIME.IMPulse.WIDTh
Window	Rise Time		SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STEP.RTIMe
Window	Kaiser Beta		SCPI.CALCulate(Ch).SELected.TRANsform.TIME.KBESsel
Conversion			

Conversion			SCPI.CALCulate(Ch).SELected.CONVersion.STATe
Function			SCPI.CALCulate(Ch).SELected.CONVersion.FUNCtion
Limit Test			
Limit Test			SCPI.CALCulate(Ch).SELected.LIMit.STATe
Limit Line			SCPI.CALCulate(Ch).SELected.LIMit.DISPlay.STATe
Edit Limit Line			SCPI.CALCulate(Ch).SELected.LIMit.DATA
Edit Limit Line	Export to CSV File		SCPI.MMEMory.STORe.LIMit
Edit Limit Line	Import from CSV File		SCPI.MMEMory.LOAD.LIMit
Limit Line Offset	Stimulus Offset		SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.STIMulus
Limit Line Offset	Amplitude Offset		SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude
Limit Line Offset	Marker -> Amplitude Offset		SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.MARKer
Fail Sign			SCPI.DISPlay.FSIGn
Ripple Limit			
Ripple Limit Test			SCPI.CALCulate(Ch).SELected.RLIMit.STATe
Ripple Limit			SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE
Ripple Value			SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue
Ripple Band			SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect
Edit Ripple Limit			SCPI.CALCulate(Ch).SELected.RLIMit.DATA
Edit Ripple Limit	Export to CSV File		SCPI.MMEMory.STORe.RLIMit
Edit Ripple Limit	Import from CSV File		SCPI.MMEMory.LOAD.RLIMit
Fail Sign			SCPI.DISPlay.FSIGn
Bandwidth Limit			
BW Test			SCPI.CALCulate(Ch).SELected.BLIMit.STATe
BW Display			SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.VALue

BW Marker			SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.MARKer
N dB Points			SCPI.CALCulate(Ch).SELected.BLIMit.DB
Min Bandwidth			SCPI.CALCulate(Ch).SELected.BLIMit.MINimum
Max Bandwidth			SCPI.CALCulate(Ch).SELected.BLIMit.MAXimum
Fail Sign			SCPI.DISPlay.FSIGn

COM | SCPI

Analysis				SCPI
Fixture Simulator	Fixture Simulator			:CALC{1-36}:FSIM:STAT
Fixture Simulator	Topology	Device	SE-Bal	:CALC{1-36}:FSIM:BAL:DEV SBAL, :CALC{1-
Fixture Simulator	Topology	Device	Bal-Bal	:CALC{1-36}:FSIM:BAL:DEV BBAL, :CALC{1-36}:FSIM:BAL:TOP:BBAL
Fixture Simulator	Topology	Device	SE-SE-Bal	:CALC{1-36}:FSIM:BAL:DEV SSB, :CALC{1-36}:FSIM:BAL:TOP:SSB
Fixture Simulator	Topology	Port x (se|bal)	See Analysis > Fixture Simulator > Topology > Device	
Fixture Simulator	Topology	Property		:CALC{1-36}:FSIM:BAL:TOP:PROP:STAT
Fixture Simulator	BalUn			:CALC{1-36}:FSIM:BAL:PAR{1-16}:STAT
Fixture Simulator	Balun OFF ALL Trace			None
Fixture Simulator	Balun ON ALL Trace			None
Fixture Simulator	Measurement			See Meas key
Fixture Simulator	Port Matching	Port Matching		:CALC{1-36}:FSIM:SEND:PMC:STAT
Fixture Simulator	Port Matching	Select Port		:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}
Fixture Simulator	Port Matching	Select Circuit		:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}
Fixture Simulator	Port Matching	C		:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:PA
Fixture Simulator	Port Matching	G		:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:PA
Fixture Simulator	Port Matching	L		:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:PA
Fixture Simulator	Port Matching	R		:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:PA
Fixture Simulator	Port Matching	User File		:CALC{1-36}:FSIM:SEND:PMC:PORT{1-4}:US
Fixture Simulator	Port ZConversion	Port ZConversion		:CALC{1-36}:FSIM:SEND:ZCON:STAT
Fixture Simulator	Port ZConversion	Port x Z0 Real		:CALC{1-36}:FSIM:SEND:ZCON:PORT{1-4}:Z :CALC{1-36}:FSIM:SEND:ZCON: PORT{1-4}:I
Fixture Simulator	Port ZConversion	Port x Z0 Imag		:CALC{1-36}:FSIM:SEND:ZCON:PORT{1-4}:II
Fixture Simulator	De-Embedding	De-Embedding		:CALC{1-36}:FSIM:SEND:DEEM:STAT
Fixture Simulator	De-Embedding	Select Port		:CALC{1-36}:FSIM:SEND:DEEM:PORT{1-4}

Fixture Simulator	De-Embedding	Select Type		:CALC{1-36}:FSIM:SEND:DEEM:PORT{1-4}
Fixture Simulator	De-Embedding	User File		:CALC{1-36}:FSIM:SEND:DEEM:PORT{1-4}:U
Fixture Simulator	Diff Matching	Diff. Matching		:CALC{1-36}:FSIM:BAL:DMC:STAT
Fixture Simulator	Diff Matching	Select Bal Port		:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}
Fixture Simulator	Diff Matching	Select Circuit		:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}
Fixture Simulator	Diff Matching	C		:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:PAR
Fixture Simulator	Diff Matching	G		:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:PAR
Fixture Simulator	Diff Matching	L		:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:PAR
Fixture Simulator	Diff Matching	R		:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:PAR
Fixture Simulator	Diff Matching	User File		:CALC{1-36}:FSIM:BAL:DMC:BPOR{1-2}:USE
Fixture Simulator	Diff Zconversion	Diff ZConversion		:CALC{1-36}:FSIM:BAL:DZC:STAT
Fixture Simulator	Diff Zconversion	Port1 (bal) Real		:CALC{1-36}:FSIM:BAL:DZC:BPOR1:Z0
Fixture Simulator	Diff Zconversion	Port1 (bal) Real		:CALC{1-36}:FSIM:BAL:DZC:BPOR1:REAL
Fixture Simulator	Diff Zconversion	Port1 (bal) Imag		:CALC{1-36}:FSIM:BAL:DZC:BPOR1:IMAG
Fixture Simulator	Diff Zconversion	Port2 (bal) Real		:CALC{1-36}:FSIM:BAL:DZC:BPOR{1-2}:Z0
Fixture Simulator	Diff Zconversion	Port2 (bal) Real		:CALC{1-36}:FSIM:BAL:DZC:BPOR{1-2}:REA
Fixture Simulator	Diff Zconversion	Port2 (bal) Imag		:CALC{1-36}:FSIM:BAL:DZC:BPOR{1-2}:IMA(
Fixture Simulator	Diff Zconversion	Port3 (bal) Real		:CALC{1-36}:FSIM:BAL:DZC:BPOR1:Z0
Fixture Simulator	Diff Zconversion	Port3 (bal) Real		:CALC{1-36}:FSIM:BAL:DZC:BPOR1:REAL
Fixture Simulator	Diff Zconversion	Port3 (bal) Imag		:CALC{1-36}:FSIM:BAL:DZC:BPOR1:IMAG
Fixture Simulator	Cmn Zconversion	Cmn ZConversion		:CALC{1-36}:FSIM:BAL:CZC:STAT
Fixture Simulator	Cmn Zconversion	Port1 (bal) Real		:CALC{1-36}:FSIM:BAL:CZC:BPOR1:Z0
Fixture Simulator	Cmn Zconversion	Port1 (bal) Real		:CALC{1-36}:FSIM:BAL:CZC:BPOR1:REAL
Fixture Simulator	Cmn Zconversion	Port1 (bal) Imag		:CALC{1-36}:FSIM:BAL:CZC:BPOR1:IMAG
Fixture Simulator	Cmn Zconversion	Port2 (bal) Real		:CALC{1-36}:FSIM:BAL:CZC:BPOR{1-2}:Z0
Fixture Simulator	Cmn Zconversion	Port2 (bal) Real		:CALC{1-36}:FSIM:BAL:CZC:BPOR{1-2}:REA

Fixture Simulator	Cmn Zconversion	Port2 (bal) Imag		:CALC{1-36}:FSIM:BAL:CZC:BPOR{1-2}:IMA(
Fixture Simulator	Cmn Zconversion	Port3 (bal) Real		:CALC{1-36}:FSIM:BAL:CZC:BPOR1:Z0
Fixture Simulator	Cmn Zconversion	Port3 (bal) Real		:CALC{1-36}:FSIM:BAL:CZC:BPOR1:REAL
Fixture Simulator	Cmn Zconversion	Port3 (bal) Imag		:CALC{1-36}:FSIM:BAL:CZC:BPOR1:IMAG
Fixture Simulator	De-Embedding S4P	De-Embedding S4P		:CALC{1-36}:FSIM:EMB:STAT
Fixture Simulator	De-Embedding S4P	Topology	Select Topology	:CALC{1-36}:FSIM:EMB:TYPE {A|B|C}
Fixture Simulator	De-Embedding S4P	Topology	Ports	:CALC{1-36}:FSIM:EMB:TOP:A:PORT {1-4},{1
Fixture Simulator	De-Embedding S4P	Topology	Ports	:CALC{1-36}:FSIM:EMB:TOP:B:PORT {1-4},{1
Fixture Simulator	De-Embedding S4P	Topology	Type (nwk1)	:CALC{1-36}:FSIM:EMB:NETW1:TYPE
Fixture Simulator	De-Embedding S4P	Topology	User File (nwk1)	:CALC{1-36}:FSIM:EMB:NETW1:FIL
Fixture Simulator	De-Embedding S4P	Topology	Type (nwk2)	:CALC{1-36}:FSIM:EMB:NETW2:TYPE
Fixture Simulator	De-Embedding S4P	Topology	User File (nwk2)	:CALC{1-36}:FSIM:EMB:NETW2:FIL
Gating	Gating			:CALC{1-36}:FILT:TIME:STAT
Gating	Start			:CALC{1-36}:FILT:TIME:STAR
Gating	Stop			:CALC{1-36}:FILT:TIME:STOP
Gating	Center			:CALC{1-36}:FILT:TIME:CENT
Gating	Span			:CALC{1-36}:FILT:TIME:SPAN
Gating	Type			:CALC{1-36}:FILT:TIME
Gating	Shape	Maximum		:CALC{1-36}:FILT:TIME:SHAP MAX
Gating	Shape	Wide		:CALC{1-36}:FILT:TIME:SHAP WIDE
Gating	Shape	Normal		:CALC{1-36}:FILT:TIME:SHAP NORM
Gating	Shape	Minimum		:CALC{1-36}:FILT:TIME:SHAP MIN
Transform	Transform			:CALC{1-36}:TRAN:TIME:STAT
Transform	Start			:CALC{1-36}:TRAN:TIME:STAR
Transform	Stop			:CALC{1-36}:TRAN:TIME:STOP
Transform	Center			:CALC{1-36}:TRAN:TIME:CENT
Transform	Span			:CALC{1-36}:TRAN:TIME:SPAN

Transform	Set Freq Low Pass			:CALC{1-36}:TRAN:TIME:LPFR
Transform	Type	Bandpass		:CALC{1-36}:TRAN:TIME BPAS
Transform	Type	Lowpass Step		:CALC{1-36}:TRAN:TIME LPAS :CALC{1-36}:TRAN:TIME:STIM STEP
Transform	Type	Lowpass Imp		:CALC{1-36}:TRAN:TIME LPAS :CALC{1-36}:TRAN:TIME:STIM STEP
Transform	Window	Maximum		:CALC{1-36}:TRAN:TIME:KBES 13
Transform	Window	Normal		:CALC{1-36}:TRAN:TIME:KBES 6
Transform	Window	Minimum		:CALC{1-36}:TRAN:TIME:KBES 0
Transform	Window	Impulse Width		:CALC{1-36}:TRAN:TIME:IMP:WIDT
Transform	Window	Rise Time		:CALC{1-36}:TRAN:TIME:STEP:RTIM
Transform	Window	Kaiser Beta		:CALC{1-36}:TRAN:TIME KBES
Conversion	Conversion			:CALC{1-36}:CONV
Conversion	Function			:CALC{1-36}:CONV:FUNC
Limit Test	Limit Test			:CALC{1-36}:LIM
Limit Test	Limit Line			:CALC{1-36}:LIM:DISP
Limit Test	Edit Limit Line			:CALC{1-36}:LIM:DATA
Limit Test	Edit Limit Line	Export to CSV File		:MMEM:STOR:LIM
Limit Test	Edit Limit Line	Import from CSV File		:MMEM:LOAD:LIM
Limit Test	Limit Line Offset	Stimulus Offset		:CALC{1-36}:LIM:OFFS:STIM
Limit Test	Limit Line Offset	Amplitude Offset		:CALC{1-36}:LIM:OFFS:AMPL
Limit Test	Limit Line Offset	Marker -> Amplitude Offset		:CALC{1-36}:LIM:OFFS:MARK
Limit Test	Fail Sign			:DISP:FSIG
Ripple Limit	Ripple Limit Test			:CALC{1-36}:RLIM
Ripple Limit	Ripple Limit			:CALC{1-36}:RLIM:DISP:LINE
Ripple Limit	Ripple Value			:CALC{1-36}:RLIM:DISP:VAL
Ripple Limit	Ripple Band			:CALC{1-36}:RLIM:DISP:SEL

Ripple Limit	Edit Ripple Limit			:CALC{1-36}:RLIM:DATA
Ripple Limit	Edit Ripple Limit	Export to CSV File		:MMEM:STOR:RLIM
Ripple Limit	Edit Ripple Limit	Import from CSV File		:MMEM:LOAD:RLIM
Ripple Limit	Fail Sign			:DISP:FSIG
Bandwidth Limit	BW Test			:CALC{1-36}:BLIM
Bandwidth Limit	BW Display			:CALC{1-36}:BLIM:DISP:VAL
Bandwidth Limit	BW Marker			:CALC{1-36}:BLIM:DISP:MARK
Bandwidth Limit	N dB Points			:CALC{1-36}:BLIM:DB
Bandwidth Limit	Min Bandwidth			:CALC{1-36}:BLIM:MIN
Bandwidth Limit	Max Bandwidth			:CALC{1-36}:BLIM:MAX
Bandwidth Limit	Fail Sign			:DISP:FSIG

Ave	COM	SCPI
Averaging Restart	SCPI.SENSe(Ch).AVERage.CLEar	:SENS{1-36}:AVER:CL
Avg Factor	SCPI.SENSe(Ch).AVERage.COUNt	:SENS{1-36}:AVER:CO
Averaging	SCPI.SENSe(Ch).AVERage.STATe	:SENS{1-36}:AVER
Ave Trigger	SCPI.TRIGger.SEQuence.AVERage	:TRIG:AVER
Smo Aperture	SCPI.CALCulate(Ch).SELected.SMOothing.APERture	:CALC{1-36}:SMO:APE
Smoothing	SCPI.CALCulate(Ch).SELected.SMOothing.STATe	:CALC{1-36}:SMO:STA
IF Bandwidth	SCPI.SENSe(Ch).BANDwidth.RESolution SCPI.SENSe(Ch).BWIDth.RESolution	:SENS{1-36}:BAND or :SENS{1-36}:BWID

COM | SCPI

Cal				COM
Correction				SCPI.SENSe(Ch).CORRection.STATe
Calibrate	Response (Open)	Select Port	SCPI.SENSe(Ch).CORRection.COLLect.METHod. RESPonse.OPEN	
Calibrate	Response (Open)	Open	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.OPEN	
Calibrate	Response (Open)	Load (Optional)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.LOAD	
Calibrate	Response (Open)	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	Response (Open)	Cancel	OK	SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	Response (Short)	Select Port	SCPI.SENSe(Ch).CORRection.COLLect.METHod. RESPonse.SHORt	
Calibrate	Response (Short)	Short	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.SHORt	
Calibrate	Response (Short)	Load (Optional)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.LOAD	
Calibrate	Response (Short)	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	Response (Short)	Cancel	OK	SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	Response (Thru)	Select Ports	SCPI.SENSe(Ch).CORRection.COLLect.METHod. RESPonse.THRU	
Calibrate	Response (Thru)	Thru	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.THRU	
Calibrate	Response (Thru)	Isolation (Optional)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.ISOLation	
Calibrate	Response (Thru)	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	Response (Thru)	Cancel	OK	SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	Enhanced Response	Select Port	SCPI.SENSe(Ch).CORRection.COLLect.METHod.ERESponse	
Calibrate	Enhanced Response	Open	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.OPEN	
Calibrate	Enhanced Response	Short	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.SHORt	
Calibrate	Enhanced Response	Load	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.LOAD	
Calibrate	Enhanced Response	Thru	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.THRU	
Calibrate	Enhanced Response	Isolation (Optional)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.ISOLation	
Calibrate	Enhanced Response	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	Enhanced Response	Cancel		SCPI.SENSe(Ch).CORRection.COLLect.CLEar

Calibrate	1-Port Cal	Select Port	SCPI.SENSe(Ch).CORRection.COLLect.METHod. SOLT1	
Calibrate	1-Port Cal	Open	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.OPEN	
Calibrate	1-Port Cal	Short	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.SHORt	
Calibrate	1-Port Cal	Load	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.LOAD	
Calibrate	1-Port Cal	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	1-Port Cal	Cancel		SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	2-Port Cal	Select Ports	SCPI.SENSe(Ch).CORRection.COLLect.METHod. SOLT2	
Calibrate	2-Port Cal	Reflection	Port x Open	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.OPI
Calibrate	2-Port Cal	Reflection	Port x Short	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.SH(
Calibrate	2-Port Cal	Reflection	Port x Load	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.LO/
Calibrate	2-Port Cal	Transmission	Port x-y Thru	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.THI
Calibrate	2-Port Cal	Isolation (Optional)	Port x-y Isol	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.ISO
Calibrate	2-Port Cal	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	2-Port Cal	Overwrite	SCPI.SENSe(Ch).CORRection.COLLect.PARTial.SAVE	
Calibrate	2-Port Cal	Cancel		SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	3-Port Cal	Select Ports	SCPI.SENSe(Ch).CORRection.COLLect.METHod. SOLT3	
Calibrate	3-Port Cal	Reflection	Port x Open	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.OPI
Calibrate	3-Port Cal	Reflection	Port x Short	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.SH(
Calibrate	3-Port Cal	Reflection	Port x Load	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.LO/
Calibrate	3-Port Cal	Transmission	Port x-y Thru	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.THI
Calibrate	3-Port Cal	Isolation (Optional)	Port x-y Isol	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.ISO
Calibrate	3-Port Cal	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	3-Port Cal	Overwrite	SCPI.SENSe(Ch).CORRection.COLLect.PARTial.SAVE	
Calibrate	3-Port Cal	Cancel		SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	4-Port Cal	Reflection	Port x Open	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.OPI
Calibrate	4-Port Cal	Reflection	Port x Short	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.SH(
Calibrate	4-Port Cal	Reflection	Port x Load	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.LO/

Calibrate	4-Port Cal	Transmission	Port x-y Thru	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.THF
Calibrate	4-Port Cal	Isolation (Optional)	Port x-y Isol	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.ISO
Calibrate	4-Port Cal	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	4-Port Cal	Overwrite	SCPI.SENSe(Ch).CORRection.COLLect.PARTial.SAVE	
Calibrate	4-Port Cal	Cancel		SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	2-Port TRL Cal	Select Ports		SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRI
Calibrate	2-Port TRL Cal	Thru/Line	x-y Thru/Line	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	2-Port TRL Cal	Reflect	Portx Reflect	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	2-Port TRL Cal	Line/Match	x-y Line/Match	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	2-Port TRL Cal	Line/Match	x-y Fwd (Syx)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	2-Port TRL Cal	Line/Match	x-y Rvs(Sxy)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	2-Port TRL Cal	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	2-Port TRL Cal	Cancel		SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	3-Port TRL Cal	Select Ports	SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL3	
Calibrate	3-Port TRL Cal	Thru/Line	x-y Thru/Line	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	3-Port TRL Cal	Reflect	Portx Reflect	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	3-Port TRL Cal	Line/Match	x-y Line/Match	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	3-Port TRL Cal	Line/Match	x-y Fwd (Syx)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	3-Port TRL Cal	Line/Match	x-y Rvs(Sxy)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	3-Port TRL Cal	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	3-Port TRL Cal	Cancel		SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	4-Port TRL Cal	Thru/Line	x-y Thru/Line	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	4-Port TRL Cal	Reflect	Portx Reflect	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	4-Port TRL Cal	Line/Match	x-y Line/Match	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	4-Port TRL Cal	Line/Match	x-y Fwd (Syx)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI
Calibrate	4-Port TRL Cal	Line/Match	x-y Rvs(Sxy)	SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRI

Calibrate	4-Port TRL Cal	Done		SCPI.SENSe(Ch).CORRection.COLLect.SAVE
Calibrate	4-Port TRL Cal	Cancel		SCPI.SENSe(Ch).CORRection.COLLect.CLEar
Calibrate	Adapter Removal	Select Port		
Calibrate	Adapter Removal	Cal Kit		
Calibrate	Adapter Removal	Open		
Calibrate	Adapter Removal	Short		
Calibrate	Adapter Removal	Load		
Calibrate	Adapter Removal	Adapter Length		
Calibrate	Adapter Removal	Done		
Calibrate	Adapter Removal	Cancel		SCPI.SENSe(Ch).CORRection.COLLect.CLEar
ECal	1-Port ECal	Port 1	SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT1	
ECal	2-Port ECal	Port 1-2	SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT2	
ECal	3-Port ECal	Port 1-2-3	SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT3	
ECal	4-Port ECal		SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT4	
ECal	Thru ECal		SCPI.SENSe(Ch).CORRection.COLLect.ECAL.THRU	
ECal	Enhanced Response		SCPI.SENSe(Ch).CORRection.COLLect.ECAL.ERESponse	
ECal	Isolation		SCPI.SENSe(Ch).CORRection.COLLect.ECAL.ISOLation.STATe	
ECal	Unknown Thru		SCPI.SENSe(Ch).CORRection.COLLect.ECAL.UTHRU.STATe	
ECal	Characterization		SCPI.SENSe(Ch).CORRection.COLLect.ECAL.UCHar	
ECal	Characterization Info			None
ECal	Confidence Check		SCPI.SENSe(Ch).CORRection.COLLect.ECAL.CCHeck.ACQuire	
ECal	Orientation	Auto	SCPI.SENSe(Ch).CORRection.COLLect.ECAL.ORIentation.STATe	
ECal	Orientation	Port 1	SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(Cpt)	
ECal	Orientation	Port 2	SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(Cpt)	
ECal	Orientation	Port 3	SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(Cpt)	
ECal	Orientation	Port 4	SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(Cpt)	
Clear	OK			SCPI.SENSe(Ch).CORRection.CLEar

Property				SCPI.SENSe(Ch).CORRection.PROPerty
Cal Kit	{Calkit Name}		SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect	
Modify Cal Kit				SCPI.CALCulate(Ch).SELected.FORMat
Modify Cal Kit	Define STDs	n. (Std Name)	Label	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	STD Type	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	C0	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	C1	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	C2	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	C3	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	L0	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	L1	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	L2	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	L3	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	Offset Delay	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	Offset Z0	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	Offset Loss	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	Arb. Impedance	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	Min. Frequency	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Define STDs	n. (Std Name)	Max. Frequency	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S

Modify Cal Kit	Define STDs	n. (Std Name)	Media	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(S
Modify Cal Kit	Specify CLSs	Sub Class x	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SELect	
Modify Cal Kit	Specify CLSs	Open	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)	
Modify Cal Kit	Specify CLSs	Short	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)	
Modify Cal Kit	Specify CLSs	Load	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)	
Modify Cal Kit	Specify CLSs	Thru	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cp	
Modify Cal Kit	Specify CLSs	TRL Thru	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLThru(Cpt_m	
Modify Cal Kit	Specify CLSs	TRL Reflect	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLReflect	
Modify Cal Kit	Specify CLSs	TRL Line/Match	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLLine(Cpt_m,	
Modify Cal Kit	TRL Option	Impedance	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.TRLoption.IMPedance	
Modify Cal Kit	TRL Option	Reference Plane	SCPI.SENSe(Ch).CORRection.COLLect.CKIT.TRLoption.RPLane	
Modify Cal Kit	Label Kit		SCPI.SENSe(Ch).CORRection.COLLect.CKIT.LABel	
Modify Cal Kit	Export Cal Kit...			SCPI.MMEMory.STORe.CKIT(Ckit)
Modify Cal Kit	Import Cal Kit...			SCPI.MMEMory.LOAD.CKIT(Ckit)
Modify Cal Kit	Restore Cal Kit		SCPI.SENSe(Ch).CORRection.COLLect.CKIT.RESet	
Port Extensions	Auto Port Extension	Select Ports	SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)	
Port Extensions	Auto Port Extension	Meaure {Std}	SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure	
Port Extensions	Auto Port Extension	Method	Current Span	SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CON
Port Extensions	Auto Port Extension	Method	*Active Marker*	SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CON

Port Extensions	Auto Port Extension	Method	User Span	SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CON
Port Extensions	Auto Port Extension	Method	User Span Start	SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STA
Port Extensions	Auto Port Extension	Method	User Span Stop	SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STC
Port Extensions	Auto Port Extension	Adjust Mismatch	SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset	
Port Extensions	Auto Port Extension	Include		SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOS
Port Extensions	Extensions			SCPI.SENSe(Ch).CORRection.EXTension.STATe
Port Extensions				SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).T
Port Extensions	Loss	Select Ports		None
Port Extensions	Loss	Loss1	SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STAT	
Port Extensions	Loss	Loss1	SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt) .LOSS(Loss)	
Port Extensions	Loss	Freq1	SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)	
Port Extensions	Loss	Loss2	SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STAT	
Port Extensions	Loss	Loss2	SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt) .LOSS(Loss)	
Port Extensions	Loss	Freq2	SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)	
Port Extensions	Loss	Loss at DC	SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC	
Velocity Factor				SCPI.SENSe(Ch).CORRection.RVELocity.COAX
Set Z0				
Power Calibration	Select Port			None
Power Calibration	Correction			SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection.ST

Power Calibration	Take Cal Sweep		SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection. COLLect.ACQuire	
Power Calibration	Abort			None
Power Calibration	Use Sensor			None
Power Calibration	Num of Readings		SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection. COLLect.AVERag	
Power Calibration	Tolerance		SCPI.SOURce.POWer.PORT.CORRection.COLLect. NTOLerance	
Power Calibration	Loss Compen	Compensation	SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection. COLLect.TABLe.L	
Power Calibration	Loss Compen	Delete	SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection. COLLect.TABLe.L	
Power Calibration	Loss Compen	Add	SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection. COLLect.TABLe.L	
Power Calibration	Loss Compen	Clear Loss Table	OK	None
Power Calibration	Loss Compen	Export to CSV File		SCPI.MMEMory.STORe.PLOSs(Pt)
Power Calibration	Loss Compen	Import from CSV File		SCPI.MMEMory.LOAD.PLOSs(Pt)
Power Calibration	Sensor A Settings	Ref Cal Factor	SCPI.SOURce.POWer.PORT.CORRection.COLLect. ASENsor.RCFact	
Power Calibration	Sensor A Settings	Delete	SCPI.SOURce.POWer.PORT.CORRection.COLLect. TABLe.ASENsor.	
Power Calibration	Sensor A Settings	Add	SCPI.SOURce.POWer.PORT.CORRection.COLLect. TABLe.ASENsor.	
Power Calibration	Sensor A Settings	Clear Loss Table	OK	None
Power Calibration	Sensor A Settings	Export to CSV File		SCPI.MMEMory.STORe.ASCFactor
Power Calibration	Sensor A Settings	Import from CSV File		SCPI.MMEMory.LOAD.ASCFactor
Power Calibration	Sensor B Settings	Ref Cal Factor	SCPI.SOURce.POWer.PORT.CORRection.COLLect. BSENsor.RCFact	
Power	Sensor B Settings	Delete	SCPI.SOURce.POWer.PORT.CORRection.COLLect. TABLe.BSENsor.	

Calibration				
Power Calibration	Sensor B Settings	Add	SCPI.SOURce.POWer.PORT.CORRection.COLLect. TABLe.BSENsor.	
Power Calibration	Sensor B Settings	Clear Loss Table	OK	None
Power Calibration	Sensor B Settings	Export to CSV File		SCPI.MMEMory.STORe.BSCFactor
Power Calibration	Sensor B Settings	Import from CSV File		SCPI.MMEMory.LOAD.BSCFactor
Receiver Calibration	Select Port			None
Receiver Calibration	Correction			SCPI.SENSe(Ch).CORRection.RECeiver(Pt).STATe
Receiver Calibration	Source Port			None
Receiver Calibration	Take Cal Sweep		SCPI.SENSe(Ch).CORRection.RECeiver(Pt).COLLect.ACQuire	
Mixer/ Converter Calibration	Scalar Cal (Manual)	Select Ports	SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.METHod.SOLT1	
Mixer/ Converter Calibration	Scalar Cal (Manual)	Select Ports	SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.METHod.SMIX2	
Mixer/ Converter Calibration	Scalar Cal (Manual)	Reflection	SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.OPEN SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.SHORt SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.LOAD	
Mixer/ Converter Calibration	Scalar Cal (ECal)	Select Ports		None
Mixer/ Converter Calibration	Scalar Cal (ECal)	Power Meter	Use Sensor	SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQ
Mixer/ Converter Calibration	Scalar Cal (ECal)	ECal & Done	SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ECAL.SMIX2 SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ECAL.SOLT1	
Mixer/ Converter Calibration	Scalar Cal (ECal)	Cancel	OK	SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.CLE
Mixer/ Converter Calibration	Clear	OK		SCPI.SENSe(Ch).CORRection.OFFSet.CLEar

Cal Trig Source	Internal			SCPI.SENSe(Ch).CORRection.TRIGger.FREE.STATe
Cal Trig Source	System			SCPI.SENSe(Ch).CORRection.TRIGger.FREE.STATe

COM | SCPI

Cal				SCPI
Correction				:SENS{1-36}:CORR:STAT
Calibrate	Response (Open)	Select Port		:SENS{1-36}:CORR:COLL:METH:OPEN {1-4
Calibrate	Response (Open)	Open		:SENS{1-36}:CORR:COLL:OPEN
Calibrate	Response (Open)	Load (Optional)		:SENS{1-36}:CORR:COLL:LOAD
Calibrate	Response (Open)	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	Response (Open)	Cancel	OK	:SENS{1-36}:CORR:COLL:CLE
Calibrate	Response (Short)	Select Port		:SENS{1-36}:CORR:COLL:METH:SHOR {1-4
Calibrate	Response (Short)	Short		:SENS{1-36}:CORR:COLL:SHOR
Calibrate	Response (Short)	Load (Optional)		:SENS{1-36}:CORR:COLL:LOAD
Calibrate	Response (Short)	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	Response (Short)	Cancel	OK	:SENS{1-36}:CORR:COLL:CLE
Calibrate	Response (Thru)	Select Ports		:SENS{1-36}:CORR:COLL:METH:THRU {1-4
Calibrate	Response (Thru)	Thru		:SENS{1-36}:CORR:COLL:THRU
Calibrate	Response (Thru)	Isolation (Optional)		:SENS{1-36}:CORR:COLL:ISOL
Calibrate	Response (Thru)	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	Response (Thru)	Cancel	OK	:SENS{1-36}:CORR:COLL:CLE
Calibrate	Enhanced Response	Select Port		:SENS{1-36}:CORR:COLL:METH:ERES {1-4
Calibrate	Enhanced Response	Open		:SENS{1-36}:CORR:COLL:OPEN
Calibrate	Enhanced Response	Short		:SENS{1-36}:CORR:COLL:SHOR
Calibrate	Enhanced Response	Load		:SENS{1-36}:CORR:COLL:LOAD
Calibrate	Enhanced Response	Thru		:SENS{1-36}:CORR:COLL:THRU
Calibrate	Enhanced Response	Isolation (Optional)		:SENS{1-36}:CORR:COLL:ISOL
Calibrate	Enhanced Response	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	Enhanced Response	Cancel		:SENS{1-36}:CORR:COLL:CLE

Calibrate	1-Port Cal	Select Port		:SENS{1-36}:CORR:COLL:METH:SOLT1 {1-
Calibrate	1-Port Cal	Open		:SENS{1-36}:CORR:COLL:OPEN
Calibrate	1-Port Cal	Short		:SENS{1-36}:CORR:COLL:SHOR
Calibrate	1-Port Cal	Load		:SENS{1-36}:CORR:COLL:LOAD
Calibrate	1-Port Cal	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	1-Port Cal	Cancel		:SENS{1-36}:CORR:COLL:CLE
Calibrate	2-Port Cal	Select Ports		:SENS{1-36}:CORR:COLL:METH:SOLT2 {1-
Calibrate	2-Port Cal	Reflection	Port x Open	:SENS{1-36}:CORR:COLL:OPEN {1-4}
Calibrate	2-Port Cal	Reflection	Port x Short	:SENS{1-36}:CORR:COLL:SHOR {1-4}
Calibrate	2-Port Cal	Reflection	Port x Load	:SENS{1-36}:CORR:COLL:LOAD {1-4}
Calibrate	2-Port Cal	Transmission	Port x-y Thru	:SENS{1-36}:CORR:COLL:THRU {1-4},{1-4}
Calibrate	2-Port Cal	Isolation (Optional)	Port x-y Isol	:SENS{1-36}:CORR:COLL:ISOL {1-4},{1-4}
Calibrate	2-Port Cal	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	2-Port Cal	Overwrite		:SENS{1-36}:CORR:COLL:PART:SAVE
Calibrate	2-Port Cal	Cancel		:SENS{1-36}:CORR:COLL:CLE
Calibrate	3-Port Cal	Select Ports	:SENS{1-36}:CORR:COLL:METH:SOLT3 {1-2},{2-3},{3-4}	
Calibrate	3-Port Cal	Reflection	Port x Open	:SENS{1-36}:CORR:COLL:OPEN {1-4}
Calibrate	3-Port Cal	Reflection	Port x Short	:SENS{1-36}:CORR:COLL:SHOR {1-4}
Calibrate	3-Port Cal	Reflection	Port x Load	:SENS{1-36}:CORR:COLL:LOAD {1-4}
Calibrate	3-Port Cal	Transmission	Port x-y Thru	:SENS{1-36}:CORR:COLL:THRU {1-4},{1-4}
Calibrate	3-Port Cal	Isolation (Optional)	Port x-y Isol	:SENS{1-36}:CORR:COLL:ISOL {1-4},{1-4}
Calibrate	3-Port Cal	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	3-Port Cal	Overwrite		:SENS{1-36}:CORR:COLL:PART:SAVE
Calibrate	3-Port Cal	Cancel		:SENS{1-36}:CORR:COLL:CLE
Calibrate	4-Port Cal	Reflection	Port x Open	:SENS{1-36}:CORR:COLL:OPEN {1-4}
Calibrate	4-Port Cal	Reflection	Port x Short	:SENS{1-36}:CORR:COLL:SHOR {1-4}
Calibrate	4-Port Cal	Reflection	Port x Load	:SENS{1-36}:CORR:COLL:LOAD {1-4}

Calibrate	4-Port Cal	Transmission	Port x-y Thru	:SENS{1-36}:CORR:COLL:THRU {1-3},{2-4}
Calibrate	4-Port Cal	Isolation (Optional)	Port x-y Isol	:SENS{1-36}:CORR:COLL:ISOL {1-3},{2-4}
Calibrate	4-Port Cal	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	4-Port Cal	Overwrite		:SENS{1-36}:CORR:COLL:PART:SAVE
Calibrate	4-Port Cal	Cancel		:SENS{1-36}:CORR:COLL:CLE
Calibrate	2-Port TRL Cal	Select Ports		:SENS{1-36}:CORR:COLL:METH:TRL2 {1-3}
Calibrate	2-Port TRL Cal	Thru/Line	x-y Thru/Line	:SENS{1-36}:CORR:COLL:TRLT {1-4},{1-4}
Calibrate	2-Port TRL Cal	Reflect	Portx Reflect	:SENS{1-36}:CORR:COLL:TRLR {1-4}
Calibrate	2-Port TRL Cal	Line/Match	x-y Line/Match	:SENS{1-36}:CORR:COLL:TRLL {1-4},{1-4}
Calibrate	2-Port TRL Cal	Line/Match	x-y Fwd (Syx)	:SENS{1-36}:CORR:COLL:TRLL {1-4},{1-4}
Calibrate	2-Port TRL Cal	Line/Match	x-y Rvs(Sxy)	:SENS{1-36}:CORR:COLL:TRLL {1-4},{1-4}
Calibrate	2-Port TRL Cal	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	2-Port TRL Cal	Cancel		:SENS{1-36}:CORR:COLL:CLE
Calibrate	3-Port TRL Cal	Select Ports	:SENS{1-36}:CORR:COLL:METH:TRL3 {1-2},{2-3},{3-4}	
Calibrate	3-Port TRL Cal	Thru/Line	x-y Thru/Line	:SENS{1-36}:CORR:COLL:TRLT {1-4},{1-4}
Calibrate	3-Port TRL Cal	Reflect	Portx Reflect	:SENS{1-36}:CORR:COLL:TRLR {1-4}
Calibrate	3-Port TRL Cal	Line/Match	x-y Line/Match	:SENS{1-36}:CORR:COLL:TRLL {1-4},{1-4}
Calibrate	3-Port TRL Cal	Line/Match	x-y Fwd (Syx)	:SENS{1-36}:CORR:COLL:TRLL {1-4},{1-4}
Calibrate	3-Port TRL Cal	Line/Match	x-y Rvs(Sxy)	:SENS{1-36}:CORR:COLL:TRLL {1-4},{1-4}
Calibrate	3-Port TRL Cal	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	3-Port TRL Cal	Cancel		:SENS{1-36}:CORR:COLL:CLE
Calibrate	4-Port TRL Cal	Thru/Line	x-y Thru/Line	:SENS{1-36}:CORR:COLL:TRLT {1-4},{1-4}
Calibrate	4-Port TRL Cal	Reflect	Portx Reflect	:SENS{1-36}:CORR:COLL:TRLR {1-4}
Calibrate	4-Port TRL Cal	Line/Match	x-y Line/Match	:SENS{1-36}:CORR:COLL:TRLL {1-4},{1-4}
Calibrate	4-Port TRL Cal	Line/Match	x-y Fwd (Syx)	:SENS{1-36}:CORR:COLL:TRLL {1-4},{1-4}
Calibrate	4-Port TRL Cal	Line/Match	x-y Rvs(Sxy)	:SENS{1-36}:CORR:COLL:TRLL {1-4},{1-4}

Calibrate	4-Port TRL Cal	Done		:SENS{1-36}:CORR:COLL:SAVE
Calibrate	4-Port TRL Cal	Cancel		:SENS{1-36}:CORR:COLL:CLE
Calibrate	Adapter Removal	Select Port		
Calibrate	Adapter Removal	Cal Kit		
Calibrate	Adapter Removal	Open		
Calibrate	Adapter Removal	Short		
Calibrate	Adapter Removal	Load		
Calibrate	Adapter Removal	Adapter Length		
Calibrate	Adapter Removal	Done		
Calibrate	Adapter Removal	Cancel		
ECal	1-Port ECal	Port 1		:SENS{1-36}:CORR:COLL:ECAL:SOLT1 {1-4
ECal	2-Port ECal	Port 1-2		:SENS{1-36}:CORR:COLL:ECAL:SOLT2 {1-3
ECal	3-Port ECal	Port 1-2-3	:SENS{1-36}:CORR:COLL:ECAL:SOLT3 {1-2},{2-3},{3-4}	
ECal	4-Port ECal			:SENS{1-36}:CORR:COLL:ECAL:SOLT4 1,2,
ECal	Thru ECal			:SENS{1-36}:CORR:COLL:ECAL:THRU {1-4
ECal	Enhanced Response			:SENS{1-36}:CORR:COLL:ECAL:ERES {1-4}
ECal	Isolation			:SENS{1-36}:CORR:COLL:ECAL:ISOL
ECal	Unknown Thru			:SENS{1-36}:CORR:COLL:ECAL:UTHRU
ECal	Characterization		:SENS{1-36}:CORR:COLL:ECAL:UCH {CHAR0-CHAR5}	
ECal	Characterization Info			None
ECal	Confidence Check			:SENS{1-36}:CORR:COLL:ECAL:CCH
ECal	Orientation	Auto		:SENS{1-36}:CORR:COLL:ECAL:ORI
ECal	Orientation	Port 1		:SENS{1-36}:CORR:COLL:ECAL:PATH 1,{1-
ECal	Orientation	Port 2		:SENS{1-36}:CORR:COLL:ECAL:PATH 2,{1-
ECal	Orientation	Port 3		:SENS{1-36}:CORR:COLL:ECAL:PATH 3,{1-
ECal	Orientation	Port 4		:SENS{1-36}:CORR:COLL:ECAL:PATH 4,{1-
Clear	OK			:SENS{1-36}:CORR:CLE

Property				:SENS{1-36}:CORR:PROP
Cal Kit	{Calkit Name}			:SENS{1-36}:CORR:COLL:CKIT
Modify Cal Kit				CALC{1-36}:SEL:FORM SCOM
Modify Cal Kit	Define STDs	n. (Std Name)	Label	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	STD Type	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	C0	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	C1	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	C2	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	C3	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	L0	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	L1	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	L2	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	L3	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	Offset Delay	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	Offset Z0	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	Offset Loss	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	Arb. Impedance	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	Min. Frequency	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	Max. Frequency	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Define STDs	n. (Std Name)	Media	:SENS{1-36}:CORR:COLL:CKIT:STAN{1-21}
Modify Cal Kit	Specify CLSs	Sub Class x		:SENS{1-36}:CORR:COLL:CKIT:ORD {1-8}
Modify Cal Kit	Specify CLSs	Open		:SENS{1-36}:CORR:COLL:CKIT:ORD:OPEN
Modify Cal Kit	Specify CLSs	Short		:SENS{1-36}:CORR:COLL:CKIT:ORD:SHOR
Modify Cal Kit	Specify CLSs	Load		:SENS{1-36}:CORR:COLL:CKIT:ORD:LOAD
Modify Cal Kit	Specify CLSs	Thru		:SENS{1-36}:CORR:COLL:CKIT:ORD:THRU
Modify Cal Kit	Specify CLSs	TRL Thru		:SENS{1-36}:CORR:COLL:CKIT:ORD:TRLT

Modify Cal Kit	Specify CLSs	TRL Reflect		:SENS{1-36}:CORR:COLL:CKIT:ORD:TRLR
Modify Cal Kit	Specify CLSs	TRL Line/Match		:SENS{1-36}:CORR:COLL:CKIT:ORD:TRLL
Modify Cal Kit	TRL Option	Impedance		:SENS{1-36}:CORR:COLL:CKIT:TRL:IMP
Modify Cal Kit	TRL Option	Reference Plane		:SENS{1-36}:CORR:COLL:CKIT:TRL:RPL
Modify Cal Kit	Label Kit			:SENS{1-36}:CORR:COLL:CKIT:LAB
Modify Cal Kit	Export Cal Kit...			:MMEM:STOR:CKIT{1-20}
Modify Cal Kit	Import Cal Kit...			:MMEM:LOAD:CKIT{1-20}
Modify Cal Kit	Restore Cal Kit			:SENS{1-36}:CORR:COLL:CKIT:RES
Port Extensions	Auto Port Extension	Select Ports		:SENS{1-36}:CORR:EXT:AUTO:PORT {1-4}
Port Extensions	Auto Port Extension	Meaure {Std}		:SENS{1-36}:CORR:EXT:AUTO:MEAS
Port Extensions	Auto Port Extension	Method	Current Span	:SENS{1-36}:CORR:EXT:AUTO:CONF CSPN
Port Extensions	Auto Port Extension	Method	*Active Marker*	:SENS{1-36}:CORR:EXT:AUTO:CONF AMKF
Port Extensions	Auto Port Extension	Method	User Span	:SENS{1-36}:CORR:EXT:AUTO:CONF USPN
Port Extensions	Auto Port Extension	Method	User Span Start	:SENS{1-36}:CORR:EXT:AUTO:STAR
Port Extensions	Auto Port Extension	Method	User Span Stop	:SENS{1-36}:CORR:EXT:AUTO:STOP
Port Extensions	Auto Port Extension	Adjust Mismatch		:SENS{1-36}:CORR:EXT:AUTO:DCOF
Port Extensions	Auto Port Extension	Include		:SENS{1-36}:CORR:EXT:AUTO:LOSS
Port Extensions	Extensions			:SENS{1-36}:CORR:EXT
Port Extensions				:SENS{1-36}:CORR:EXT:PORT{1-4}
Port Extensions	Loss	Select Ports		None
Port Extensions	Loss	Loss1		:SENS{1-36}:CORR:EXT:PORT{1-4}:INCL1
Port Extensions	Loss	Loss1		:SENS{1-36}:CORR:EXT:PORT{1-4}:LOSS1
Port Extensions	Loss	Freq1		:SENS{1-36}:CORR:EXT:PORT{1-4}:FREQ1
Port Extensions	Loss	Loss2		:SENS{1-36}:CORR:EXT:PORT{1-4}:INCL2
Port Extensions	Loss	Loss2		:SENS{1-36}:CORR:EXT:PORT{1-4}:LOSS2
Port Extensions	Loss	Freq2		:SENS{1-36}:CORR:EXT:PORT{1-4}:FREQ2

Port Extensions	Loss	Loss at DC		:SENS{1-36}:CORR:EXT:PORT{1-4}:LDC
Velocity Factor				:SENS{1-36}:CORR:RVEL:COAX
Set Z0				
Power Calibration	Select Port			None
Power Calibration	Correction			:SOUR{1-36}:POW:PORT{1-4}:CORR
Power Calibration	Take Cal Sweep			:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL
Power Calibration	Abort			None
Power Calibration	Use Sensor			None
Power Calibration	Num of Readings			:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:
Power Calibration	Tolerance			:SOUR:POW:PORT {1-4}:CORR:COLL:NTOI
Power Calibration	Loss Compen	Compensation	:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:TABL:LOSS	
Power Calibration	Loss Compen	Delete	:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:TABL:LOSS:DA	
Power Calibration	Loss Compen	Add	:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:TABL:LOSS:DA	
Power Calibration	Loss Compen	Clear Loss Table	OK	None
Power Calibration	Loss Compen	Export to CSV File		:MMEM:STOR:PLOS{1-4}
Power Calibration	Loss Compen	Import from CSV File		:MMEM:LOAD:PLOS{1-4}
Power Calibration	Sensor A Settings	Ref Cal Factor		:SOUR:POW:PORT:CORR:COLL:ASEN:RCF
Power Calibration	Sensor A Settings	Delete		:SOUR:POW:PORT:CORR:COLL:TABL:ASE
Power Calibration	Sensor A Settings	Add		:SOUR:POW:PORT:CORR:COLL:TABL:ASE
Power Calibration	Sensor A Settings	Clear Loss Table	OK	None
Power Calibration	Sensor A Settings	Export to CSV File		:MMEM:STOR:ASCF
Power Calibration	Sensor A Settings	Import from CSV File		:MMEM:LOAD:ASCF
Power Calibration	Sensor B Settings	Ref Cal Factor		:SOUR:POW:PORT:CORR:COLL:BSEN:RCF
Power Calibration	Sensor B Settings	Delete		:SOUR:POW:PORT:CORR:COLL:TABL:BSE
Power Calibration	Sensor B Settings	Add		:SOUR:POW:PORT:CORR:COLL:TABL:BSE

Power Calibration	Sensor B Settings	Clear Loss Table	OK	None
Power Calibration	Sensor B Settings	Export to CSV File		:MMEM:STOR:BSCF
Power Calibration	Sensor B Settings	Import from CSV File		:MMEM:LOAD:BSCF
Receiver Calibration	Select Port			None
Receiver Calibration	Correction			:SENS{1-36}:CORR:REC{1-4}
Receiver Calibration	Source Port			None
Receiver Calibration	Take Cal Sweep			:SENS{1-36}:CORR:REC{1-4}:COLL:ACQ
Mixer/Converter Calibration	Scalar Cal (Manual)	Select Ports		:SENS{1-36}:CORR:OFFS:COLL:METH:SOL
Mixer/Converter Calibration	Scalar Cal (Manual)	Select Ports		:SENS{1-36}:CORR:OFFS:COLL:METH:SMI
Mixer/Converter Calibration	Scalar Cal (Manual)	Reflection		:SENS{1-36}:CORR:OFFS:COLL:OPEN :SENS{1-36}:CORR:OFFS:COLL:SHOR :SENS{1-36}:CORR:OFFS:COLL:LOAD
Mixer/Converter Calibration	Scalar Cal (ECal)	Select Ports		None
Mixer/Converter Calibration	Scalar Cal (ECal)	Power Meter	Use Sensor	:SENS{1-36}:CORR:OFFS:COLL:PMET
Mixer/Converter Calibration	Scalar Cal (ECal)	ECal & Done		:SENS{1-36}:CORR:OFFS:COLL:ECAL:SMI :SENS{1-36}:CORR:OFFS:COLL:ECAL:SOL
Mixer/Converter Calibration	Scalar Cal (ECal)	Cancel	OK	:SENS{1-36}:CORR:OFFS:COLL:CLE
Mixer/Converter Calibration	Clear	OK		:SENS{1-36}:CORR:OFFS:CLE
Cal Trig Source	Internal			:SENS{1-36}:CORR:TRIG:SOUR:FREE ON
Cal Trig Source	System			:SENS{1-36}:CORR:TRIG:SOUR:FREE OFF

Display		COM
Allocate Channels		SCPI.DISPlay.SPLit
Number of Traces		SCPI.CALCulate(Ch).PARameter.COUNt
Allocate Traces		SCPI.DISPlay.WINDow(Ch).SPLit
Display	Data	SCPI.DISPlay.WINDow(Ch).TRACe(Tr).STATe
Display	Mem	SCPI.DISPlay.WINDow(Ch).TRACe(Tr).MEMory.STATe
Display	Data & Mem	SCPI.DISPlay.WINDow(Ch).TRACe(Tr).MEMory.STATe SCPI.DISPlay.WINDow(Ch).TRACe
Display	OFF	SCPI.DISPlay.WINDow(Ch).TRACe(Tr).MEMory.STATe SCPI.DISPlay.WINDow(Ch).TRACe(Tr).STATe
Data -> Mem		SCPI.CALCulate(Ch).SELected.MATH.MEMorize
Data Math	OFF	SCPI.CALCulate(Ch).SELected.MATH.FUNCtion
Data Math	Data / Mem	SCPI.CALCulate(Ch).SELected.MATH.FUNCtion
Data Math	Data * Mem	SCPI.CALCulate(Ch).SELected.MATH.FUNCtion
Data Math	Data - Mem	SCPI.CALCulate(Ch).SELected.MATH.FUNCtion
Data Math	Data + Mem	SCPI.CALCulate(Ch).SELected.MATH.FUNCtion
Edit Title Label		SCPI.DISPlay.WINDow(Ch).TITLe.DATA
Title Label		SCPI.DISPlay.WINDow(Ch).TITLe.STATe
Graticule Label		SCPI.DISPlay.WINDow(Ch).LABel
Invert Color		SCPI.DISPlay.IMAGe
Frequency		SCPI.DISPlay.ANNotation.FREQuency.STATe
Update		SCPI.DISPlay.ENABle

Format		COM	SCPI
Log Mag		SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM MLOG
Phase		SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM PHAS
Group Delay		SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM GDEL
Smith	Lin / Phase	SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM SLIN
Smith	Log / Phase	SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM SLOG
Smith	Real / Imag	SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM SCOM
Smith	R + jX	SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM SMIT
Smith	G + jB	SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM SMGB
Ploar	Lin / Phase	SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM PLIN
Ploar	Log / Phase	SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM PLOG
Ploar	Real / Imag	SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM POL
Lin Mag		SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM MLIN
SWR		SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM SWR
Real		SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM REAL
Imaginary		SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM IMAG
Expand Phase		SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM UPH
Positive Phase		SCPI.CALCulate(Ch).SELected.FORMat	:CALC{1-36}:SEL:FORM PPH

SMacro Setup		COM	SCPI
VBA Editor		None	None
New Project		None	None
Load Project		None	:MMEM:LOAD:PROG
Load & Run	(program name)	None	None
Save Project		None	:MMEM:STOR:PROG
Close Editor		None	None
Select Macro	(macro name)	None	:PROG:NAME :PROG:STAT
Stop		None	:PROG:STAT
Continue		None	:PROG:STAT
Echo Window		SCPI.DISPlay.TABLe.STATe	:DISP:TABL :DISP:TABL:TYPE
Clear Echo		SCPI.DISPlay.ECHO.CLEar	:DISP:ECHO:CLE
User Menu		None	None
Preset User Menu		None	None

Marker Function		COM	SCPI
Marker -> Start		SCPI.CALCulate(Ch).SELected.MARKer(Mk).SET	:CALC{1-36}:MARK{1-10}:SET STAR
Marker -> Stop		SCPI.CALCulate(Ch).SELected.MARKer(Mk).SET	:CALC{1-36}:MARK{1-10}:SET STOP
Marker -> Center		SCPI.CALCulate(Ch).SELected.MARKer(Mk).SET	:CALC{1-36}:MARK{1-10}:SET CENT
Marker -> Reference		SCPI.CALCulate(Ch).SELected.MARKer(Mk).SET	:CALC{1-36}:MARK{1-10}:SET RLEV
Marker -> Delay		SCPI.CALCulate(Ch).SELected.MARKer(Mk).SET	:CALC{1-36}:MARK{1-10}:SET DEL
Discrete		SCPI.CALCulate(Ch).SELected.MARKer(Mk).DISCrete	:CALC{1-36}:MARK{1-10}:DISC
Couple		SCPI.CALCulate(Ch).SELected.MARKer.COUPle	:CALC{1-36}:MARK{1-10}:COUP
Marker Table		SCPI.DISPlay.TABLe.STATe SCPI.DISPlay.TABLe.TYPE	:DISP:TABL :DISP:TABL:TYPE MARK
Statistics		SCPI.CALCulate(Ch).SELected.MSTatistics.STATe	:CALC{1-36}:MST
Annotation Options	Marker Info X Pos	SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.X	:DISP{1-36}:WIND:TRAC{1-36}:ANN:MARK:POS:X
Annotation Options	Marker Info Y Pos	SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.Y	:DISP{1-36}:WIND:TRAC{1-36}:ANN:MARK:POS:Y
Annotation Options	Align	SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.ALIGn. STATe	:DISP:WIND{1-36}:ANN:MARK:ALIG
Annotation Options	Active Only	SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.SINGle. STATe	:DISP:WIND{1-36}:ANN:MARK:SING

Marker Search		COM
Max		SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute SCPI.CALCulate(Ch).SELected
Min		SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
Peak	Search Peak	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
Peak	Search Left	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
Peak	Search Right	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
Peak	Peak Excursion	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PEXCursion
Peak	Peak Polarity	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PPOLarity
Target	Search Target	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
Target	Search Left	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
Target	Search Right	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE
Target	Target Value	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TARGet
Target	Target Transition	SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TTRansition
Tracking		SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TRACking
Search Range	Search Range	SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STATe
Search Range	Start	SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STARt
Search Range	Stop	SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STOP
Search Range	Couple	SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.COUPle
Bandwidth		SCPI.CALCulate(Ch).SELected.MARKer.BWIDth.STATe
Bandwidth Value		SCPI.CALCulate(Ch).SELected.MARKer(Mk).BWIDth.THReshold

Marker		COM
Marker 1 to 4		SCPI.CALCulate(Ch).SELected.MARKer(Mk).STATe SCPI.CALCulate(Ch).SELected.M SCPI.CALCulate(Ch).SELected.MARKer(Mk).X
More Markers	Marker 5 to 9	
Ref Marker		SCPI.CALCulate(Ch).SELected.MARKer(10).STATe SCPI.CALCulate(Ch).SELected.MARKer(10).ACTivate SCPI.CALCulate(Ch).SELected.MARKer(10).X
Clear Marker Menu	All OFF	None
Clear Marker Menu	Marker 1 to 9	SCPI.CALCulate(Ch).SELected.MARKer(Mk).STATe
Clear Marker Menu	Ref Marker	SCPI.CALCulate(Ch).SELected.MARKer(10).STATe
Marker -> Ref Marker		None
Ref Marker Mode		SCPI.CALCulate(Ch).SELected.MARKer.REFerence. STATe

COM| .SCPI

Meas		COM
{S11-S44}		SCPI.CALCulate(Ch).PARameter(Tr).DEFine
Absolute	{A(1)-R4(4)}	SCPI.CALCulate(Ch).PARameter(Tr).DEFine SCPI.CALCulate(Ch).PAR
{Sss11-Scc22}		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).SBALanced.DE
Imbalance		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).SBALanced.DE
Sds21/Scs21		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).SBALanced.DE
Ssd12/Ssc12		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).SBALanced.DE
{Sdd11-Scc22}		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).BBALanced.DE
{Imbalance1-Imbalance2}		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).BBALanced.DE
Sdd21/Scc21		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).BBALanced.DE
{Sss11-Scc33}		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).SSBalanced.DE
{Imbalance1-Imbalance4}		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).SSBalanced.DE
Sds31/Scs31		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).SSBalanced.DE
Sds32/Scs32		SCPI.CALCulate(Ch).FSIMulator.BALun.PARameter(Tr).SSBalanced.DE
Aux Input {1-2}	Range	SCPI.SENSe(Ch).VOLTage(Auxpt).DC.RANGe.UPPer
Aux Input {1-2}	Sweep Port	SCPI.CALCulate(Ch).PARameter(Tr).SPORt

COM | SCPI

Meas		SCPI
{S11-S44}		:CALC{1-36}:PAR{1-36}:DEF {S11-S44}
Absolute	{A(1)-R4(4)}	:CALC{1-36}:PAR{1-36}:DEF {A-R4} :CALC{1-36}:PAR{1-36}:SPOR {1-4
{Sss11-Scc22}		:CALC{1-36}:FSIM:BAL:PAR{1-36}:SBAL {SSS11-SCC22}
Imbalance		:CALC{1-36}:FSIM:BAL:PAR{1-36}:SBAL IMB
Sds21/Scs21		:CALC{1-36}:FSIM:BAL:PAR{1-36}:SBAL CMRR1
Ssd12/Ssc12		:CALC{1-36}:FSIM:BAL:PAR{1-36}:SBAL CMRR2
{Sdd11-Scc22}		:CALC{1-36}:FSIM:BAL:PAR{1-36}:BBAL {SDD11-SCC22}
{Imbalance1-Imbalance2}		:CALC{1-36}:FSIM:BAL:PAR{1-36}:BBAL {IMB1-IMB2}
Sdd21/Scc21		:CALC{1-36}:FSIM:BAL:PAR{1-36}:BBAL CMRR1
{Sss11-Scc33}		:CALC{1-36}:FSIM:BAL:PAR{1-36}:SSB {SSS11-SCC33}
{Imbalance1-Imbalance4}		:CALC{1-36}:FSIM:BAL:PAR{1-36}:SSB {IMB1-IMB4}
Sds31/Scs31		:CALC{1-36}:FSIM:BAL:PAR{1-36}:SSB CMRR1
Sds32/Scs32		:CALC{1-36}:FSIM:BAL:PAR{1-36}:SSB CMRR2
Aux Input {1-2}	Range	:SENS{1-36}:VOLT{1-2}:DC:RANGe:UPP
Aux Input {1-2}	Sweep Port	:CALC{1-36}:PAR{1-36}:SPOR {1-4}

Preset		COM	SCPI
Preset Menu	OK	SCPI.SYSTem.PRESet	:SYST:PRES

Save/Recall			COM	SCPI
Save State	{State01-State08}		SCPI.MMEMory.STORe.STATe	:MMEM:STOR {"State01.sta"-"State
Save State	Autorec		SCPI.MMEMory.STORe.STATe	:MMEM:STOR "Autorec.sta"
Save State	UserPres		SCPI.MMEMory.STORe.STATe	:MMEM:STOR "UserPres.sta"
Save State	File Dialog...		SCPI.MMEMory.STORe.STATe	:MMEM:STOR
Recall State	{State01-State08}		SCPI.MMEMory.LOAD.STATe	:MMEM:LOAD {"State01.sta"-"State
Recall State	Autorec		SCPI.MMEMory.LOAD.STATe	:MMEM:LOAD "Autorec.sta"
Recall State	UserPres		SCPI.MMEMory.LOAD.STATe	MMEM:LOAD "UserPres.sta"
Recall State	File Dialog...		SCPI.MMEMory.LOAD.STATe	:MMEM:LOAD
Recall by File Name			SCPI.MMEMory.LOAD.STATe	:MMEM:LOAD
Save Channel	{State A-State D}		SCPI.MMEMory.STORe.CHANnel.STATe	:MMEM:STOR:CHAN {A-D}
Save Channel	Clear States		SCPI.MMEMory.STORe.CHANnel.CLEar	:MMEM:STOR:CHAN:CLE
Recall Channel	{State A-State D}		SCPI.MMEMory.LOAD.CHANnel.STATe	:MMEM:LOAD:CHAN {A-D}
Save Type	State Only		SCPI.MMEMory.STORe.STYPe	:MMEM:STOR:STYP STAT
Save Type	State & Cal		SCPI.MMEMory.STORe.STYPe	:MMEM:STOR:STYP CST
Save Type	State & Trace		SCPI.MMEMory.STORe.STYPe	:MMEM:STOR:STYP DST
Save Type	All		SCPI.MMEMory.STORe.STYPe	:MMEM:STOR:STYP CDST
Channel/Trace			SCPI.MMEMory.STORe.STYPe	:MMEM:STOR:STYP SALL
Save Trace Data			SCPI.MMEMory.STORe.FDATa	:MMEM:STOR:FDAT
Explorer...			SCPI.MMEMory.MDIRectory SCPI.MMEMory.COPY SCPI.MMEMory.DELete	:MMEM:MDIR :MMEM:COPY :MMEM:DEL
Save SnP	SnP Format	Auto	SCPI.MMEMory.STORe.SNP.FORMat	:MMEM:STOR:SNP:FORM AUTO
Save SnP	SnP Format	LinMag/Angle	SCPI.MMEMory.STORe.SNP.FORMat	:MMEM:STOR:SNP:FORM MA

Save SnP	SnP Format	LogMag/ Angle	SCPI.MMEMory.STORe.SNP.FORMat	:MMEM:STOR:SNP:FORM DB
Save SnP	SnP Format	Real/ Imaginary	SCPI.MMEMory.STORe.SNP.FORMat	:MMEM:STOR:SNP:FORM RI
Save SnP	s1p	{1-4}	SCPI.MMEMory.STORe.SNP.TYPE.S1P SCPI.MMEMory.STORe.SNP.DATA	:MMEM:STOR:SNP:TYPE:S1P {1- :MMEM:STOR:SNP
Save SnP	s2p	{1-3}-{2-4}	SCPI.MMEMory.STORe.SNP.TYPE.S2P SCPI.MMEMory.STORe.SNP.DATA	:MMEM:STOR:SNP:TYPE:S2P {1- :MMEM:STOR:SNP
Save SnP	s3p	{1-2}-{2-3}-{3-4}	SCPI.MMEMory.STORe.SNP.TYPE.S3P SCPI.MMEMory.STORe.SNP.DATA	:MMEM:STOR:SNP:TYPE:S3P {1- :MMEM:STOR:SNP
SnP	s4p	1-2-3-4	SCPI.MMEMory.STORe.SNP.TYPE.S4P SCPI.MMEMory.STORe.SNP.DATA	:MMEM:STOR:SNP:TYPE:S4P 1,2 :MMEM:STOR:SNP

Scale		COM	SCPI
Auto Scale		SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.AUTO	:DISP:WIND{1-36}:TRA
Auto Scale All		None	None
Divisions		SCPI.DISPlay.WINDow(Ch).Y.SCALe.DIVisions	:DISP:WIND{1-36}:Y:DI
Scale/Div		SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.PDIVision	:DISP:WIND{1-36}:TRA
Reference Position		SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RPOSition	:DISP:WIND{1-36}:TRA
Reference Value		SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RLEVel	:DISP:WIND{1-36}:TRA
Marker -> Reference		SCPI.CALCulate(Ch).SELected.MARKer(Mk).SET	:CALC{1-36}:MARK{1-1
Electrical Delay	Electrical Delay	SCPI.CALCulate(Ch).SELected.CORRection.EDELay.TIME	:CALC{1-36}:CORR:ED
Electrical Delay	Media	SCPI.CALCulate(Ch).SELected.CORRection.EDELay.MEDium	:CALC{1-36}:CORR:ED
Electrical Delay	Cutoff Frequency	SCPI.CALCulate(Ch).SELected.CORRection.EDELay.WGCutoff	:CALC{1-36}:CORR:ED
Phase Offset		SCPI.CALCulate(Ch).SELected.CORRection.OFFSet.PHASe	:CALC{1-36}:CORR:OF

		COM	SCPI
Start	Start	SCPI.SENSe(Ch).FREQuency.STARt	:SENS{1-36}:FREQ:STAR
Stop	Stop	SCPI.SENSe(Ch).FREQuency.STOP	:SENS{1-36}:FREQ:STOP
Center	Center	SCPI.SENSe(Ch).FREQuency.CENTer	:SENS{1-36}:FREQ:CENT
Span	Span	SCPI.SENSe(Ch).FREQuency.SPAN	:SENS{1-36}:FREQ:SPAN

COM| SCPI

Sweep Setup			COM
Power	Power		SCPI.SOURce(Ch).POWer.LEVel.IMMediate.AMPLitude
Power	Port Couple		SCPI.SOURce(Ch).POWer.PORT.COUPle
Power	Port Power	Port {1-4} Power	SCPI.SOURce(Ch).POWer.PORT(Pt).LEVel.IMMediate.AMPLitude
Power	Slope [xx dB/GHz]		SCPI.SOURce(Ch).POWer.LEVel.SLOPe.DATA
Power	Slope [ON/OFF]		SCPI.SOURce(Ch).POWer.LEVel.SLOPe.STATe
Power	CW Freq		SCPI.SENSe(Ch).FREQuency.CW
Power	RF Out		SCPI.OUTPut.STATe
Sweep Time			SCPI.SENSe(Ch).SWEep.TIME.DATA SCPI.SENSe(Ch).SWEep.T
Sweep Delay			SCPI.SENSe(Ch).SWEep.DELay
Sweep Mode	Stepped		SCPI.SENSe(Ch).SWEep.GENeration
Points			SCPI.SENSe(Ch).SWEep.POINts
Sweep Type			SCPI.SENSe(Ch).SWEep.TYPE
Edit Segment Table			SCPI.SENSe(Ch).SEGMent.DATA
Edit Segment Table	Clear Segment Table	OK	None
Edit Segment Table	Export to CSV File		SCPI.MMEMory.STORe.SEGMent
Edit Segment Table	Import from CSV File		SCPI.MMEMory.LOAD.SEGMent
Segment Display			SCPI.DISPlay.WINDow(Ch).X.SPACing
Frequency Offset	Frequency Offset		SCPI.SENSe(Ch).OFFSet.STATe
Frequency Offset	Port {1-4}	Multiplier	SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.MULTiplier
Frequency Offset	Port {1-4}	Divisor	SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.DIVisor
Frequency Offset	Port {1-4}	Offset	SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.OFFSet
Frequency Offset	Port {1-4}	Start	SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STARt
Frequency Offset	Port {1-4}	Stop	SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STOP

Frequency Offset	External Source	LO Frequency	SCPI.SENSe(Ch).OFFSet.LOCal.STATe
Frequency Offset	External Source	Multiplier	SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.MULTiplier
Frequency Offset	External Source	Divisor	SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.DIVisor
Frequency Offset	External Source	Offset	SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.OFFSet
Frequency Offset	External Source	Start	SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STARt
Frequency Offset	External Source	Stop	SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STOP
Frequency Offset	External Source	Control	SCPI.SENSe(Ch).OFFSet.LOCal.CONTrol.STATe
Frequency Offset	External Source	Power	SCPI.SENSe(Ch).OFFSet.LOCal.POWer.LEVel.IMMediate.AMPLit
Frequency Offset	External Source	Slope [xdB/GHz]	SCPI.SENSe(Ch).OFFSet.LOCal.POWer.LEVel.SLOPe.DATA
Frequency Offset	External Source	Slope [ON/OFF]	SCPI.SENSe(Ch).OFFSet.LOCal.POWer.LEVel.SLOPe.STATe
Frequency Offset	X-Axis		SCPI.CALCulate(Ch).SELected.OFFset.XAXis

COM | SCPI

Sweep Setup			SCPI
Power	Power		:SOUR{1-36}:POW
Power	Port Couple		:SOUR{1-36}:POW:PORT:COUP
Power	Port Power	Port {1-4} Power	:SOUR{1-36}:POW:PORT{1-4}
Power	Slope [xx dB/GHz]		:SOUR{1-36}:POW:SLOP
Power	Slope [ON/OFF]		:SOUR{1-36}:POW:SLOP:STAT
Power	CW Freq		:SENS{1-36}:FREQ
Power	RF Out		:OUTP
Sweep Time			:SENS{1-36}:SWE:TIME :SENS{1-36}:SWE:TIME:AUTO
Sweep Delay			:SENS{1-36}:SWE:DEL
Sweep Mode	Stepped		:SENS{1-36}:SWE:GEN STEP
Points			:SENS{1-36}:SWE:POIN
Sweep Type			:SENS{1-36}:SWE:TYPE
Edit Segment Table			:SENS{1-36}:SEGM:DATA
Edit Segment Table	Clear Segment Table	OK	None
Edit Segment Table	Export to CSV File		:MMEM:STOR:SEGM
Edit Segment Table	Import from CSV File		:MMEM:LOAD:SEGM
Segment Display			:DISP:WIND{1-36}:X:SPAC LIN :DISP:WIND{1-36}:X:SPAC OBAS
Frequency Offset	Frequency Offset		:SENS{1-36}:OFFS
Frequency Offset	Port {1-4}	Multiplier	:SENS{1-36}:OFFS:PORT{1-4}:MULT
Frequency Offset	Port {1-4}	Divisor	:SENS{1-36}:OFFS:PORT{1-4}:DIV
Frequency Offset	Port {1-4}	Offset	:SENS{1-36}:OFFS:PORT{1-4}:OFFS
Frequency Offset	Port {1-4}	Start	:SENS{1-36}:OFFS:PORT{1-4}:STAR
Frequency Offset	Port {1-4}	Stop	:SENS{1-36}:OFFS:PORT{1-4}:STOP
Frequency Offset	External Source	LO Frequency	:SENS{1-36}:OFFS:LOC:STAT

Frequency Offset	External Source	Multiplier	:SENS{1-36}:OFFS:LOC:MULT
Frequency Offset	External Source	Divisor	:SENS{1-36}:OFFS:LOC:DIV
Frequency Offset	External Source	Offset	:SENS{1-36}:OFFS:LOC:OFFS
Frequency Offset	External Source	Start	:SENS{1-36}:OFFS:LOC:STAR
Frequency Offset	External Source	Stop	:SENS{1-36}:OFFS:LOC:STOP
Frequency Offset	External Source	Control	:SENS{1-36}:OFFS:LOC:CONT
Frequency Offset	External Source	Power	:SENS{1-36}:OFFS:LOC:POW
Frequency Offset	External Source	Slope [xdB/GHz]	:SENS{1-36}:OFFS:LOC:POW:SLOP
Frequency Offset	External Source	Slope [ON/OFF]	:SENS{1-36}:OFFS:LOC:POW:SLOP:STAT
Frequency Offset	X-Axis		:CALC{1-36}:OFFS:XAX

COM | SCPI

System				COM
Print				SCPI.HCOPy.IMMediate
Abort Printing				SCPI.HCOPy.ABORt
Printer Setup				None
Invert Image				SCPI.HCOPy.IMAGe
Dump Screen Image				SCPI.MMEMory.STORe.IMAGe
Multiport Test Set Setup				
Test Set {1-2}	Select Test Set			SCPI.SENSe.MULTiplexer(Id).NAME
Test Set {1-2}	Control			SCPI.SENSe(Ch).MULTiplexer(Id).STA
Test Set {1-2}	Property		SCPI.SENSe(Ch).MULTiplexer(Id).DISPlay.STATe	
Test Set {1-2}	Port {1-4}		SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).SELect	
Test Set {1-2}	Control Lines	Line {1-8}	SCPI.SENSe(Ch).MULTiplexer(Id).OUTPut.DATA	
Misc Setup				
Beeper	Beep Complete		SCPI.SYSTem.BEEPer.COMPlete.STATe	
Beeper	Test Beep Complete		SCPI.SYSTem.BEEPer.COMPlete.IMMediate	
Beeper	Beep Warning		SCPI.SYSTem.BEEPer.WARNing.STATe	
Beeper	Test Beep Warning		SCPI.SYSTem.BEEPer.WARNing.IMMediate	
GPIB Setup	Talker/Listener Address			None
GPIB Setup	System Controller Configuration			None
GPIB Setup	Power Meater Address		SCPI.SYSTem.COMMunicate.GPIB.PMETer.ADDRess	
GPIB Setup	Signal Generator Address	Address	SCPI.SYSTem.COMMunicate.GPIB.SGENerator.ADDR	
GPIB Setup	Signal Generator Address	Switching Time	SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWEL	
GPIB Setup	Signal Generator Address	Custom Commands	Preset	SCPI.SYSTem.COMMunicate.GPIB.S SCPI.SYSTem.COMMunicate.GPIB.S
GPIB Setup	Signal Generator Address	Custom Commands	Turn RF Out On	SCPI.SYSTem.COMMunicate.GPIB.S SCPI.SYSTem.COMMunicate.GPIB.S

GPIB Setup	Signal Generator Address	Custom Commands	Set Frequency	SCPI.SYSTem.COMMunicate.GPIB.S SCPI.SYSTem.COMMunicate.GPIB.S
GPIB Setup	Signal Generator Address	Custom Commands	Set Power Level	SCPI.SYSTem.COMMunicate.GPIB.S SCPI.SYSTem.COMMunicate.GPIB.S
GPIB Setup	Signal Generator Address	8643A, 8644B, 8664A, 8665A/B		SCPI.SYSTem.COMMunicate.GPIB.S
GPIB Setup	Signal Generator Address	8648A/B/C/D, ESG/PSG Series		SCPI.SYSTem.COMMunicate.GPIB.S
Network Setup	Telnet Server			None
Network Setup	SICL-LAN Server			None
Network Setup	SICL-LAN Address			None
Network Setup	Web Server			None
Network Setup	VNC Server Configuration...			None
Network Setup	Network Identification			None
Network Setup	Network Configuration			None
Network Setup	MAC Address			None
Clock Setup	Set Date and Time			SCPI.SYSTem.DATE SCPI.SYSTem.TIME
Clock Setup	Show Clock			SCPI.DISPlay.CLOCk
Key Lock	Front Panel & Keyboard Lock			SCPI.SYSTem.KLOCk.KBD
Key Lock	Mouse Lock			SCPI.SYSTem.KLOCk.MOUSe
Display Setup	Color Setup	Normal	{Data Trace 1-Data Trace 10}	SCPI.DISPlay.COLor(Dnum).TRACe(T
			{Mem Trace 1-Mem Trace 9}	SCPI.DISPlay.COLor(Dnum).TRACe(T
			Graticule Main	SCPI.DISPlay.COLor(Dnum).GRATicu
			Graticule Sub	SCPI.DISPlay.COLor(Dnum).GRATicu
			Limit Fail	SCPI.DISPlay.COLor(Dnum).LIMit(Lnu
			Limit Line	SCPI.DISPlay.COLor(Dnum).LIMit(Lnu
			Background	SCPI.DISPlay.COLor(Dnum).BACK
			Reset Color	SCPI.DISPlay.COLor(Dnum).RESet

	Magnification			
Color Setup	Normal	{Data Trace 1-Data Trace 10}		
Color Setup	Normal	{Mem Trace 1-Mem Trace 9}		
Color Setup	Normal	Graticule Main		
Color Setup	Normal	Graticule Sub		
Color Setup	Normal	Limit Fail		
Color Setup	Normal	Limit Line		
Color Setup	Normal	Background		
Color Setup	Normal	Reset Color		
Channel/ Trace Setup				None
Preset Setup				None
Control Panel...				None
Backlight				SCPI.SYSTem.BACKlight
Firmware Revision				SCPI.IEEE4882.IDN
Service Menu				
Test Menu				None
System Correction				SCPI.SYSTem.CORRection.STATe
External Test Set				SCPI.SYSTem.TSET.EXTernal
Init Src Ctrl				SCPI.SYSTem.ISPC.STAT
Init Src Port	{1-4}			SCPI.SYSTem.ISPC.PORT
Security Level				SCPI.SYSTem.SECurity.LEVel
Enable Options				None
Restart Firmware				None
Update Firmware				None
Service				

Functions				None
Exit				None

COM | SCPI

System				SCPI
Print				:HCOP
Abort Printing				:HCOP:ABOR
Printer Setup				None
Invert Image				:HCOP:IMAG
Dump Screen Image				:MMEM:STOR:IMAG
Multiport Test Set Setup				
Test Set {1-2}	Select Test Set			:SENS{1-36}:MULT{1-2}:NAME
Test Set {1-2}	Control			:SENS{1-36}:MULT{1-2}:STAT
Test Set {1-2}	Property			:SENS{1-36}:MULT{1-2}:DISP
Test Set {1-2}	Port {1-4}			:SENS{1-36}:MULT{1-2}:PORT{1-4}
Test Set {1-2}	Control Lines	Line {1-8}		:SENS{1-36}:MULT{1-2}:OUTP
Misc Setup				
Beeper	Beep Complete			:SYST:BEEP:COMP:STAT
Beeper	Test Beep Complete			:SYST:BEEP:COMP:IMM
Beeper	Beep Warning			:SYST:BEEP:WARN:STAT
Beeper	Test Beep Warning			:SYST:BEEP:WARN:IMM
GPIB Setup	Talker/Listener Address			None
GPIB Setup	System Controller Configuration			None
GPIB Setup	Power Meater Address			:SYST:COMM:GPIB:PMET:ADDR
GPIB Setup	Signal Generator Address	Address		:SYST:COMM:GPIB:SGEN:ADDR
GPIB Setup	Signal Generator Address	Switching Time		:SYST:COMM:GPIB:SGEN:DWEL

GPIB Setup	Signal Generator Address	Custom Commands	Preset	:SYST:COMM:GPIB:SGEN:TYPE 1 :SYST:COMM:GPIB:SGEN:CCOM:PF
GPIB Setup	Signal Generator Address	Custom Commands	Turn RF Out On	:SYST:COMM:GPIB:SGEN:TYPE 1 :SYST:COMM:GPIB:SGEN:CCOM:RF
GPIB Setup	Signal Generator Address	Custom Commands	Set Frequency	:SYST:COMM:GPIB:SGEN:TYPE 1 :SYST:COMM:GPIB:SGEN:CCOM:FF
GPIB Setup	Signal Generator Address	Custom Commands	Set Power Level	:SYST:COMM:GPIB:SGEN:TYPE 1 :SYST:COMM:GPIB:SGEN:CCOM:PC
GPIB Setup	Signal Generator Address	8643A, 8644B, 8664A, 8665A/B		:SYST:COMM:GPIB:SGEN:TYPE 2
GPIB Setup	Signal Generator Address	8648A/B/C/D, ESG/PSG Series		:SYST:COMM:GPIB:SGEN:TYPE 3
Network Setup	Telnet Server			None
Network Setup	SICL-LAN Server			None
Network Setup	SICL-LAN Address			None
Network Setup	Web Server			None
Network Setup	VNC Server Configuration...			None
Network Setup	Network Identification			None
Network Setup	Network Configuration			None
Network Setup	MAC Address			None
Clock Setup	Set Date and Time			:SYST:DATE :SYST:TIME
Clock Setup	Show Clock			:DISP:CLOC
Key Lock	Front Panel & Keyboard Lock			:SYST:KLOC:KBD
Key Lock	Mouse Lock			:SYST:KLOC:MOUS
Display Setup	Color Setup	Normal	{Data Trace 1-Data Trace 10}	:DISP:COL{1-2}:TRAC{1-10}:DATA
			{Mem Trace 1-Mem Trace 9}	:DISP:COL{1-2}:TRAC{1-9}:MEM
			Graticule Main	:DISP:COL{1-2}:GRAT1
			Graticule Sub	:DISP:COL{1-2}:GRAT2
			Limit Fail	:DISP:COL{1-2}:LIM1
			Limit Line	:DISP:COL{1-2}:LIM2

			Background	:DISP:COL{1-2}:BACK
			Reset Color	:DISP:COL{1-2}:RES
	Magnification			
Color Setup	Normal	{Data Trace 1-Data Trace 10}		:DISP:COL{1-2}:TRAC{1-10}:DATA
Color Setup	Normal	{Mem Trace 1-Mem Trace 9}		:DISP:COL{1-2}:TRAC{1-9}:MEM
Color Setup	Normal	Graticule Main		:DISP:COL{1-2}:GRAT1
Color Setup	Normal	Graticule Sub		:DISP:COL{1-2}:GRAT2
Color Setup	Normal	Limit Fail		:DISP:COL{1-2}:LIM1
Color Setup	Normal	Limit Line		:DISP:COL{1-2}:LIM2
Color Setup	Normal	Background		:DISP:COL{1-2}:BACK
Color Setup	Normal	Reset Color		:DISP:COL{1-2}:RES
Channel/Trace Setup				None
Preset Setup				None
Control Panel...				None
Backlight				:SYST:BACK
Firmware Revision				*IDN?
Service Menu				
Test Menu				None
System Correction				:SYST:CORR
External Test Set				:SYST:TSET:EXT
Init Src Ctrl				:SYST:ISPC
Init Src Port	{1-4}			:SYST:ISPC:PORT {1-4}
Security Level				:SYST:SEC:LEV
Enable Options				None
Restart Firmware				None
Update Firmware				None

Service Functions				None
Exit				None

Trigger	COM	SCPI
Hold	SCPI.INITiate(Ch).CONTinuous	:INIT{1-36}:CONT
Single	SCPI.INITiate(Ch).IMMediate	:INIT{1-36}
Continuous	SCPI.INITiate(Ch).CONTinuous	:INIT{1-36}:CONT
Hold All Channels	None	None
Continuous Disp Channels	None	None
Trigger Source	SCPI.TRIGger.SEQuence.SOURce	:TRIG:SOUR
Trigger Event	SCPI.TRIGger.SEQuence.POINt	:TRIG:POIN
Restart	SCPI.ABORt	:ABOR
Trigger	SCPI.TRIGger.SEQuence.IMMediate	:TRIG
Ext Trig Input	SCPI.TRIGger.SEQuence.EXTernal.SLOPe	:TRIG:SEQ:EXT:SLOP
Trigger Delay	SCPI.TRIGger.SEQuence.EXTernal.DELay	:TRIG:SEQ:EXT:DEL
Low Latency	SCPI.TRIGger.SEQuence.EXTernal.LLATency.STATe	:TRIG:SEQ:EXT:LLAT
Ext Trig Output	SCPI.TRIGger.OUTPut.STATe	:TRIG:OUTP:STAT
Polarity	SCPI.TRIGger.OUTPut.POLarity	:TRIG:OUTP:POL
Position	SCPI.TRIGger.OUTPut.POSition	:TRIG:OUTP:POS

ACTIVE CH/TRACE	RESPONSE		STIMULUS		MKR/ANALYSIS	INSTRUMENT STATE	
Channel Next	Channel Max	Trace Max	Start	Stop	Marker	Macro Setup	Macro Run
Channel Prev	Meas	Format	Center	Span	Marker Function	Macro Break	Save/Recall
Trace Next	Scale	Display	Sweep Setup	Trigger	Marker Search	System	Preset
Trace Prev	Ave	Cal			Analysis		

ACTIVE CH/TRACE	RESPONSE		STIMULUS		MKR/ANALYSIS	INSTRUMENT STATE	
Channel Next	Channel Max	Trace Max	Start	Stop	Marker	Macro Setup	Macro Run
Channel Prev	Meas	Format	Center	Span	Marker Function	Macro Break	Save/Recall
Trace Next	Scale	Display	Sweep Setup	Trigger	Marker Search	System	Preset
Trace Prev	Ave	Cal			Analysis		

SCPI.CONTrol.HANDler.A.DATA

SCPI.CONTrol.HANDler.A.DATA = *Value*

/ I/OAA0A78A0LSBA7MSB

	Value
	(Long)
	0255

SCPI.CONTrol.HANDler.A.DATA = 15

SCPI

:CONTrol:HANDler:A[:DATA] <numeric>

```
10 OUTPUT 717;":CONT:HAND:A 15"
20 OUTPUT 717;":CONT:HAND:A:DATA 15"
```

SCPI.CONTrol.HANDler.B.DATA

SCPI.CONTrol.HANDler.B.DATA = *Value*

/ I/OBB0B78B0LSBB7MSB

- INDEX SCPI.CONTrol.HANDler.EXTension.INDex.STATe
  6
- EADY FOR TRIGGER
  “ SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe
  7

	Value
	(Long)
	0255

SCPI.CONTrol.HANDler.B.DATA = 15

SCPI

:CONTrol:HANDler:B[:DATA] <numeric>

```
10 OUTPUT 717;":CONT:HAND:B 15"
20 OUTPUT 717;":CONT:HAND:B:DATA 15"
```

SCPI.CONTrol.HANDler.C.DATA

SCPI.CONTrol.HANDler.C.DATA = *Value* (for output port)

Value = SCPI.CONTrol.HANDler.C.DATA (for input port)

I/O/CCC0C3
I/O/CCC0C3
/4C0LSBC3MSB

	Value
	/
	(Long)
	015

```
SCPI.CONTrol.HANDler.C.MODE = "outp"
SCPI.CONTrol.HANDler.C.DATA = 8

Dim HdlCinp As Long
SCPI.CONTrol.HANDler.C.MODE = "inp"
HdlCinp = SCPI.CONTrol.HANDler.C.DATA
```

SCPI.CONTrol.HANDler.C.MODE

SCPI

:CONTrol:HANDler:C[:DATA] <numeric>

:CONTrol:HANDler:C[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CONT:HAND:C:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:C 15"
```

```
10 OUTPUT 717;":CONT:HAND:C:MODE INP"
20 OUTPUT 717;":CONT:HAND:C?"
30 ENTER 717;A
```

SCPI.CONTrol.HANDler.C.MODE

SCPI.CONTrol.HANDler.C.MODE = *Param*

Param = SCPI.CONTrol.HANDler.C.MODE

/ I/OC/

	Param
	C/
	(String)
	“INPut”C “OUTPut”C
	“INPut”

```
Dim HdlCmode As String
SCPI.CONTrol.HANDler.C.MODE = "OUTP"
HdlCmode = SCPI.CONTrol.HANDler.C.MODE
```

SCPI.CONTrol.HANDler.C.DATA

SCPI

:CONTrol:HANDler:C:MODE {INPut|OUTPut}

:CONTrol:HANDler:C:MODE?

{INP|OUTP}<newline><^END>

```
10 OUTPUT 717;":CONT:HAND:C:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:C:MODE?"
30 ENTER 717;A$
```

SCPI.CONTrol.HANDler.D.DATA

SCPI.CONTrol.HANDler.D.DATA = *Value* (for output port)

Value = SCPI.CONTrol.HANDler.D.DATA (for input port)

I/O/DDD0D3

I/O/DDD0D3

4D0LSBD3MSB

Value

/

(Long)

015

```
SCPI.CONTrol.HANDler.D.MODE = "outp"
SCPI.CONTrol.HANDler.D.DATA = 8

Dim HdlDinp As Long
SCPI.CONTrol.HANDler.D.MODE = "inp"
HdlDinp = SCPI.CONTrol.HANDler.D.DATA
```

SCPI.CONTrol.HANDler.D.MODE

SCPI

:CONTrol:HANDler:D[:DATA] <numeric>

:CONTrol:HANDler:D[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CONT:HAND:D:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:D 15"
```

```
10 OUTPUT 717;":CONT:HAND:D:MODE INP"
20 OUTPUT 717;":CONT:HAND:D?"
30 ENTER 717;A
```

SCPI.CONTrol.HANDler.D.MODE

SCPI.CONTrol.HANDler.D.MODE = *Param*

Param = SCPI.CONTrol.HANDler.D.MODE

/ I/OD/

	Param
	D/
	(String)
	“INPut”D “OUTPut”D
	“INPut”

```
Dim HdlDmode As String
SCPI.CONTrol.HANDler.D.MODE = "OUTP"
HdlDmode = SCPI.CONTrol.HANDler.D.MODE
```

SCPI.CONTrol.HANDler.D.DATA

SCPI

:CONTrol:HANDler:C:MODE {INPut|OUTPut}

:CONTrol:HANDler:C:MODE?

{INP|OUTP}<newline><^END>

```
10 OUTPUT 717;":CONT:HAND:D:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:D:MODE?"
30 ENTER 717;A$
```

SCPI.CONTrol.HANDler.E.DATA

SCPI.CONTrol.HANDler.E.DATA = *Value* (for output)

Value = SCPI.CONTrol.HANDler.E.DATA (for input port)

I/O/EC+DEC0D3
I/O/EEC0D3
8C0LSBD3MSB

	Value
	/
	(Long)
	0255

```
SCPI.CONTrol.HANDler.C.MODE = "outp"
SCPI.CONTrol.HANDler.D.MODE = "outp"
SCPI.CONTrol.HANDler.E.DATA = 128
Dim HdlEinp As Long
SCPI.CONTrol.HANDler.C.MODE = "inp"
SCPI.CONTrol.HANDler.D.MODE = "inp"
HdlEinp = SCPI.CONTrol.HANDler.E.DATA
```

SCPI.CONTrol.HANDler.C.MODE
SCPI.CONTrol.HANDler.D.MODE
SCPI.CONTrol.HANDler.C.DATA
SCPI.CONTrol.HANDler.D.DATA

SCPI

:CONTrol:HANDler:E[:DATA] <numeric>

:CONTrol:HANDler:E[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":CONT:HAND:C:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:D:MODE OUTP"
30 OUTPUT 717;":CONT:HAND:E 128"
```

```
10 OUTPUT 717;":CONT:HAND:C:MODE INP"
20 OUTPUT 717;":CONT:HAND:D:MODE INP"
30 OUTPUT 717;":CONT:HAND:E?"
40 ENTER 717;A
```

SCPI.CONTrol.HANDler.EXTension.INDex.STATe

SCPI.CONTrol.HANDler.EXTension.INDex.STATe = *Status*

Status = SCPI.CONTrol.HANDler.EXTension.INDex.STATe

/ I/O B6INDEX

B6INDEXINDEX SCPI.CONTrol.HANDler.B.DATA SCPI.CONTrol.HANDler.F.DATA146

	Status
	/INDEX
	(Boolean)
	INDEX INDEX

```
Dim Indx As Boolean
SCPI.CONTrol.HANDler.EXTension.INDex.STATe = 1
Indx = SCPI.CONTrol.HANDler.EXTension.INDex.STATe
```

SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe

SCPI

:CONTrol:HANDler[:EXTension]:INDex:STATe {ON|OFF|1|0}

:CONTrol:HANDler[:EXTension]:INDex:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CONT:HAND:IND:STAT ON"
20 OUTPUT 717;":CONT:HAND:IND:STAT?"
30 ENTER 717;A
```

SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe

SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe = *Status*

Status = SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe

/ I/OB7READY FOR TRIGGER

B7READY FOR TRIGGERREADY
SCPI.CONTrol.HANDler.B.DATA
SCPI.CONTrol.HANDler.F.DATA157

	Status
	/READY FOR TRIGGER
	(Boolean)
	INDEX INDEX

```
Dim RdyTrig As Boolean
SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe = 0
RdyTrig = SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe
```

SCPI.CONTrol.HANDler.EXTension.INDex.STATe

SCPI

:CONTrol:HANDler[:EXTension]:RTRigger:STATe {ON|OFF|1|0}

:CONTrol:HANDler[:EXTension]:RTRigger:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CONT:HAND:RTR:STAT ON"
20 OUTPUT 717;":CONT:HAND:RTR:STAT?"
30 ENTER 717;A
```

SCPI.CONTrol.HANDler.F.DATA

SCPI.CONTrol.HANDler.F.DATA = *Value*

I/OFA+B16A0LSBB7MSB

- INDEX SCPI.CONTrol.HANDler.EXTension.INDex.STATe
  14
- READY FOR TRIGGER
  SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe
  15

	Value
	(Long)
	065535

SCPI.CONTrol.HANDler.F.DATA = 511

SCPI.CONTrol.HANDler.A.DATA

SCPI.CONTrol.HANDler.B.DATA

SCPI

:CONTrol:HANDler:F[:DATA] <numeric>

```
10 OUTPUT 717;":CONT:HAND:F 511"
```

SCPI.CONTrol.HANDler.OUTPut(*Num*).DATA

SCPI.CONTrol.HANDler.OUTPut(*Num*) = *Value*

Value = SCPI.CONTrol.HANDler.OUTPut(*Num*)

/ I/OOUTPUT1OUTPUT2

	Num
	OUTPUT
	(Long)
	12
	1

	Value
	/
	(Long)
	• 1 • 0

```
Dim HdlPol As Long
SCPI.CONTrol.HANDler.OUTPut(1).DATA = 1
HdlPol = SCPI.CONTrol.HANDler.OUTPut(1).DATA
```

SCPI

:CONTrol:HANDler:OUTPut{[1]|2}[:DATA] {1|0}

:CONTrol:HANDler:OUTPut{[1]|2}[:DATA]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":CONT:HAND:OUTP1 1"
20 OUTPUT 717;":CONT:HAND:OUTP1?"
30 ENTER 717;A
```

SCPI.DISPlay.ANNotation.FREQuency.STATe

()

SCPI.DISPlay.ANNotation.FREQuency.STATe = *Status*

Status = SCPI.DISPlay.ANNotation.FREQuency.STATe

/LCD

	Status
	//
	(Boolean)
	• •

```
Dim DispFreq As Boolean
SCPI.DISPlay.ANNotation.FREQuency.STATe = 0
DispFreq = SCPI.DISPlay.ANNotation.FREQuency.STATe
```

Display > **Frequency**

>

SCPI

:DISPlay:ANNotation:FREQuency[:STATe] {ON|OFF|1|0}

:DISPlay:ANNotation:FREQuency[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:ANN:FREQ OFF"
20 OUTPUT 717;":DISP:ANN:FREQ?"
30 ENTER 717;A
```

SCPI.DISPlay.CCLear

SCPI.DISPlay.CCLear

LCD

SCPI.DISPlay.CCLear

SCPI

:DISPlay:CCLear

```
10 OUTPUT 717;":DISP:CCL"
```

SCPI.DISPlay.CLOCk

()

SCPI.DISPlay.CLOCk = *Status*

Status = SCPI.DISPlay.CLOCk

/LCD

	Status
	/
	(Boolean)
	• •

```
Dim DispTime As Boolean
SCPI.DISPlay.CLOCk = ON
DispTime = SCPI.DISPlay.CLOCk
```

System > **Misc Setup** > **Clock Setup** > **Show Clock**
> **Misc** > >

SCPI

:DISPlay:CLOCk {ON|OFF|1|0}

:DISPlay:CLOCk?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:CLOC OFF"
20 OUTPUT 717;":DISP:CLOC?"
30 ENTER 717;A
```

SCPI.DISPlay.COLor(*Dnum*).BACK

()

SCPI.DISPlay.COLor(*Dnum*).BACK = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).BACK

/Dnum:1(Dnum:2)

	Dnum
	1 2
	(Long)
	1 2
	1

	Data
	3 • *(0)* • *(1)* • *(2)* 0
	(Variant)
	• *(0)* 05 • *(1)* 05 • *(2)* 05
	1

```
Dim BackColor As Variant
SCPI.DISPlay.COLor(1).BACK = Array(1,2,3)
BackColor = SCPI.DISPlay.COLor(1).BACK
```

SCPI.DISPlay.COLor(Dnum).RESet

System > **Misc Setup** > **Display Setup** > **Color Setup** > **Normal|Invert** > **Background**

> **Misc** > > > | >

SCPI

:DISPlay:COLor{[1]|2}:BACK <numeric 1>,<numeric 2>,<numeric 3>

:DISPlay:COLor{[1]|2}:BACK?

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

```
10 OUTPUT 717;":DISP:COL:BACK 1,2,3"
20 OUTPUT 717;":DISP:COL:BACK?"
30 ENTER 717;A,B,C
```

SCPI.DISPlay.COLor(*Dnum*).GRATicule(*Gnum*)

()

SCPI.DISPlay.COLor(*Dnum*).GRATicule(*Gnum*) = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).GRATicule(*Gnum*)

/

1.
2. (*Gnum*:1)
3. (*Gnum*:2)

(*Dnum*:1)(*Dnum*:2)

	Gnum
	1 2
	(Long)
	12
	1

	Data
	3 • *(0)* • *(1)* • *(2)* 0
	(Variant)
	• *(0)*05 • *(1)*05 • *(2)*05
	1

```
Dim GritColor As Variant
SCPI.DISPlay.COLor(1).GRATicule(1) = Array(1,2,3)
GritColor = SCPI.DISPlay.COLor(1).GRATicule(1)
```

SCPI.DISPlay.COLor(Dnum).RESet

System > **Misc Setup** > **Display Setup** > **Color Setup** > **Normal|Invert** > **Graticule Main|Graticule Sub**

> **Misc** > > > | > |

SCPI

:DISPlay:COLor{[1]|2}:GRATicule{[1]|2} <numeric 1>,<numeric 2>, <numeric 3>

:DISPlay:COLor{[1]|2}:GRATicule{[1]|2}?

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

```
10 OUTPUT 717;":DISP:COL1:GRAT1 1,2,3"
20 OUTPUT 717;":DISP:COL1:GRAT1?"
30 ENTER 717;A,B,C
```

SCPI.DISPlay.COLor(*Dnum*).LIMit(*Lnum*)

()

SCPI.DISPlay.COLor(*Dnum*).LIMit(*Lnum*) = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).LIMit(*Lnum*)

/

- Lnum:1
- (*Dnum*:1)(*Dnum*: 2)Lnum:2

	Lnum
	1/ 2
	(Long)
	12
	1

	Data
	3 • *(0)* • *(1)* • *(2)* 0
	(Variant)
	• *(0)*05 • *(1)*05 • *(2)*05
	1

```
Dim LimColor As Variant
SCPI.DISPlay.COLor(1).LIMit(1) = Array(1,2,3)
LimColor = SCPI.DISPlay.COLor(1).LIMit(1)
```

SCPI.DISPlay.COLor(Dnum).RESet

System > **Misc Setup** > **Color Setup** > **Normal|Invert** > **Limit Fail|Limit Line**

> **Misc** > > | > |

SCPI

:DISPlay:COLor{[1]|2}:LIMit{[1]|2} <numeric 1>,<numeric 2>,<numeric 3>

:DISPlay:COLor{[1]|2}:LIMit{[1]|2}?

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

```
10 OUTPUT 717;":DISP:COL1:LIM1 1,2,3"
20 OUTPUT 717;":DISP:COL1:LIM1?"
30 ENTER 717;A,B,C
```

SCPI.DISPlay.COLor(*Dnum*).RESet

SCPI.DISPlay.COLor(*Dnum*).RESet

(*Dnum*: 1)(*Dnum*: 2)

SCPI.DISPlay.COLor(1).RESet

SCPI.DISPlay.COLor(Dnum).BACK
SCPI.DISPlay.COLor(Dnum).GRATicule(Gnum)
SCPI.DISPlay.COLor(Dnum).LIMit(Lnum)
SCPI.DISPlay.COLor(Dnum).TRACe(Tr).DATA
SCPI.DISPlay.COLor(Dnum).TRACe(Tr).MEMory

System > **Misc Setup** > **Color Setup** > **Normal|Invert** > **Reset Color** > **OK**

> **Misc** > > | > >

SCPI

:DISPlay:COLor{[1]|2}:RESet

```
10 OUTPUT 717;":DISP:COL1:RES"
```

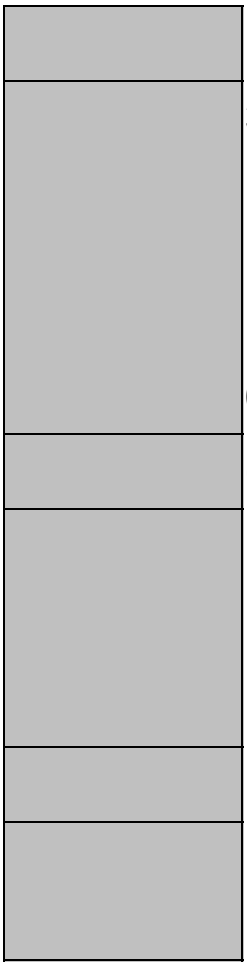
SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).DATA

()

SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).DATA = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).DATA

(*Dnum*: 1)(*Dnum*: 2)/136 (*Tr*)



Data

3

- *(0)*
- *(1)*
- *(2)*

0

(Variant)

- *(0)*05
- *1)*05
- *(2)*05

1

```
Dim TrColor As Variant
SCPI.DISPlay.COLor(1).TRACe(1).DATA = Array(1,2,3)
TrColor = SCPI.DISPlay.COLor(1).TRACe(1).DATA
```

SCPI.DISPlay.COLor(Dnum).RESet

System > **Misc Setup** > **Color Setup** > **Normal|Invert** > **Data Trace 1|Data Trace 2|**

> Misc > > | > 1|2|
Data Trace 3|Data Trace 4|Data Trace 5|Data Trace 6|Data Trace 7|Data Trace 8|Data Trace 9

3|4|5|6|7|8|9

SCPI

:DISPlay:COLor{[1]|2}:TRAC{[1]-36}:DATA <numeric 1>,<numeric 2>, <numeric 3>

:DISPlay:COLor{[1]|2}:TRAC{[1]-36}:DATA?

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

```
10 OUTPUT 717;":DISP:COL1:TRAC1:DATA 1,2,3"
20 OUTPUT 717;":DISP:COL1:TRAC1:DATA?"
30 ENTER 717;A,B,C
```

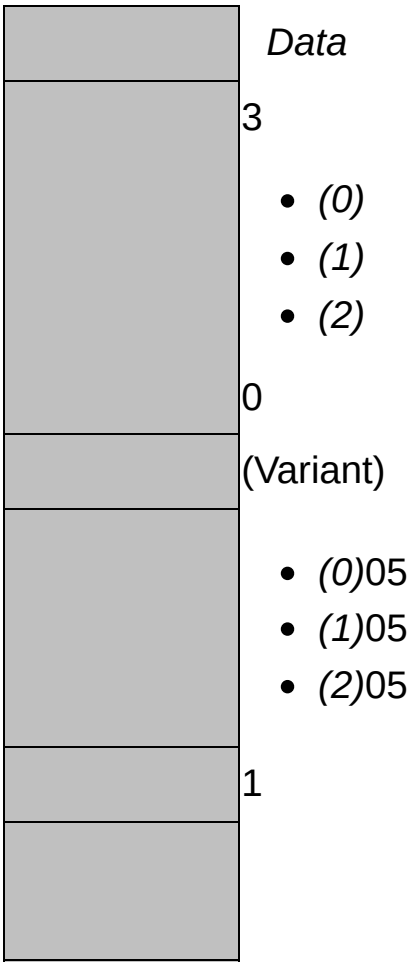
SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).MEMory

()

SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).MEMory = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).MEMory

(*Dnum*: 1)(*Dnum*: 2)/136 (*Tr*)


Data
3
(0)
(1)
(2)
0
(Variant)
(0)05
(1)05
(2)05
1

```
Dim TrColor As Variant
SCPI.DISPlay.COLor(1).TRACe(1).MEMory = Array(1,2,3)
TrColor = SCPI.DISPlay.COLor(1).TRACe(1).MEMory
```

SCPI.DISPlay.COLor(Dnum).RESet

System > **Misc Setup** > **Color Setup** > **Normal|Invert** > **Mem Trace 1|Mem Trace 2|**

> Misc > > | > 1|2|Mem Trace 3|Mem 5|Mem Trace 6|Mem Trace 7|Mem Trace 8|Mem Trace 9

3|4|5|6|7|8|9

SCPI

:DISPlay:COLor{[1]|2}:TRAC{[1]-36}:MEMory <numeric 1>,<numeric 2>, <numeric 3>

:DISPlay:COLor{[1]|2}:TRAC{[1]-36}:MEMory?

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

```
10 OUTPUT 717;":DISP:COL1:TRAC1:MEM 1,2,3"
20 OUTPUT 717;":DISP:COL1:TRAC1:MEM?"
30 ENTER 717;A,B,C
```

SCPI.DISPlay.ECHO.CLEar

SCPI.DISPlay.ECHO.CLEar

echo

SCPI.DISPlay.ECHO.CLEar

SCPI.DISPlay.ECHO.DATA

Macro Setup > **Clear Echo**

> **Echo**

SCPI

:DISPlay:ECHO:CLEar

```
10 OUTPUT 717;":DISP:ECHO:CLE"
```

SCPI.DISPlay.ECHO.DATA

SCPI.DISPlay.ECHO.DATA = *Cont*

ECHOECHO

	Cont
	echo
	(String)
	254

```
SCPI.DISPlay.ECHO.DATA = "Test Result"
SCPI.DISPlay.TABLe.TYPE = "echo"
SCPI.DISPlay.TABLe.STATe = True
```

ECHO

SCPI.DISPlay.TABLe.TYPE

SCPI.DISPlay.TABLe.STATe

SCPI.DISPlay.ECHO.CLEar

SCPI

:DISPlay:ECHO[:DATA] <string>

```
10 OUTPUT 717;":DISP:ECHO ""TEST RESULT"""
```

SCPI.DISPlay.ENABle

()

SCPI.DISPlay.ENABle = *Status*

Status = SCPI.DISPlay.ENABle

/E5071C

	Status
	/E5071C
	(Boolean)
	• •

```
Dim DispUpdt As Boolean
SCPI.DISPlay.ENABle = False
DispUpdt = SCPI.DISPlay.ENABle
```

Display > **Update**

>

SCPI

:DISPlay:ENABle {ON|OFF|1|0}

:DISPlay:ENABle?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:ENAB OFF"
20 OUTPUT 717;":DISP:ENAB?"
30 ENTER 717;A
```

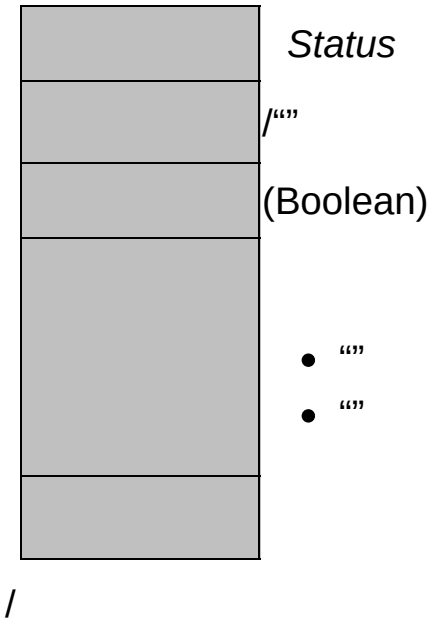
SCPI.DISPlay.FSIGn

()

SCPI.DISPlay.FSIGn = *Status*

Status = SCPI.DISPlay.FSIGn

/LCD“”


Status
/“”
(Boolean)
“”
“”
/

```
Dim DispFail As Boolean
SCPI.DISPlay.FSIGn = False
DispFail = SCPI.DISPlay.FSIGn
```

SCPI.CALCulate(Ch).SELected.LIMit.STATe
SCPI.CALCulate(Ch).SELected.RLIMit.STATe
SCPI.CALCulate(Ch).SELected.BLIMit.STATe

Analysis > **Limit Test > Fail Sign**

> >

Analysis > **Ripple Limit > Fail Sign**

> >

Analysis > **Bandwidth limit > Fail Sign**

> >

SCPI

:DISPlay:FSIGn {ON|OFF|1|0}

:DISPlay:FSIGn?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:FSIG OFF"
20 OUTPUT 717;":DISP:FSIG?"
30 ENTER 717;A
```

SCPI.DISPlay.IMAGe

()

SCPI.DISPlay.IMAGe = *Param*

Param = SCPI.DISPlay.IMAGe

/LCD

	Param
	LCD
	(String)
	• “NORMal” • “INVert”
	“NORMal”

```
Dim DispImg As String
SCPI.DISPlay.IMAGe = "inv"
DispImg = SCPI.DISPlay.IMAGe
```

Display > **Invert Color**

>

SCPI

:DISPlay:IMAGe {NORMal|INVert}

:DISPlay:IMAGe?

{NORM|INV}<newline><^END>

```
10 OUTPUT 717;":DISP:IMAG INV"
20 OUTPUT 717;":DISP:IMAG?"
30 ENTER 717;A$
```

SCPI.DISPlay.MAXimize

()

SCPI.DISPlay.MAXimize = *Status*

Status = SCPI.DISPlay.MAXimize

/

LCD

	Status
	/
	(Boolean)
	• •

```
Dim ChMax As Boolean
SCPI.DISPlay.SPLit = "d1_2"
SCPI.DISPlay.WINDow(2).ACTivate
SCPI.DISPlay.MAXimize = True
ChMax = SCPI.DISPlay.MAXimize
```

SCPI.DISPlay.WINDow(Ch).ACTivate

Channel Max

SCPI

:DISPlay:MAXimize {ON|OFF|1|0}

:DISPlay:MAXimize?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:MAX ON"
20 OUTPUT 717;":DISP:MAX?"
30 ENTER 717;A
```

SCPI.DISPlay.SKEY.STATe

()

SCPI.DISPlay.SKEY.STATe = *Status*

Status = SCPI.DISPlay.SKEY.STATe

/

Status

/

(Boolean)

-
-

```
Dim DispSkey As Boolean
SCPI.DISPlay.SKEY.STATe = False
DispSkey = SCPI.DISPlay.SKEY.STATe
```

Entry Off

SCPI

:DISPlay:SKEY[:STATe] {ON|OFF|1|0}

:DISPlay:SKEY[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:MAX ON"
20 OUTPUT 717;":DISP:MAX?"
30 ENTER 717;A
```

SCPI.DISPlay.SPLit

()

SCPI.DISPlay.SPLit = *Param*

Param = SCPI.DISPlay.SPLit

LCD

	Param
	(String)
	• “D1” • “D12” • “D1_2” • “D112” • “D1_1_2” • “D123” • “D1_2_3” • “D12_33” • “D11_23” • “D13_23” • “D12_13” • “D1234” • “D1_2_3_4” • “D12_34” • “D123_456” • “D12_34_56” • “D1234_5678” • “D12_34_56_78” • “D123_456_789” • “D123__ABC” • “D1234__9ABC” • “D1234__DEFG”

	“D1”

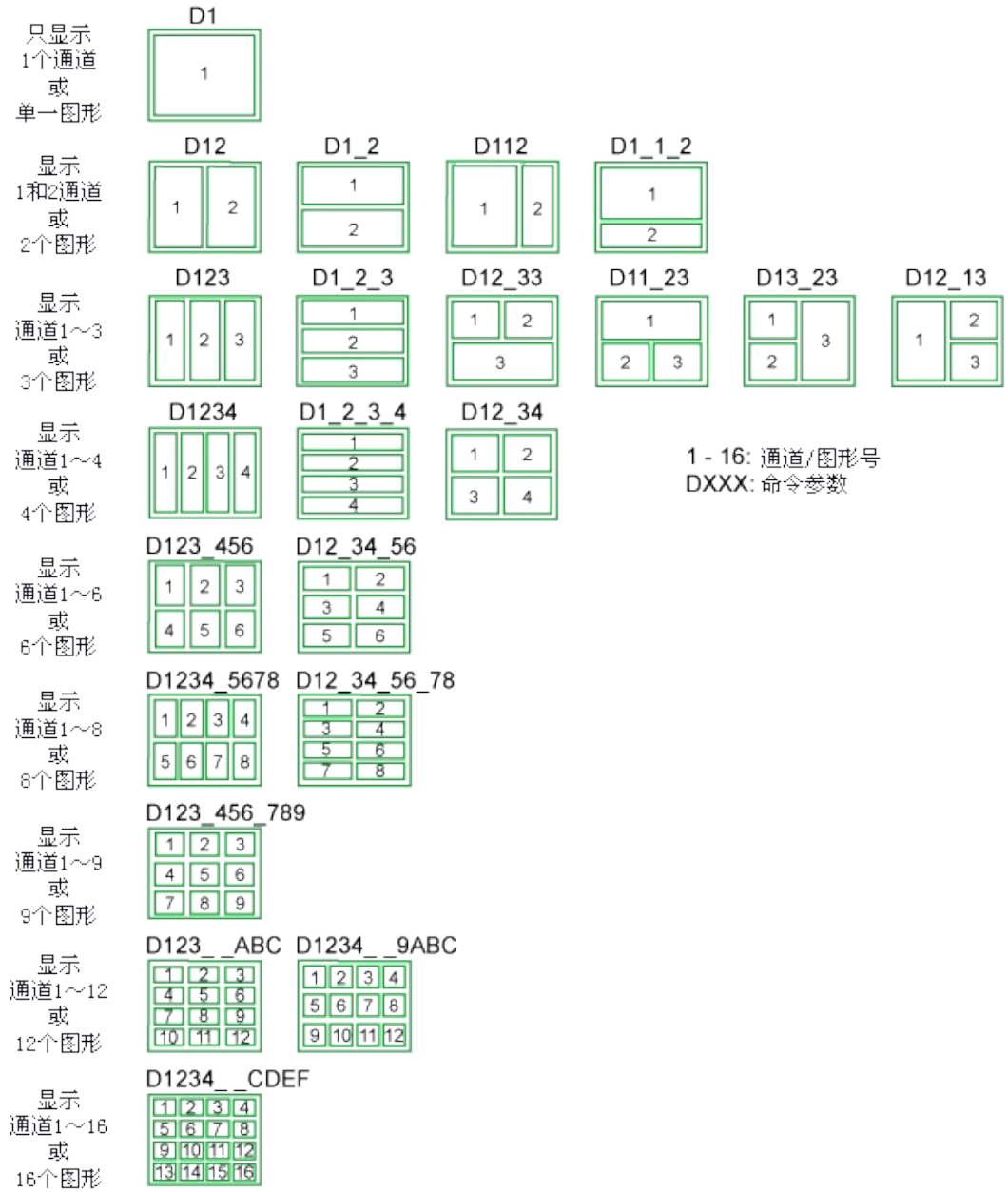
```
Dim ChanAloc As String
SCPI.DISPlay.SPLit = "d12_34"
ChanAloc = SCPI.DISPlay.SPLit
```

SCPI.DISPlay.WINDow(Ch).SPLit

Display > Allocate Channels

>

/



e5071c314

SCPI

:DISPlay:SPLit
{D1|D12|D1_2|D112|D1_1_2|D123|D1_2_3|D12_33|D11_23|D13_23|D12_
D1234|D1_2_3_4|D12_34|D123_456|D12_34_56|D1234_5678|D12_34_5
D123__ABC|D1234__9ABC|D1234__DEFG}

:DISPlay:SPLit?

{D1|D12|D1_2|D112|D1_1_2|D123|D1_2_3|D12_33|D11_23|D13_23|D12_
D12_34|D123_456|D12_34_56|D1234_5678|D12_34_56_78|D123_456_7
D1234__9ABC|D1234__DEFG}<newline><^END>

```
10 OUTPUT 717;":DISP:SPL D1_2"
20 OUTPUT 717;":DISP:SPL?"
30 ENTER 717;A$
```

SCPI.DISPlay.TABLe.POSition.RECTangle

()

Param = SCPI.DISPlay.TABLe.POSition.RECTangle

[0, 0]	SCPI.DISPlay.TABLe.STATe	0000

	Param
	• *(0)* X • *(1)* Y • *(2)* X • *(3)* Y 0
	(Variant)
	1

```
Dim TablePos() As Variant
SCPI.DISPlay.TABLe.STATe = True
TablePos = SCPI.DISPlay.TABLe.POSition.RECTangle
```

SCPI.DISPlay.TABLe.STATe

SCPI

:DISPlay:TABLe:POSition[:RECTangle]?

{numeric1},{numeric2},{numeric3},{numeric4},<newline><^END>

```
10 OUTPUT 717;":DISP:TABL:STAT ON"
20 OUTPUT 717;":DISP:TABL:POS?"
20 ENTER 717;A, B, C, D
```

SCPI.DISPlay.TABLe.STATe

()

SCPI.DISPlay.TABLe.STATe = *Status*

Status = SCPI.DISPlay.TABLe.STATe

/LCD SCPI.DISPlay.TABLe.TYPE

	Status
	/LCD
	(Boolean)
	• •

```
Dim DispTbl As Boolean
SCPI.DISPlay.TABLe.TYPE = "echo"
SCPI.DISPlay.TABLe.STATe = True
DispTbl = SCPI.DISPlay.TABLe.STATe
```

SCPI.DISPlay.TABLe.TYPE

Sweep Setup > **Edit Segment Table**

>

Marker Fctn > **Marker Table**

>

Analysis > **Limit Test** > **Edit Limit Line**

> >

Analysis > **Ripple Limit** > **Edit Ripple Line**

> >

Macro Setup > **Echo Window**

>

Cal > **Power Calibration** > **Loss Compen**

> >

Cal > **Power Calibration** > **Sensor A Settings | Sensor B Settings**

> > **A|B**

LCD/

SCPI

:DISPlay:TABLe[:STATe] {ON|OFF|1|0}

:DISPlay:TABLe[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:TABL ON"
20 OUTPUT 717;":DISP:TABL?"
30 ENTER 717;A
```

SCPI.DISPlay.TABLe.TYPE

()

SCPI.DISPlay.TABLe.TYPE = *Param*

Param = SCPI.DISPlay.TABLe.TYPE

LCD

	Param
	(String)
	• “MARKer” • “LIMit” • “SEGMent” • “ECHO”echo • “PLOSs” • “SCFactor” • “RLIMit”
	“MARKer”

```
Dim TblType As String
SCPI.DISPlay.TABLe.TYPE = "echo"
SCPI.DISPlay.TABLe.STATe = True
TblType = SCPI.DISPlay.TABLe.TYPE
```

SCPI.DISPlay.TABLe.STATe

Sweep Setup > **Edit Segment Table**

>

Marker Fctn > **Marker Table**

>

Analysis > **Limit Test** > **Edit Limit Line**

> >

Analysis > **Ripple Limit** > **Edit Ripple Line**

> >

Macro Setup > **Echo Window**

> **Echo**

Cal > **Power Calibration** > **Loss Compen**

> >

Cal > **Power Calibration** > **Sensor A Settings|Sensor B Settings**

> > **A|B**

LCD/

SCPI

:DISPlay:TABLe:TYPE
{MARKer|LIMit|SEGMent|ECHO|PLOSs|SCFactor|RLIMit}

:DISPlay:TABLe:TYPE?

{MARK|LIM|SEGM|ECHO|PLOS|SCF|RLIM}<newline><^END>

```
10 OUTPUT 717;":DISP:TABL:TYPE SEGM"
20 OUTPUT 717;":DISP:TABL:TYPE?"
30 ENTER 717;A$
```

SCPI.DISPlay.UPDate.IMMediate

SCPI.DISPlay.UPDate.IMMediate

LCD SCPI.DISPlay.ENABle

```
SCPI.DISPlay.ENABle = False
SCPI.DISPlay.UPDate.IMMediate
```

SCPI.DISPlay.ENABle

SCPI

:DISPlay:UPDate[:IMMediate]

```
10 OUTPUT 717;":DISP:UPD"
```

SCPI.DISPlay.WINDow(*Ch*).ACTivate

SCPI.DISPlay.WINDow(*Ch*).ACTivate

(*Ch*)

```
SCPI.DISPlay.SPLit = "d1_2"
SCPI.DISPlay.WINDow(2).ACTivate
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI

:DISPlay:WINDow{[1]-36}:ACTivate

```
10 OUTPUT 717;":DISP:WIND1:ACT"
```

SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.ALIGn.STAT

()

SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.ALIGn.STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.ALIGn.STATe

(*Ch*)/1

	Status
	/1
	(Boolean)
	• 1 •

```
Dim AnnMarkAlig As Boolean
SCPI.DISPlay.WINDow(1).ANNotation.MARKer.ALIGn.STATe = False
AnnMarkAlig =
SCPI.DISPlay.WINDow(1).ANNotation.MARKer.ALIGn.STATe
```

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.SINGle.STATe
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.X
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.Y

Marker Fctn > **Annotation Options** > **Align**

> >

SCPI

:DISPlay:WINDow{[1]-36}:ANNotation:MARKer:ALIGn[:STATe] {ON|OFF|1|0}

:DISPlay:WINDow{[1]-36}:ANNotation:MARKer:ALIGn[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:ANN:MARK:ALIG OFF"
20 OUTPUT 717;":DISP:WIND1:ANN:MARK:ALIG?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.SINGle.STA

()

SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.SINGle.STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.SINGle.STATe

(Ch)/

Status

/

(Boolean)

-
-

```
Dim AnnMarkAlig As Boolean
SCPI.DISPlay.WINDow(1).ANNotation.MARKer.SINGle.STATe = False
AnnMarkAlig =
SCPI.DISPlay.WINDow(1).ANNotation.MARKer.SINGle.STATe
```

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.ALIGn.STATe
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.X
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.Y

Marker Fctn > **Annotation Options** > **Active Only**

> >

SCPI

:DISPlay:WINDow{[1]-36}:ANNotation:MARKer:SINGle[:STATe] {ON|OFF|1|0}

:DISPlay:WINDow{[1]-36}:ANNotation:MARKer:SINGle[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:ANN:MARK:SING OFF"
20 OUTPUT 717;":DISP:WIND1:ANN:MARK:SING?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).LABel

()Property (**Read-Write**)

SCPI.DISPlay.WINDow(*Ch*).LABel = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).LABel

/136 *(Ch)*

	Status
	/
	(Boolean)
	• •

```
Dim DispGrat As Boolean
SCPI.DISPlay.WINDow(1).LABel = False
DispGrat = SCPI.DISPlay.WINDow(1).LABel
```

Display > **Graticule Label**

>

SCPI

:DISPlay:WINDow{[1]-36}:LABel {ON|OFF|1|0}

:DISPlay:WINDow{[1]-36}:LABel?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:LAB ON"
20 OUTPUT 717;":DISP:WIND1:LAB?"
30 ENTER 717;A
```

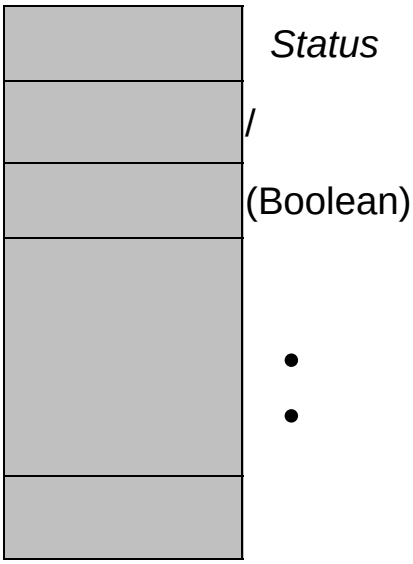
SCPI.DISPlay.WINDow(*Ch*).MAXimize

()

SCPI.DISPlay.WINDow(*Ch*).MAXimize = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).MAXimize

/ (*Ch*)


Status
/
(Boolean)

```
Dim TracMax As Boolean
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.DISPlay.WINDow(1).MAXimize = True
TracMax = SCPI.DISPlay.WINDow(1).MAXimize
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.DISPlay.MAXimize

Trace Max

SCPI

:DISPlay:WINDow{[1]-36}:MAXimize {ON|OFF|1|0}

:DISPlay:WINDow{[1]-36}:MAXimize?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:MAX ON"
20 OUTPUT 717;":DISP:WIND1:MAX?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).SPLit

()

SCPI.DISPlay.WINDow(*Ch*).SPLit = *Param*

Param = SCPI.DISPlay.WINDow(*Ch*).SPLit

/ (*Ch*)

	Param
	(String)
	• “D1” • “D12” • “D1_2” • “D112” • “D1_1_2” • “D123” • “D1_2_3” • “D12_33” • “D11_23” • “D13_23” • “D12_13” • “D1234” • “D1_2_3_4” • “D12_34” • “D123_456” • “D12_34_56” • “D1234_5678” • “D12_34_56_78” • “D123_456_789” • “D123__ABC” • “D1234__9ABC” • “D1234__DEFG”

	“D1”

```
Dim TracAloc As String
SCPI.DISPlay.WINDow(1).SPLit = "d1_2"
TracAloc = SCPI.DISPlay.WINDow(1).SPLit
```

SCPI.DISPlay.SPLit

Display > **Allocate Traces**

>

SCPI

:DISPlay:WINDow{[1]-36}:SPLit {D1|D12|D1_2|D112|D1_1_2|
D123|D1_2_3|D12_33|D11_23|D13_23|D12_13|D1234|D1_2_3_4|D12_3
D1234_5678|D12_34_56_78|D123_456_789|D123__ABC|D1234__9ABC

:DISPlay:WINDow{[1]-36}:SPLit?

{D1|D12|D1_2|D112|D1_1_2|D123|D1_2_3|D12_33|D11_23|D13_23|D12
D12_34|D123_456|D12_34_56|D1234_5678|D12_34_56_78|D123_456_
D1234__9ABC|D1234__DEFG}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND:SPL D1_2"
20 OUTPUT 717;":DISP:WIND:SPL?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).TITLe.DATA

()

SCPI.DISPlay.WINDow(*Ch*).TITLe.DATA = *Lbl*

Lbl = SCPI.DISPlay.WINDow(*Ch*).TITLe.DATA

/ (*Ch*)

	Lbl
	(String)
	254
	“”

```
Dim TtlLbl As String
SCPI.DISPlay.WINDow(1).TITLe.DATA = "Filter"
SCPI.DISPlay.WINDow(1).TITLe.STATe = True
TtlLbl = SCPI.DISPlay.WINDow(1).TITLe.DATA
```

SCPI.DISPlay.WINDow(Ch).TITLe.STATe

Display > **Edit Title Label**

>

SCPI

:DISPlay:WINDow{[1]-36}:TITLe:DATA <string>

:DISPlay:WINDow{[1]-36}:TITLe:DATA?

{string}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:TITL:DATA ""Title"""
20 OUTPUT 717;":DISP:WIND1:TITL?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).TITLe.STATe

()

SCPI.DISPlay.WINDow(*Ch*).TITLe.STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).TITLe.STATe

/136 *(Ch)*

	Status
	/
	(Boolean)
	• •

```
Dim DispTtl As Boolean
SCPI.DISPlay.WINDow(1).TITLe.DATA = "Filter"
SCPI.DISPlay.WINDow(1).TITLe.STATe = True
DispTtl = SCPI.DISPlay.WINDow(1).TITLe.STATe
```

SCPI.DISPlay.WINDow(Ch).TITLe.DATA

Display > **Title Label**

>

SCPI

:DISPlay:WINDow{[1]-36}:TITLe[:STATe] {ON|OFF|1|0}

:DISPlay:WINDow{[1]-36}:TITLe[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:TITL ON"
20 OUTPUT 717;":DISP:WIND1:TITL?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.F

()

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.X = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.X

/X (*Ch*)(*Tr*)

	Value
	X
	(Double)
	-15100
	1
	%()

```
Dim AnnMPosX As Double
SCPI.DISPlay.WINDow(1).TRACe(1).ANNotation.MARKer.POSition.X =
15
AnnMPosX =
SCPI.DISPlay.WINDow(1).TRACe(1).ANNotation.MARKer.POSition.X
```

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.ALIGn.STATe
SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.SINGle.STATe
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.Y

Marker Fctn > **Annotation Options** > **Marker Info X Pos**

> > Info X Pos

SCPI

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16}:ANNotation:MARKer:POSition:X
<numeric>

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16}:ANNotation:MARKer:POSition:X

{numeric}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:ANN:MARK:POS:X 33"
20 OUTPUT 717;":DISP:WIND1:TRAC1:ANN:MARK:POS:X?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.F

()

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.Y = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.X

/X (*Ch*)(*Tr*)/Y

	Value
	Y
	(Double)
	-15100
	1
	%()

```
Dim AnnMPosY As Double
SCPI.DISPlay.WINDow(1).TRACe(1).ANNotation.MARKer.POSition.Y =
23
AnnMPosY =
SCPI.DISPlay.WINDow(1).TRACe(1).ANNotation.MARKer.POSition.Y
```

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.ALIGn.STATe
SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.SINGle.STATe
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.X

Marker Fctn > **Annotation Options** > **Marker Info X Pos**

> > Info X Pos

SCPI

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16}:ANNotation:MARKer:POSition:Y
<numeric>

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16}:ANNotation:MARKer:POSition:Y

{numeric}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:ANN:MARK:POS:Y 33"
20 OUTPUT 717;":DISP:WIND1:TRAC1:ANN:MARK:POS:Y?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).MEMory. STATe

()

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).MEMory.STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).MEMory.STATe

(*Ch*)(*Tr*)/

	Status
	/
	(Boolean)
	• •

```
Dim DispMem As Boolean
SCPI.DISPlay.WINDow(1).TRACe(2).MEMory.STATe = True
DispMem = SCPI.DISPlay.WINDow(1).TRACe(2).MEMory.STATe
```

SCPI.CALCulate(Ch).SELected.MATH.MEMorize

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).STATe

Display > **Display** > **Mem** (when the data trace display is OFF)

> > ()

Display > **Display** > **Data & Mem** (when the data trace display is ON)

> > ()

SCPI

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16} :MEMory[:STATe] {ON|OFF|1|0}

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16} :MEMory[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:MEM ON"
20 OUTPUT 717;":DISP:WIND1:TRAC1:MEM?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).STATe

()

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).STATe

(*Ch*)(*Tr*)/

Status

/

(Boolean)

-
-

```
Dim DispTrac As Boolean
SCPI.DISPlay.WINDow(1).TRACe(2).STATe = False
DispTrac = SCPI.DISPlay.WINDow(1).TRACe(2).STATe
```

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).MEMory.STATe

Display > **Display** > **Data** (when the memory trace display is OFF)

> > ()

Display > **Display** > **Data & Mem** (when the memory trace display is ON)

> > ()

SCPI

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16} :STATe {ON|OFF|1|0}

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16} :STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:STAT ON"
20 OUTPUT 717;":DISP:WIND1:TRAC1:STAT?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.AUTO

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.AUTO

(*Ch*)(*Tr*)

SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.AUTO

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.PDIVision
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RLEVel

Scale > Auto Scale

>

SCPI

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16} :Y[:SCALe]:AUTO

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:AUTO"
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe. PDIVision

()

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.PDIVision = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.PDIVision

Smith (*Ch*)(*Tr*)Smith

	Value
	(Double)
	1E-181E8
	• 10 • 90 • 1E-8 • Smith1 • 0.1 • 0.2
	• dB • ° • s •

```
Dim Pdiv As Double
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.FORMat = "gdel"
SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.PDIVision = 1E-9
Pdiv = SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.PDIVision
```

SCPI.CALCulate(Ch).SELected.FORMat
SCPI.DISPlay.WINDow(Ch).Y.SCALe.DIVisions
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RLEVel
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RPOSition

Scale > **Scale/Div**

> **/**

SCPI

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16} :Y[:SCALe]:PDIVision <numeric>

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16} :Y[:SCALe]:PDIVision?

{numeric}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:PDIV 2.5"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:PDIV?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RLEVel

()

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RLEVel = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RLEVel

(*Ch*)(*Tr*) /

	Value
	(Double)
	-5E85E8
	0
	• (MLOG): dB () • (PHAS)(UPH)(PPH) °() • (GDEL)s() •

```
Dim RefLvl As Double
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.FORMat = "phas"
SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.RLEVel = 90
Pdiv = SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.RLEVel
```

SCPI.CALCulate(Ch).SELected.FORMat
SCPI.DISPlay.WINDow(Ch).Y.SCALe.DIVisions
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.PDIVision
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RPOSition

[Scale] > **Reference Value**

[] >

SCPI

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16} :Y[:SCALe]:RLEVel <numeric>

:DISPlay:WINDow{[1]-36}:TRACe{[1]-16} :Y[:SCALe]:RLEVel?

{numeric}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:RLEV 1E2"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:RLEV?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RPOSition

()

SCPI.DISPlay.WINDow(*Ch*).TRACe(*T36r*).Y.SCALe.RPOSition = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RPOSition

(*Ch*)(*Tr*)0

	Value
	(Long)
	0
	5

```
Dim RefPos As Long
SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.RPOSition = 6
RefPos = SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.RPOSition
```

SCPI.CALCulate(Ch).SELected.FORMat
SCPI.DISPlay.WINDow(Ch).Y.SCALe.DIVisions
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.PDIVision
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RLEVel

Scale > Reference Position

>

SCPI

:DISPlay:WINDow{[1]-36}:TRACe{[1]-36} :Y[:SCALe]:RPOSition <numeric>

:DISPlay:WINDow{[1]-36}:TRACe{[1]-36} :Y[:SCALe]:RPOSition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:RPOS 6"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:RPOS?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).X.SPACing

()

SCPI.DISPlay.WINDow(*Ch*).X.SPACing = *Param*

Param = SCPI.DISPlay.WINDow(*Ch*).X.SPACing

(*Ch*)

	Param
	(String)
	• “LINear” • “OBASe”
	“OBASe”

```
Dim DispSegm As String
SCPI.SENSe(1).SWEep.TYPE = "segm"
SCPI.DISPlay.WINDow(1).X.SPACing = "obas"
DispSegm = SCPI.DISPlay.WINDow(1).X.SPACing
```

SCPI.SENSe(Ch).SWEep.TYPE

>

SCPI

:DISPlay:WINDow{[1]-36}:X:SPACing {LINear|OBASe}

:DISPlay:WINDow{[1]-36}:X:SPACing?

{LIN|OBAS}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:X:SPAC OBAS"
20 OUTPUT 717;":DISP:WIND1:X:SPAC?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).Y.SCALe.DIVisions

()

SCPI.DISPlay.WINDow(*Ch*).Y.SCALe.DIVisions = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).Y.SCALe.DIVisions

(Ch)/

	Value
	(Long)
	430
	10
	2

```
Dim Divs As Long
SCPI.DISPlay.WINDow(1).Y.SCALe.DIVisions = 12
Divs = SCPI.DISPlay.WINDow(1).Y.SCALe.DIVisions
```

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.PDIVision
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RLEVel
SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RPOSition

Scale > Divisions

>

SCPI

:DISPlay:WINDow{[1]-36}:Y[:SCALe]:DIVisions <numeric>

:DISPlay:WINDow{[1]-36}:Y[:SCALe]:DIVisions?

{numeric}<newline><^END>

```
10 OUTPUT 717;":DISP:WIND1:Y:DIV 12"
20 OUTPUT 717;":DISP:WIND1:Y:DIV?"
30 ENTER 717;A
```

SCPI.FORMat.BORDer

()

SCPI.FORMat.BORDer = *Param*

Param = SCPI.FORMat.BORDer

SCPI.FORMat.DATA“REAL”/()

COME5071C VBAE5071C

	Param
	(String)
	“NORMal”MSB() “SWAPped”LSB()
	“NORMal”

```
Dim BitOrd As String
SCPI.FORMat.BORDer "swap"
BitOrd = SCPI.FORMat.BORDer
```

SCPI.FORMat.DATA

SCPI

:FORMat:BORDer {NORMal|SWAPped}

:FORMat:BORDer?

{NORM|SWAP}<newline><^END>

```
10 OUTPUT 717;":FORM:BORD SWAP"
20 OUTPUT 717;":FORM:BORD?"
30 ENTER 717;A$
```

SCPI.FORMat.DATA

()

SCPI.FORMat.DATA = *Param*

Param = SCPI.FORMat.DATA

/SCPI

:CALC{1-36}:DATA:FDAT
:CALC{1-36}:DATA:FMEM
:CALC{1-36}:DATA:SDAT?
:CALC{1-36}:DATA:SMEM?
:CALC{1-36}:FUNC:DATA?
:CALC{1-36}:LIM:DATA
:CALC{1-36}:LIM:REP?
:CALC{1-36}:LIM:REP:ALL?
:CALC{1-36}:BLIM:REP?
:CALC{1-36}:RLIM:DATA?
:CALC{1-36}:RLIM:REP?
:SENS{1-36}:FREQ:DATA?
:SENS{1-36}:SEGM:DATA
:SOUR:POW:PORT:CORR:COLL:TABL:ASEN:DATA
:SOUR:POW:PORT:CORR:COLL:TABL:BSEN:DATA
:SOUR{1-36}:POW:PORT{1-4}:CORR:COLL:TABL:LOSS:DATA
:SOUR{1-36}:POW:PORT{1-4}:CORR:DATA

E5071C VBASCPIE5071CASCII

	Param
	(String)
	“ASCii”ASCII “REAL”IEEE 64 “REAL32”IEEE 32
	“ASCii”

```
Dim Fmt As String
SCPI.FORMat.DATA = "ASC"
Fmt = SCPI.FORMat.DATA
```

SCPI.FORMat.BORDer

SCPI

:FORMat:DATA {ASCii|REAL|REAL32}

:FORMat:DATA?

{ASC|REAL|REAL32}<newline><^END>

```
10 OUTPUT 717;":FORM:DATA REAL"
20 OUTPUT 717;":FORM:DATA?"
30 ENTER 717;A$
```

SCPI.HCOPy.ABORt

SCPI.HCOPy.ABORt

SCPI.HCOPy.ABORt

SCPI.HCOPy.IMMediate

System > **Abort Printing**

>

SCPI

:HCOPy:ABORt

```
10 OUTPUT 717;":HCOP:ABOR"
```

SCPI.HCOPy.IMAGe

SCPI.HCOPy.IMAGe = *Param*

Param = SCPI.HCOPy.IMAGe

/

	Param
	(String)
	• “NORMal” • “INVert”
	“INVert”

```
Dim Img As String
SCPI.HCOPy.IMAGe = "norm"
Img = SCPI.HCOPy.IMAGe
```

SCPI.HCOPy.IMMediate

System > **Invert Image**

>

SCPI

:HCOPy:IMAGe {NORMal|INVert}

:HCOPy:IMAGe?

{NORM|INV}<newline><^END>

```
10 OUTPUT 717;":HCOP:IMAG NORM"
20 OUTPUT 717;":HCOP:IMAG?"
30 ENTER 717;A$
```

SCPI.HCOPy.IMMediate

SCPI.HCOPy.IMMediate

LCDE5071C

E5071CVisual BasicVBA

SCPI.HCOPy.IMMediate

SCPI.HCOPy.ABORt
SCPI.HCOPy.IMAGe

[System] > **Print**

[] >

LCDLCDLCD

SCPI.HCOPy.IMMediate

SCPI

:HCOPy[:IMMediate]

```
10 OUTPUT 717;":HCOP"
```

SCPI.IEEE4882.CLS

SCPI.IEEE4882.CLS

SCPI.IEEE4882.CLS

SCPI

*CLS

```
10 OUTPUT 717;"*CLS"
```

SCPI.IEEE4882.ESE

()

SCPI.IEEE4882.ESE = *Value*

Value = SCPI.IEEE4882.ESE

/

	Value
	(Long)
	0255
	0
	2550xffAND

```
Dim Stat As Long
SCPI.IEEE4882.ESE = 16
Stat = SCPI.IEEE4882.ESE
```

SCPI.IEEE4882.SRE

SCPI

*ESE <numeric>

*ESE?

{numeric}<newline><^END>

```
10 OUTPUT 717;"*ESE 16"
20 OUTPUT 717;"*ESE?"
30 ENTER 717;A
```

SCPI.IEEE4882.ESR

()

Value = SCPI.IEEE4882.ESR

	Value
	(Long)
	128

```
Dim Stat As Long
Stat = SCPI.IEEE4882.ESR
```

SCPI

*ESR?

{numeric}<newline><^END>

```
10 OUTPUT 717;"*ESR?"
20 ENTER 717;A
```

SCPI.IEEE4882.IDN

()

Cont = SCPI.IEEE4882.IDN

E5071C

	Cont
	“{1}{2}{3}{4}” • {1} • {2}E5071C • {3}MY123400101 • {4}:A.07.00
	(String)

```
Dim Who As String
Who = SCPI.IEEE4882.IDN
```

System > **Firmware Revision**

>

System > **Service Menu** > **Enable Options** > **Serial Number**

> > >

SCPI

*IDN?

{string 1},{string 2},{string 3},{string 4}<newline><^END>

```
10 OUTPUT 717;"*IDN?"
20 ENTER 717;A$
```

SCPI.IEEE4882.OPC

()

SCPI.IEEE4882.OPC = *Dummy Value* = SCPI.IEEE4882.OPC

/“1”OPC0

Case (2):

	Value
	1
	(Long)

```
SCPI.IEEE4882.OPC = 1
Dmy = SCPI.IEEE4882.OPC
```

SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.ISOLation
SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.LOAD
SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.OPEN
SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.SHORt
SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.THRU
SCPI.TRIGger.SEQuence.SINGle

SCPI

*OPC

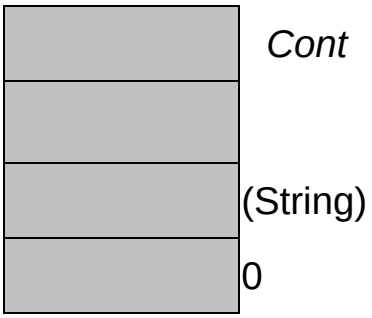
```
10 OUTPUT 717;"*OPC"
```

SCPI.IEEE4882.OPT

()

Cont = SCPI.IEEE4882.OPT

E5071C


Cont
(String)
0

```
Dim OptNum As String
OptNum = SCPI.IEEE4882.OPT
```

SCPI

*OPT?

{numeric}<newline><^END>

```
Call viVPrintf(vi, "*OPT?" + vbLf, 0)
Call viVScanf(vi, "%t", Result)
```

SCPI.IEEE4882.RST

SCPI.IEEE4882.RST

E5071C 1 SCPI.INITiate(Ch).CONTinuousSCPI.SYSTem.PRESet

SCPI.IEEE4882.RST

SCPI.INITiate(Ch).CONTinuous

SCPI.SYSTem.PRESet

SCPI.SYSTem.UPReset

SCPI

*RST

```
10 OUTPUT 717;"*RST"
```

SCPI.IEEE4882.SRE

()

SCPI.IEEE4882.SRE = *Value*

Value = SCPI.IEEE4882.SRE

/

	Value
	(Long)
	0255
	0
	AND 2550xff61

```
Dim Stat As Long
SCPI.IEEE4882.SRE = 8
Stat = SCPI.IEEE4882.SRE
```

SCPI.IEEE4882.ESE

SCPI.STATus.OPERation.ENABle

SCPI.STATus.QUEStionable.ENABle

SCPI

*SRE <numeric>

*SRE?

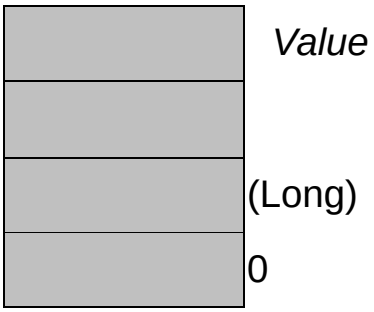
{numeric}<newline><^END>

```
10 OUTPUT 717;"*SRE 128"
20 OUTPUT 717;"*SRE?"
30 ENTER 717;A
```

SCPI.IEEE4882.STB

()

Value = SCPI.IEEE4882.STB


Value
(Long)
0

```
Dim Stat As Long
Stat = SCPI.IEEE4882.STB
```

SCPI

*STB?

{numeric}<newline><^END>

```
10 OUTPUT 717;"*STB?"
20 ENTER 717;A
```

SCPI.IEEE4882.TRG

SCPI.IEEE4882.TRG

GPIB/LAN SCPI.TRIGger.SEQuence.SOURceE5071C

```
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.IEEE4882.TRG
```

SCPI.TRIGger.SEQuence.SOURce

SCPI

*TRG

```
10 OUTPUT 717;"*TRG"
```

SCPI.IEEE4882.WAI

SCPI.IEEE4882.WAI

```
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.TRIGger.SEQuence.SINGle
SCPI.IEEE4882.WAI
MsgBox "Done"
```

SCPI

*WAI

```
10 OUTPUT 717;"*WAI"
```

SCPI.INITiate(*Ch*).CONTinuous

SCPI.INITiate(*Ch*).CONTinuous = *Status*

Status = SCPI.INITiate(*Ch*).CONTinuous

/ (Ch)

	Status
	/
	(Boolean)
	• •
	[(*Ch*)]

```
Dim ContMode As Boolean
SCPI.INITiate(2).CONTinuous = True
ContMode = SCPI.INITiate(2).CONTinuous
```

SCPI.INITiate(Ch).IMMediate

Trigger > **Continuous** (continuous initiation mode ON)

>

Trigger > **Hold** (continuous initiation mode OFF)

>

SCPI

:INITiate{[1]-36}:CONTinuous {ON|OFF|1|0}

:INITiate{[1]-36}:CONTinuous?

{1|0}<newline><^END>

```
10 OUTPUT 717;":INIT1:CONT OFF"
20 OUTPUT 717;":INIT1:CONT?"
30 ENTER 717;A
```

SCPI.INITiate(*Ch*).IMMediate

SCPI.INITiate(*Ch*).IMMediate

(*Ch*)

```
SCPI.INITiate(1).CONTinuous = False
SCPI.INITiate(1).IMMediate
```

SCPI.INITiate(Ch).CONTinuous

Trigger > **Single**

>

SCPI

:INITiate{[1]-36}[:IMMediate]

```
10 OUTPUT 717;":INIT1"
```

:MMEMory:LOAD:PROGram

COM

:MMEMory:LOAD:PROGram <string>

VBA.vba.bas.frm.cls

	String
	VBA
	254
	“”

```
10 OUTPUT 717;":MMEM:LOAD:PROG ""Test1/Test1_01.vba"""
10 OUTPUT 717;":MMEM:LOAD:PROG ""A:Test1_01.vba"""
```

:MMEM:STOR:PROG

Macro Setup > **Load VBA Project**

Macro > **VBA**

:MMEMory:STORe:PROGram

COM

:MMEMory:STORe:PROGram <string>

VBAVBA.vba

	String
	VBA
	254
	“”

```
10 OUTPUT 717;":MMEM:STOR:PROG ""Test1/Test1_01.vba"""
10 OUTPUT 717;":MMEM:STOR:PROG ""D:Test1_01.vba"""
```

:MMEM:LOAD:PROG

Macro Setup > **Save VBA Project**
> **VBA**

:MMEMory:TRANsfer

COM

:MMEMory:TRANsfer <string>,<block>

:MMEMory:TRANsfer? <string>

E5071CE5071CE5071CE5071C

E5071CE5071C

“/”“\”

	String
	254

	block
	GPIB20 M20 M LAN100 K100 K

{block}<newline><^END>

```
10 OUTPUT 717;":MMEM:TRAN ""Trace01.csv"",#6012345";Dat$
10 OUTPUT 717;":MMEM:TRAN? ""Trace01.csv"""
20 ENTER 717 USING "#,A";A$
30 ENTER 717 USING "#,A";Digit$
40 Img$="#,"&Digit$&"A"
50 ENTER 717 USING Img$;Byte$
60 Img$=Byte$&"A"
70 ALLOCATE Dat$[VAL(Byte$)]
80 ENTER 717 USING Img$;Dat$
```

SCPI.MMEMory.CATalog(*Dir*)

Cont = SCPI.MMEMory.CATalog(*Dir*)

E5071C

-
-
-

“\”“\”“/”

	Cont
	“{A}{B}{1}{1}{2}{2}……{N}{N}” Nn1N {A} {B} {n}n {n}n0
	(String)

	Dir
	(String)
	254

```
Dim DirCont As String
DirCont = SCPI.MMEMory.CATalog("d:\")
```

SCPI

:MMEMory:CATalog? <string 1>

{string 2}<newline><^END>

"{used_size},{free_size},{name 1},,{size 1}, ... ,{name N},,{size N}"

Nn1N

{used_size}:

.

{free_size}:

.

n

{size n}:

n0

```
10 DIM A$[1000]
20 OUTPUT 717;":MMEM:CAT? ""\""""
30 ENTER 717;A$
```

SCPI.MMEMory.COPY

SCPI.MMEMory.COPY = *File*

“\”“/”

	File
	2 *0* *1* 0
	(Variant)
	254

```
SCPI.MMEMory.COPY = Array("test/state01.sta","d:test01.sta")
Dim File(1) As Variant
File(0) = "test/state01.sta"
File(1) = "d:test01.sta"
SCPI.MMEMory.COPY = File
```

Save/Recall > **Save State** > **File Dialog...**

/ > > .....

SCPI

:MMEMory:COPY <string 1>,<string 2>

```
10 OUTPUT 717;":MMEM:COPY
""Test1/State01.sta"","d:Test1_01.sta"""
```

SCPI.MMEMory.DELete

SCPI.MMEMory.DELete = *File*

“\”“/”

“\”

	File
	(String)
	254

```
SCPI.MMEMory.DELete = "d:\"
SCPI.MMEMory.DELete = "test/state01.sta"
```

Save/Recall > **Save State** > **File Dialog...**

/ > > .....

SCPI

:MMEMory:DELete <string>

```
10 OUTPUT 717;":MMEM:DEL ""Test1/State01.sta"""
10 OUTPUT 717;":MMEM:DEL ""D:State01.sta"""
```

SCPI.MMEMory.LOAD.ASCFactor

SCPI.MMEMory.LOAD.ASCFactor = *File*

A“.csv” SCPI.MMEMory.STORe.ASCFactor

“\”“/”

	File
	A“.csv”
	(String)
	254

```
SCPI.MMEMory.LOAD.ASCFactor = "d:\sensor01.csv"
SCPI.MMEMory.LOAD.ASCFactor = "test/sensor01.csv"
```

SCPI.MMEMory.STORe.ASCFactor

Cal > **Power Calibration** > **Sensor A Settings** > **Import from CSV File**
> > > **CSV**

SCPI

:MMEMory:LOAD:ASCFactor <string>

```
10 OUTPUT 717;":MMEM:LOAD:ASCF ""Test1/Sensor01.csv"""
10 OUTPUT 717;":MMEM:LOAD:ASCF ""D:Sensor01.csv"""
```

SCPI.MMEMory.LOAD.BSCFactor

SCPI.MMEMory.LOAD.BSCFactor = *File*

B“.csv” SCPI.MMEMory.STORe.BSCFactor

“\”“/”

	File
	B“.csv”
	(String)
	254

```
SCPI.MMEMory.LOAD.BSCFactor = "d:\sensor01.csv"
SCPI.MMEMory.LOAD.BSCFactor = "test/sensor01.csv"
```

SCPI.MMEMory.STORe.BSCFactor

Cal > **Power Calibration** > **Sensor B Settings** > **Import from CSV File**
> > > **CSV**

SCPI

:MMEMory:LOAD:BSCFactor <string>

```
10 OUTPUT 717;":MMEM:LOAD:BSCF ""Test1/Sensor01.csv"""
10 OUTPUT 717;":MMEM:LOAD:BSCF ""D:Sensor01.csv"""
```

SCPI.MMEMory.LOAD.CHANnel.STATe

SCPI.MMEMory.LOAD.CHANnel.STATe = *Register*

SCPI.MMEMory.STORe.CHANnel.STATe

	Register
	(String)
	“A”A “B”B “C”C “D”D

SCPI.MMEMory.LOAD.CHANnel.STATe = "a"

SCPI.MMEMory.STORe.CHANnel.STATe

SCPI.DISPlay.WINDow(Ch).ACTivate

Save/Recall > **Recall Channel** > **A|B|C|D**

/ > > **A|B|C|D**

SCPI

:MMEMory:LOAD:CHANnel[:STATe] {A|B|C|D}

```
10 OUTPUT 717;":MMEM:LOAD:CHAN A"
```

SCPI.MMEMory.LOAD.CKIT(*Ckit*)

SCPI.MMEMory.LOAD.CKIT(*Ckit*) = *File*

SCPI.MMEMory.STORe.CHANnel.STATe

	Ckit
	(Long)
	120
	1

	File
	(String)
	254

```
SCPI.MMEMory.LOAD.CKIT(1) = "Test1/Ckit01.ckx"
SCPI.MMEMory.LOAD.CKIT(1) = "A:\Ckit01.ckx"
```

SCPI.MMEMory.STORe.CKIT(Ckit)

Cal > **Modify Cal Kit** > **Import Cal Kit...**

> > **...**

SCPI

:MMEMory:LOAD:CKIT{[1]|2|3|4|5|6|7|8|9|10|11|12|13|14|15|16|17|18|19|2
<string>

```
10 OUTPUT 717;":MMEM:LOAD:CKIT1 ""Test1/Ckit01.ckx"""
10 OUTPUT 717;":MMEM:LOAD:LIM ""A:Ckit01.ckx"""
```

SCPI.MMEMory.LOAD.LIMit

SCPI.MMEMory.LOAD.LIMit = *File*

.csv SCPI.MMEMory.STORe.LIMit)

“\”“/”

	File
	“.csv”
	(String)
	254

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.LOAD.LIMit = "d:\limit01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.LOAD.LIMit = "test/limit01.csv"
```

SCPI.DISPlay.WINDow(Ch).ACTivate
SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.MMEMory.STORe.LIMit

Analysis > **Limit Test** > **Edit Limit Line** > **Import from CSV File**

> > > **CSV**

SCPI

:MMEMory:LOAD:LIMit <string>

```
10 OUTPUT 717;":MMEM:LOAD:LIM ""Test1/Limit01.csv"""
10 OUTPUT 717;":MMEM:LOAD:LIM ""D:Limit01.csv"""
```

SCPI.MMEMory.LOAD.PLOSs(*Pt*)

SCPI.MMEMory.LOAD.PLOSs(*Pt*) = *File*

“.csv” SCPI.MMEMory.STORe.PLOSs(Pt) (*Pt*)

“\”“/”

	File
	“.csv”
	(String)
	254

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.LOAD.PLOSs(1) = "d:\loss01.csv"
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.LOAD.PLOSs(1) = "test/loss01.csv"
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.STORe.PLOSs(Pt)

Cal > **Power Calibration** > **Loss Compen** > **Import from CSV File**
> > > **CSV**

SCPI

:MMEMory:LOAD:PLOSs{[1]|2|3|4} <string>

```
10 OUTPUT 717;":MMEM:LOAD:PLOS1 ""Test1/Loss01.csv"""
10 OUTPUT 717;":MMEM:LOAD:PLOS1 ""D:Loss01.csv"""
```

SCPI.MMEMory.LOAD.RLIMit

SCPI.MMEMory.LOAD.RLIMit = File

SCPI.DISPlay.WINDow(Ch).ACTivate .csv
SCPI.MMEMory.STORe.RLIMit
SCPI.CALCulate(Ch).PARameter(Tr).SELect)

“/”“\”

	File
	“.csv”
	(String)
	254

```
SCPI.DISPlay.WINDow(1).ACTive
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.LOAD.RIMit = "D:\Rlimit01.csv"

SCPI.DISPlay.WINDow(1).ACTive
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.LOAD.RLIMit = "test/Rlimit01.csv"
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.MMEMory.STORe.RLIMit

Analysis > **Ripple Limit** > **Edit Ripple Line** > **Import from CSV File**
> > > **CSV**

SCPI

:MMEMory:LOAD:RLIMit <string>

```
10 OUTPUT 717;":MMEM:LOAD:RLIM ""RTest1/Rlim01.csv"""
10 OUTPUT 717;":MMEM:LOAD:RLIM ""D:Rlim01.csv"""
```

SCPI.MMEMory.LOAD.SEGMent

SCPI.MMEMory.LOAD.SEGMent = *File*

.csv SCPI.MMEMory.STORe.SEGMent

“\”“/”

	File
	“.csv”
	(String)
	254

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.LOAD.SEGMent = "d:\segm01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.LOAD.SEGMent = "test/segm01.csv"
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.STORe.SEGMent

Sweep Setup > **Edit Segment Table** > **Import from CSV File**

> > **CSV**

SCPI

:MMEMory:LOAD:SEGMent <string>

```
10 OUTPUT 717;":MMEM:LOAD:SEGM ""Test1/Segm01.csv"""
10 OUTPUT 717;":MMEM:LOAD:SEGM ""D:Segm01.csv"""
```

SCPI.MMEMory.LOAD.STATe

SCPI.MMEMory.LOAD.STATe = *File*

.sta SCPI.MMEMory.STORe.STATe

“\”“/”

	File
	“.sta”
	(String)
	254

```
SCPI.MMEMory.LOAD.STATe = "d:\state01.sta"
SCPI.MMEMory.LOAD.STATe = "test/state01.sta"
```

SCPI.MMEMory.STORe.STATe

Save/Recall > **Recall State**

/ >

SCPI

:MMEMory:LOAD[:STATe] <string>

```
10 OUTPUT 717;":MMEM:LOAD ""Test1/State01.sta"""
10 OUTPUT 717;":MMEM:LOAD ""d:State01.sta"""
```

SCPI.MMEMory.MDIRectory

SCPI.MMEMory.MDIRectory = *File*

“\”“/”

	File
	(String)
	254

```
SCPI.MMEMory.MDIRectory = "d:\test"
SCPI.MMEMory.MDIRectory = "test"
```

Save/Recall > **Save State** > **File Dialog...**

/ > > .....

SCPI

:MMEMory:MDIRectory <string>

```
10 OUTPUT 717;":MMEM:MDIR ""Test1"""
10 OUTPUT 717;":MMEM:MDIR ""d:Test1"""
```

SCPI.MMEMory.STORe.ASCFactor

SCPI.MMEMory.STORe.ASCFactor = *File*

ACSV“.csv”

“\”“/”

	File
	A“.csv”
	(String)
	254

```
SCPI.MMEMory.STORe.ASCFactor = "d:\sensor01.csv"
SCPI.MMEMory.STORe.ASCFactor = "test/sensor01.csv"
```

SCPI.MMEMory.LOAD.ASCFactor

Cal > **Power Calibration** > **Sensor A Settings** > **Export to CSV File**
> > **A** > **CSV**

SCPI

:MMEMory:STORe:ASCFactor <string>

```
10 OUTPUT 717;":MMEM:STOR:ASCF ""Test1/Sensor01.csv"""
10 OUTPUT 717;":MMEM:STOR:ASCF ""D:Sensor01.csv"""
```

SCPI.MMEMory.STORe.BSCFactor

SCPI.MMEMory.STORe.BSCFactor = *File*

BCSV“.csv”

“\”“/”

	File
	B“.csv”
	(String)
	254

SCPI.MMEMory.STORe.BSCFactor = "d:\sensor01.csv"

SCPI.MMEMory.STORe.BSCFactor = "test/sensor01.csv"

SCPI.MMEMory.LOAD.BSCFactor

Cal > **Power Calibration** > **Sensor B Settings** > **Export to CSV File**

> > **B** > **CSV**

SCPI

:MMEMory:STORe:BSCFactor <string>

```
10 OUTPUT 717;":MMEM:STOR:BSCF ""Test1/Sensor01.csv"""
10 OUTPUT 717;":MMEM:STOR:BSCF ""D:Sensor01.csv"""
```

SCPI.MMEMory.STORe.CHANnel.CLEar

SCPI.MMEMory.STORe.CHANnel.CLEar

SCPI.MMEMory.STORe.CHANnel.STATe

SCPI.MMEMory.STORe.CHANnel.CLEar

SCPI.MMEMory.STORe.CHANnel.STATe

Save/Recall > **Save Channel** > **Clear States** > **OK**

/ > > > **OK**

SCPI

:MMEMory:STORe:CHANnel:CLEar

```
10 OUTPUT 717;":MMEM:STOR:CHAN:CLE"
```

SCPI.MMEMory.STORe.CHANnel.STATe

SCPI.MMEMory.STORe.CHANnel.STATe = *Register*

	Register
	(String)
	“A”A “B”B “C”C “D”D

SCPI.MMEMory.STORe.CHANnel.STATe = "a"

SCPI.MMEMory.LOAD.CHANnel.STATe

SCPI.DISPlay.WINDow(Ch).ACTivate

Save/Recall > **Save Channel** > **A|B|C|D**

/ > > **A|B|C|D**

SCPI

:MMEMory:STORe:CHANnel[:STATe] {A|B|C|D}

```
10 OUTPUT 717;":MMEM:STOR:CHAN A"
```

SCPI.MMEMory.STORe.CKIT(*Ckit*)

SCPI.MMEMory.STORe.CKIT(*Ckit*) = *File*

.ckx“\”“/”

	File
	(String)
	254

(Ckit) Val Ckit

```
SCPI.MMEMory.STORe.CKIT(1) = "d:\Ckit01.ckx"
SCPI.MMEMory.STORe.CKIT(1) = "test/trace01.csv"
```

SCPI.MMEMory.LOAD.CKIT(Ckit)

Save/Recall > **Save Trace Data**

/ >

SCPI

:MMEMory:STORe:CKIT{[1]|2|3|4|5|6|7|8|9|10|11|12|13|14|15|16|17|18|19
<string>

```
10 OUTPUT 717;":MMEM:STOR:CKIT1 ""Test1/Ckit01.ckx"""
10 OUTPUT 717;":MMEM:STOR:CKIT1 ""d:Ckit01.ckx"""
```

SCPI.MMEMory.STORe.FDATa

SCPI.MMEMory.STORe.FDATa = *File*

CSV“.csv”

“\”“/”

	File
	“.csv”
	(String)
	254

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.FDATa = "d:\trace01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.FDATa = "test/trace01.csv"
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Save/Recall > **Save Trace Data**

/ >

SCPI

:MMEMory:STORe:FDATa <string>

```
10 OUTPUT 717;":MMEM:STOR:FDAT ""Result/Trace01.csv"""
10 OUTPUT 717;":MMEM:STOR:FDAT ""D:Trace01.csv"""
```

SCPI.MMEMory.STORe.IMAGe

SCPI.MMEMory.STORe.IMAGe = *File*

LCD“.bmp”“.png”

E5071CVisual BasicVBA [Running a Program from the E5071C Measurement Screen](#)

“\”“/”

	File
	LCD“.bmp”“.png”
	(String)
	254

```
SCPI.MMEMory.STORe.IMAGe = "d:\image01.bmp"
SCPI.MMEMory.STORe.IMAGe = "test/image01.png"
```

System > **Dump Screen Image**

>

LCD **Capture (System)** LCD

LCD SCPI.MMEMory.STORe.IMAGe

SCPI

:MMEMory:STORe:IMAGe <string>

```
10 OUTPUT 717;":MMEM:STOR:IMAG ""Result/Image01.bmp"""
10 OUTPUT 717;":MMEM:STOR:IMAG ""D:Image01.png"""
```

SCPI.MMEMory.STORe.LIMit

SCPI.MMEMory.STORe.LIMit = *File*

CSV“.csv”

“\”“/”

	File
	“.csv”
	(String)
	254

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.LIMit = "d:\limit01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.LIMit = "test/limit01.csv"
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.MMEMory.LOAD.LIMit

Analysis > **Limit Test** > **Edit Limit Line** > **Export to CSV File**

> > > **CSV**

SCPI

:MMEMory:STORe:LIMit <string>

```
10 OUTPUT 717;":MMEM:STOR:LIM ""Test1/Limit01.csv"""
10 OUTPUT 717;":MMEM:STOR:LIM ""D:Limit01.csv"""
```

SCPI.MMEMory.STORe.PLOSs(*Pt*)

SCPI.MMEMory.STORe.PLOSs(*Pt*) = *File*

CSV“.csv” (*Pt*)

“\”“/”

	File
	“.csv”
	(String)
	254

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.STORe.PLOSs(1) = "d:\loss01.csv"
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.STORe.PLOSs(1) = "test/loss01.csv"
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.LOAD.PLOSs(Pt)

Cal > **Power Calibration** > **Loss Compen** > **Export to CSV File**
> > > **CSV**

SCPI

:MMEMory:STORe:PLOSs{[1]|2|3|4} <string>

```
10 OUTPUT 717;":MMEM:STOR:PLOS1 ""Test1/Loss01.csv"""
10 OUTPUT 717;":MMEM:STOR:PLOS1 ""D:Loss01.csv"""
```

SCPI.MMEMory.STORe.RLIMit

SCPI.MMEMory.STORe.RLIMit = File

SCPI.DISPlay.WINDow(Ch).ACTivate
SCPI.CALCulate(Ch).PARameter(Tr).SELectCSV

.sta“\”“/”

	File
	“.csv”
	(String)
	254

1

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.RLIMit = "D:\Rlimit01.csv"
```

2

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.RLIMit = "test/Rlimit01.csv"
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.MMEMory.LOAD.RLIMit

Analysis > **Ripple Limit** > **Edit Ripple Line** > **Export to CSV File**
> > > **CSV**

SCPI

:MMEMory:STORe:RLIMit <string>

```
10 OUTPUT 717;":MMEM:STOR:RLIM ""RTest1/Rlim01.csv"""
10 OUTPUT 717;":MMEM:STOR:RLIM ""D:Rlim01.csv"""
```

SCPI.MMEMory.STORe.SALL

SCPI.MMEMory.STORe.SALL = *Status*

Status = SCPI.MMEMory.STORe.SALL

///

	Status
	(Boolean)
	• / • /

```
Dim Obj As Boolean
SCPI.MMEMory.STORe.SALL = True
Obj = SCPI.MMEMory.STORe.SALL
```

SCPI.MMEMory.STORe.STATe

Save/Recall > **Channel/Trace**

/ > **/**

SCPI

:MMEMory:STORe:SALL {ON|OFF|1|0}

:MMEMory:STORe:SALL?

{1|0}<newline><^END>

```
10 OUTPUT 717;":MMEM:STOR:SALL ON"
20 OUTPUT 717;":MMEM:STOR:SALL?"
30 ENTER 717;A
```

SCPI.MMEMory.STORe.SEGMent

SCPI.MMEMory.STORe.SEGMent = *File*

CSV“.csv”

“\”“/”

	File
	“.csv”
	(String)
	254

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.STORe.SEGMent = "d:\segm01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.STORe.SEGMent = "test/segm01.csv"
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.LOAD.SEGMent

Sweep Setup > **Edit Segment Table** > **Export to CSV File**

> > **CSV**

SCPI

:MMEMory:STORe:SEGMent <string>

```
10 OUTPUT 717;":MMEM:STOR:SEGM ""Test1/Segm01.csv"""
10 OUTPUT 717;":MMEM:STOR:SEGM ""D:Segm01.csv"""
```

SCPI.MMEMory.STORe.SNP.DATA

SCPI.MMEMory.STORe.SNP.DATA = *File*

SCPI.DISPlay.WINDow(Ch).ACTivate

<>	<>
1	s1p
2	s2p
3	s3p
4	s4p

“/”“\”

	File
	254

```
Dim SnpPorts(2) As Variant
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.STORe.SNP.FORMat = "RI"
SnpPorts(0) = 1
SnpPorts(1) = 2
SnpPorts(2) = 4
SCPI.MMEMory.STORe.SNP.TYPE.S3P = SnpPorts
SCPI.MMEMory.STORe.SNP.DATA = "SNP01.s3p"
```

SCPI.DISPlay.WINDow(Ch).ACTivate
SCPI.MMEMory.STORe.SNP.FORMat
SCPI.MMEMory.STORe.SNP.TYPE.S1P
SCPI.MMEMory.STORe.SNP.TYPE.S2P
SCPI.MMEMory.STORe.SNP.TYPE.S3P
SCPI.MMEMory.STORe.SNP.TYPE.S4P

SCPI

:MMEMory:STORe:SNP[:DATA] <string>

```
10 OUTPUT 717;"DISP:WIND1:ACT"
20 OUTPUT 717;":MMEM:STOR:SNP:FORM RI"
30 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S3P 1,2,4"
40 OUTPUT 717;":MMEM:STOR:SNP ""SNP01.s3p"""
```

SCPI.MMEMory.STORe.SNP.FORMat

SCPI.MMEMory.STORe.SNP.FORMat = *Param*

Param = SCPI.MMEMory.STORe.SNP.FORMat

/ SCPI.DISPlay.WINDow(Ch).ACTivate

	Param
	(String)
	“AUTO” “MA”“ > ” “DB”“ > ” “RI”“ > ”
	“AUTO”

```
Dim Fmt As String
SCPI.MMEMory.STORe.SNP.FORMat = "MA"
Fmt = SCPI.MMEMory.STORe.SNP.FORMat
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.STORe.SNP.DATA

Save/Recall > **Save Snp > Snp Format > AUTO|LogMag/Angle|LinMag/Angle|Real/Imaginary**

/ > Snp > Snp > AUTO|/|/|/

SCPI

:MMEMory:STORe:SNP:FORMat {AUTO|MA|DB|RI}

:MMEMory:STORe:SNP:FORMat?

{AUTO|MA|DB|RI}<newline><^END>

```
10 OUTPUT 717;":MMEM:STOR:SNP:FORM MA"
20 OUTPUT 717;":MMEM:STOR:SNP:FORM?"
30 ENTER 717;A$
```

SCPI.MMEMory.STORe.SNP.TYPE.S1P

SCPI.MMEMory.STORe.SNP.TYPE.S1P = *Port*

Port = SCPI.MMEMory.STORe.SNP.TYPE.S1P

SCPI.DISPlay.WINDow(Ch).ACTivate/1

	Port
	14
	1

```
10 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S1P 2"
20 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S1P?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(Ch).ACTivate
SCPI.MMEMory.STORe.SNP.DATA
SCPI.MMEMory.STORe.SNP.FORMat

Save/Recall > **Save SnP** > **S1P** > **1|2|3|4**

/ > **SnP** > **S1P > 1|2|3|4**

SCPI

:MMEMory:STORe:SNP:TYPE:S1P <numeric>

:MMEMory:STORe:SNP:TYPE:S1P?

```
10 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S1P 2"
20 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S1P?"
30 ENTER 717;A$
```

SCPI.MMEMory.STORe.SNP.TYPE.S2P

SCPI.MMEMory.STORe.SNP.TYPE.S2P = *Ports*

Ports = SCPI.MMEMory.STORe.SNP.TYPE.S2P

SCPI.DISPlay.WINDow(Ch).ACTivate/2

	Ports
	2 • *Ports(0)* • *Ports(1)* 0
	(Variant)
	14
	1

```
Dim Ports(1) As Long
Ports(0) = 2
Ports(1) = 3
SCPI.MMEMory.STORe.SNP.TYPE.S2P = Ports
Ports = SCPI.MMEMory.STORe.SNP.TYPE.S2P
```

SCPI.DISPlay.WINDow(Ch).ACTivate
SCPI.MMEMory.STORe.SNP.DATA
SCPI.MMEMory.STORe.SNP.FORMat

Save/Recall > **Save Snp** > **S2p** > **1-2**|**1-3**|**1-4**|**2-3**|**2-4**|**3-4**

/ > **Snp** > **S2p** > **1-2**|**1-3**|**1-4**|**2-3**|**2-4**|**3-4**

SCPI

:MMEMory:STORe:SNP:TYPE:S2P <numeric1>, <numeric 2>

:MMEMory:STORe:SNP:TYPE:S2P?

```
10 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S2P 2,3"
20 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S2P?"
30 ENTER 717;A$
```

SCPI.MMEMory.STORe.SNP.TYPE.S3P

SCPI.MMEMory.STORe.SNP.TYPE.S3P = *Ports*

Ports = SCPI.MMEMory.STORe.SNP.TYPE.S3P

SCPI.DISPlay.WINDow(Ch).ACTivate/3

	Ports
	3 • *Ports(0)* • *Ports(1)* • *Ports(2)* 0
	(Variant)
	14
	1

```
Dim Ports(2) As Long
Ports(0) = 2
Ports(1) = 3
Ports(2) = 4
SCPI.MMEMory.STORe.SNP.TYPE.S3P = Ports
Ports = SCPI.MMEMory.STORe.SNP.TYPE.S3P
```

SCPI.DISPlay.WINDow(Ch).ACTivate
SCPI.MMEMory.STORe.SNP.DATA
SCPI.MMEMory.STORe.SNP.FORMat

Save/Recall > **Save Snp** > **S3p** > **1-2-3**|**1-2-4**|**1-3-4**|**2-3-4**

/ > **Snp** > **S3p** > **1-2-3**|**1-2-4**|**1-3-4**|**2-3-4**

SCPI

:MMEMory:STORe:SNP:TYPE:S3P <numeric 1>, <numeric 2>, <numeric 3>

:MMEMory:STORe:SNP:TYPE:S3P?

```
10 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S3P 2,3,4"
20 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S3P?"
30 ENTER 717;A$
```

SCPI.MMEMory.STORe.SNP.TYPE.S4P

SCPI.MMEMory.STORe.SNP.TYPE.S4P = *Ports*

Ports = SCPI.MMEMory.STORe.SNP.TYPE.S4P

SCPI.DISPlay.WINDow(Ch).ACTivate/4

	Ports
	4 • *Ports(0)* • *Ports(1)* • *Ports(2)* • *Ports(3)* 0
	(Variant)
	14
	1

```
Dim Ports(3) As Long
Ports(0) = 1
Ports(1) = 2
Ports(2) = 3
Ports(3) = 4
SCPI.MMEMory.STORe.SNP.TYPE.S4P = Ports
Ports = SCPI.MMEMory.STORe.SNP.TYPE.S4P
```

SCPI.DISPlay.WINDow(Ch).ACTivate
SCPI.MMEMory.STORe.SNP.DATA
SCPI.MMEMory.STORe.SNP.FORMat

Save/Recall > **Save Snp** > **S4p** > **1-2-3-4**

/ > **Snp** > **S4p** > **1-2-3-4**

SCPI

:MMEMory:STORe:SNP:TYPE:S4P 1, 2, 3, 4

:MMEMory:STORe:SNP:TYPE:S4P?

```
10 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S4P 1,2,3,4"
20 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S3P?"
30 ENTER 717;A$
```

SCPI.MMEMory.STORe.STATe

SCPI.MMEMory.STORe.STATe = *File*

SCPI.MMEMory.STORe.STYPe.sta

“\”“/”

	File
	“.sta”
	(String)
	254

```
Dim StaType As String
SCPI.MMEMory.STORe.STYPe = "cdst"
SCPI.MMEMory.STORe.STATe = "d:\state01.sta"

Dim StaType As String
SCPI.MMEMory.STORe.STYPe = "cdst"
SCPI.MMEMory.STORe.STATe = "test/state01.sta"
```

SCPI.MMEMory.STORe.STYPe

SCPI.MMEMory.LOAD.STATe

Save/Recall > **Save State**

/ >

SCPI

:MMEMory:STORe[:STATe] <string>

```
10 OUTPUT 717;":MMEM:STOR ""Test1/State01.sta"""
10 OUTPUT 717;":MMEM:STOR ""D:State01.sta"""
```

SCPI.MMEMory.STORe.STYPe

SCPI.MMEMory.STORe.STYPe = *Param*

Param = SCPI.MMEMory.STORe.STYPe

SCPI.MMEMory.STORe.STATe

	Param
	(String)
	• “STATe” • “CSTate” • “DSTate” • “CDSTate”
	“CSTate”

```
Dim StaType As String
SCPI.MMEMory.STORe.STYPe = "cdst"
StaType = SCPI.MMEMory.STORe.STYPe
```

SCPI.MMEMory.STORe.STATe

Save/Recall > **Save Type** > **State Only|State & Cal|State & Trace|All**

/ > > **|||**

SCPI

:MMEMory:STORe:STYPe {STATe|CSTate|DSTate|CDSTate}

:MMEMory:STORe:STYPe?

{STAT|CST|DST|CDST}<newline><^END>

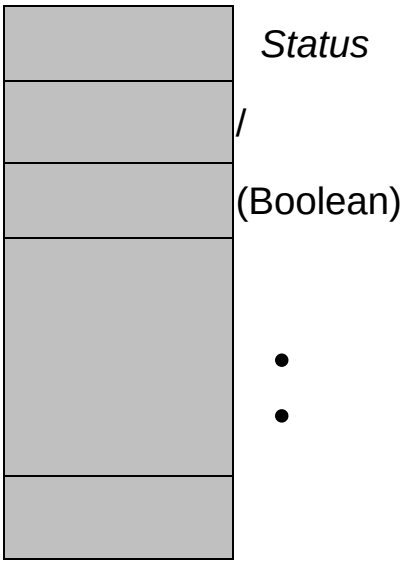
```
10 OUTPUT 717;":MMEM:STOR:STYP CDST"
20 OUTPUT 717;":MMEM:STOR:STYP?"
30 ENTER 717;A$
```

SCPI.OUTPut.STATe

SCPI.OUTPut.STATe = *Status*

Status = SCPI.OUTPut.STATe

/


Status
/
(Boolean)

```
Dim Outp As Boolean
SCPI.OUTPut.STATe = True
Outp = SCPI.OUTPut.STATe
```

Sweep Setup > **Power** > **RF Out**

> >

SCPI

:OUTPut[:STATe] {ON|OFF|1|0}

:OUTPut[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":OUTP ON"
20 OUTPUT 717;":OUTP?"
30 ENTER 717;A
```

:PROGram:CATalog?

COM

:PROGram:CATalog?

VBAVBAVBA

{string}<newline><^END>

"{macro 1},{macro 2}, ... ,{macro N}"
NVBA
{macro n}: VBA macro name (module name.procedure name)

```
10 DIM A$[1000]
20 OUTPUT 717;":PROG:CAT?"
30 ENTER 717;A$
```

:PROGram[:SELected]:NAME

COM

:PROGram[:SELected]:NAME <string>

:PROGram[:SELected]:NAME?

/:PROG:STATVBA

:PROG:CAT?VBA

	String
	VBA
	254
	“”

{string}<newline><^END>

```
10 OUTPUT 717;":PROG:NAME ""Module1.main"""
20 OUTPUT 717;":PROG:NAME?"
30 ENTER 717;A$
```

:PROG:CAT?

:PROG:STAT

Macro Setup > **Select Macro**

>

VBA

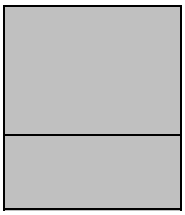
:PROGram[:SELected]:STATe

COM

:PROGram[:SELected]:STATe {STOP|RUN}

:PROGram[:SELected]:STATe?

/:PROG:STATVBA/



{STOP|RUN}<newline><^END>

```
10 OUTPUT 717;":PROG:STAT RUN"
20 OUTPUT 717;":PROG:STAT?"
30 ENTER 717;A$
```

:PROG:NAME

Macro Break (to stop)

Macro Setup > **Select Macro** (to run)

>

VBA

SCPI.SENSe(*Ch*).AVERage.CLEar

SCPI.SENSe(*Ch*).AVERage.CLEar

0 *(Ch)*

SCPI.SENSe(1).AVERage.CLEar

SCPI.SENSe(Ch).AVERage.COUNt

SCPI.SENSe(Ch).AVERage.STATe

Avg > Averaging Restart

>

SCPI

:SENSe{[1]-36}:AVERage:CLEar

```
10 OUTPUT 717;":SENS1:AVER:CLE"
```

SCPI.SENSe(*Ch*).AVERage.COUNt

SCPI.SENSe(*Ch*).AVERage.COUNt = *Value*

Value = SCPI.SENSe(*Ch*).AVERage.COUNt

/ (*Ch*)

	Value
	Long
	1999
	16

```
Dim AvgCnt As Long
SCPI.SENSe(1).AVERage.COUNt = 4
AvgCnt = SCPI.SENSe(1).AVERage.COUNt
```

SCPI.SENSe(Ch).AVERage.STATe

SCPI.SENSe(Ch).AVERage.CLEar

Avg > **Avg Factor**

>

SCPI

:SENSe{[1]-36}:AVERage:COUNt <numeric>

:SENSe{[1]-36}:AVERage:COUNt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:AVER:COUN 4"
20 OUTPUT 717;":SENS1:AVER:COUN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).AVERage.STATe

SCPI.SENSe(*Ch*).AVERage.STATe = *Status*

Status = SCPI.SENSe(*Ch*).AVERage.STATe

/ (Ch)

	Status
	/
	Boolean
	• •

```
Dim Avg As Boolean
SCPI.SENSe(1).AVERage.STATe = True
Avg = SCPI.SENSe(1).AVERage.STATe
```

SCPI.SENSe(Ch).AVERage.COUNt

SCPI.SENSe(Ch).AVERage.CLEar

Avg > **Averaging**

>

SCPI

:SENSe{[1]-36}:AVERage[:STATe] {ON|OFF|1|0}

:SENSe{[1]-36}:AVERage[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:AVER ON"
20 OUTPUT 717;":SENS1:AVER?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).BANDwidth.RESolution

SCPI.SENSe(*Ch*).BANDwidth.RESolution = *Value*

Value = SCPI.SENSe(*Ch*).BANDwidth.RESolution

/ (*Ch*)

SCPI.SENSe(Ch).BWIDth.RESolution

	Value
	Double
	10500000
	70000
	Hz
	11.523457

```
Dim IfBw As Double
SCPI.SENSe(1).BANDwidth.RESolution = 1.5E3
IfBw = SCPI.SENSe(1).BANDwidth.RESolution
```

SCPI.SENSe(Ch).BWIDth.RESolution

Avg > **IF Bandwidth**

>

SCPI

:SENSe{[1]-36}:BANDwidth[:RESolution] <numeric>

:SENSe{[1]-36}:BANDwidth[:RESolution]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:BAND 1.5E3"
20 OUTPUT 717;":SENS1:BAND?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).BWIDth.RESolution

SCPI.SENSe(*Ch*).BWIDth.RESolution = *Value*

Value = SCPI.SENSe(*Ch*).BWIDth.RESolution

/ (*Ch*)

SCPI.SENSe(Ch).BANDwidth.RESolution

	Value
	Double
	10500000
	70000
	Hz
	11.523457

```
Dim IfBw As Double
SCPI.SENSe(1).BWIDth.RESolution = 1.5E3
IfBw = SCPI.SENSe(1).BWIDth.RESolution
```

SCPI.SENSe(Ch).BANDwidth.RESolution

Avg > **IF Bandwidth**

>

SCPI

:SENSe{[1]-36}:BWIDth[:RESolution] <numeric>

:SENSe{[1]-36}:BWIDth[:RESolution]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:BWID 1.5E3"
20 OUTPUT 717;":SENS1:BWID?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.CLEar

SCPI.SENSe(*Ch*).CORRection.CLEar

(*Ch*)

SCPI.SENSe(1).CORRection.CLEar

SCPI.SENSe(Ch).OFFSet.STATe

SCPI.SENSe(Ch).CORRection.OFFSet.CLEar

Cal > **Clear()** > **OK**

> >

SCPI

:SENSe{[1]-36}:CORRection:CLEar

```
10 OUTPUT 717;":SENS1:CORR:CLE"
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

( )

SCPI.SENSe(Ch).CORRection.COEFficient.DATA (Str, Int1 , Int2) = Array

Array = SCPI.SENSe(Ch).CORRection.COEFficient.DATA (Str, Int1, Int2)

/

SCPI.SENSe(Ch).CORRection.CC

	Array
	NOPX2 n1NOP *Data(nX2-2)* n *Data(nX2-1)* n 0
	Variant

	Str
	String
	• “ES” • “ER” • “ED” • “EL” • “ET” • “EX”

	Int1

	Integer
	14
	ESEREDELETEX

	Int2
	Integer
	14
	ESEREDELETEX

(*Ch*) Ch

```
DIM Array(200) As Variant
Array = SCPI.SENs(1).CORRection.COEFficient("EL", 1, 2)
```

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.ERESponse
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.OPEN
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.SHORt
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.THRU
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT1
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT2
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT3
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT4
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL2
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL3
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL4

SCPI

:SENSe{[1]-36}:CORRection:COEFficient[:DATA] {ES|ER|ED|EL|ET|EX}, <numeric 1>, <numeric 2>, <numeric 3>,..., <numeric 3 n*2>

:SENSe{[1]-36}:CORRection:COEFficient[:DATA]? {ES|ER|ED|EL|ET|EX},<numeric 1>, <numeric 2>

{numeric 1}, ... ,{numeric NOP*2}<newline><^END>

{numeric nX 2-1}	n
{numeric nX 2}	n

NOPn1NOP

```
10 DIM A(1:201)
20 OUTPUT 717;":SENS1:CORR:COEF? EL,1,2"
30 ENTER 717;A(*)
```

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.ERESpc

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.ERESponse = Ports

SCPI.SENSe(Ch).CORRection.COEFficient.DATACh

	Ports
	2 • *Ports(0)* • *Ports(1)* 0
	Variant
	14
	1

Ch Ch

```
Dim ERESPort(1) As Variant
ERESPort(0) = 1
ERESPort(1) = 2
SCPI.SENSe(1).CORRection.COEFficient.METHod.ERESponse =
ERESPort
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

SCPI

:SENSe{[1]-36}:CORRection:COEFficient:METHod:ERESponse
<numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:ERES 1,2"
```

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPor

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.OPEN = Port

SCPI.SENSe(Ch).CORRection.COEFficient.DATACh

ChPort Val Ch (Ch) Val Port

SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.OPEN
= 1

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

SCPI

:SENSe{[1]-36}:CORRection:COEFficient:METHod[:RESPonse]:OPEN <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:OPEN 1"
```

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPor

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.SHORt = Port

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

ChPort Val Ch (Ch) Val Port

SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.SHORt = 1

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

SCPI

:SENSe{[1]-36}:CORRection:COEFficient:METHod[:RESPonse]:SHORt <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:SHOR 1"
```

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPor

)

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.THRU = Ports

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

ChPort Val Ch (Ch) Val Port

1

```
SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.THRU
= Array(2, 1)
```

2

```
Dim ThruPort(1) As Variant
ThruPort(0) = 2
ThruPort(1) = 1
SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.THRU
= ThruPort
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

SCPI

:SENSe{[1]-36}:CORRection:COEFficient:METHod[:RESPonse]:THRU <numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:THRU 2,1"
```

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT1

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT1 = Port

SCPI.SENSe(Ch).CORRection.COEFficient.DATA1

ChPort Val Ch (Ch) Val Port

SCPI.SENSe(1).CORREction.COEFficient.METHod.SLOT1 = 1

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

SCPI

:SENSe{[1]-36}:CORRection:COEFficient:METHod:SOLT1 <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:SOLT1 1"
```

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT2

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT2 = Ports

SCPI.SENSe(Ch).CORRection.COEFficient.DATA2

	Ports
	2 • *Ports(0)* • *Ports(1)* 0
	Variant
	14
	1

Ch Ch

1

```
SCPI.SENSe(1).CORRection.COEFficient.METHod.SOLT2 = Array(1,
2)
```

2

```
Dim CalPort(1) As Variant
CalPort(0) = 1
CalPort(1) = 2
SCPI.SENSe(1).CORRection.COEFficient.METHod.SOLT2 = CalPort
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

SCPI

:SENSe{[1]-36}:CORRection:COEFficient:METHod:SOLT2 <numeric 1>, <numeric 2>

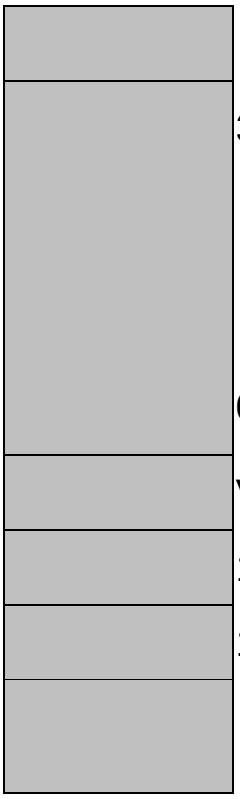
```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:SOLT2 1,2"
```

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT3

(

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT3 = Ports

SCPI.SENSe(Ch).CORRection.COEFficient.DATA3



Ports

3

- *Ports(0)* 3
- *Ports(1)* 3
- *Ports(2)* 3

0

Variant

14

1

Ch Ch

1

```
SCPI.SENSe(1).CORRection.COEFficient.METHod.SOLT3 = Array(1,
2, 3)
```

2

```
Dim CalPort(2) As Variant
CalPort(0) = 1
CalPort(1) = 2
CalPort(2) = 3
SCPI.SENSe(1).CORRection.COEFficient.METHod.SOLT3 = CalPort
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

SCPI

:SENSe{[1]-36}:CORRection:COEFficient:METHod:SOLT3 <numeric 1>, <numeric 2>, <numeric 3>

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:SOLT3 1,2,3"
```

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT4

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT4 = Ports

SCPI.SENSe(Ch).CORRection.COEFficient.DATA4

	Ports
	4 • *Ports(0)*4 • *Ports(1)*4 • *Ports(2)*4 • *Ports(3)*4 0
	Variant
	14
	1

Ch Ch

1

```
SCPI.SENSe(1).CORRection.COEFficient.METHod.SOLT4 = Array(1,
2, 3, 4)
```

2

```
Dim CalPort(3) As Variant
CalPort(0) = 1
CalPort(1) = 2
CalPort(2) = 3
CalPort(3) = 4
SCPI.SENSe(1).CORRection.COEFfient.METHod.SLOT4 = CalPort
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

SCPI

:SENSe{[1]-36}:CORRection:COEFficient:METHod:SOLT4 <numeric 1>, <numeric 2>, <numeric 3>, <numeric 4>

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:SOLT4 1,2,3,4"
```

SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

Ch Ch

```
Dim Dmy As Long
Dim Data(3) as Variant
Data(0) = -1.123
Data(1) = 2.456
Data(2) = -2.249
Data(3) = 2.608
SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.THRU
= Array(2, 1)
SCPI.SENSe(1).CORRection.COEFficient("ET", 2, 1) = Data
Dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(1).CORRection.COEFficient.SAVE
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.ERESponse
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.OPEN
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.SHORt
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.THRU
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT1
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT2
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT3
SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT4
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL2
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL3
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL4

SCPI

:SENSe{[1]-36}:CORRection:COEFficient:SAVE

```
10 OUTPUT 717;":SENS1:CORR:COEF:SAVE"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.ISOLation

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.ISOLation = *Ports*

(*Ch*)

	Ports
	2 • *Ports(0)* • *Ports(1)* 0
	Variant
	14
	1
	2

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.ISOLation = Array(1,2)
Dmy = SCPI.IEEE4882.OPC

Dim IsPort(1) As Variant
Dim Dmy As Long
IsPort(0) = 1
IsPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.ACQuire.ISOLation = IsPort
Dmy = SCPI.IEEE4882.OPC
```

SCPI.IEEE4882.OPC

Cal > **Calibrate** > **Response (Thru)** > **Isolation (Optional)**

> > **>**

Cal(> **Calibrate** > **n-Port Caln** > **Isolation (Optional)**
> **Port m-n Isolm-n**

> > **n** > > **m-n**

SCPI

:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:ISOLation <numeric 1>, <numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ISOL 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.LOAD

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.LOAD = *Port*

(*Ch*)

	Port
	Long
	1 4

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.LOAD = 1
Dmy = SCPI.IEEE4882.OPC
```

SCPI.IEEE4882.OPC

Cal > **Calibrate** > **Response (Open)|Response (Short)** > **Load (Optional)**

> > > **>**

Cal > **Calibrate** > **1-Port Cal** > **Load**

> > **1** >

Cal > **Calibrate** > **n-Port Cal** > **Reflection** > **Port m Load**

> > **n** > > **m**

SCPI

:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:LOAD <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COLL:LOAD 1"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.OPEN

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.OPEN = *Port*

(*Ch*)

ChPort ChPort

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.OPEN = 1
Dmy = SCPI.IEEE4882.OPC
```

SCPI.IEEE4882.OPC

Cal > **Calibrate** > **Response (Open)|1-Port Cal** > **Open**
> > > **1** >

Cal > **Calibrate** > **n-Port Cal** > **Reflection** > **Port m Open**
> > **n** > > **m**

SCPI

:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:OPEN <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COLL:OPEN 1"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.SHORt

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.SHORt = *Port*

(*Ch*)

ChPort ChPort

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.SHORt = 1
Dmy = SCPI.IEEE4882.OPC
```

SCPI.IEEE4882.OPC

Cal > **Calibrate** > **Response (Short)|1-Port Cal** > **Short**
> > > **1** >

Cal > **Calibrate** > **n-Port Cal** > **Reflection** > **Port m Short**
> > **n** > > **m**

SCPI

:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:SHORt <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COLL:SHOR 1"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.SUBClass

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.SUBClass = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.SUBClass

(Ch)///

	Value
	Long
	18
	1
	1

```
Dim Subc As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.SUBClass = 3
Subc = SCPI.SENSe(1).CORRection.COLLect.ACQuire.SUBClass
```

SCPI

:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:SUBClass <numeric>
:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:SUBClass?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:SUBC 1"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.THRU

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.THRU = *Ports*

(*Ch*)

ChPort ChPort

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.THRU = Array(2,1)
Dmy = SCPI.IEEE4882.OPC
```

```
Dim ThruPort(1) As Variant
Dim Dmy As Long
ThruPort(0) = 2
ThruPort(1) = 1
SCPI.SENSe(1).CORRection.COLLect.ACQuire.THRU = ThruPort
Dmy = SCPI.IEEE4882.OPC
```

SCPI.IEEE4882.OPC

Cal > **Calibrate** > **Response (Thru)** > **Thru**

> > >

Cal > **Calibrate** > **n-Port Cal** > **Transmission** > **Port m-n Thru**

> > **n** > > **m-m**

SCPI

:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:THRU <numeric 1>, <numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:THRU 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.TRLLine

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.TRLLine = *Ports*

(*Ch*)TRL

	Ports
	2 • *Ports(0)* • *Ports(1)* 0
	Variant
	1 4
	1

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.TRLLine = Array(1,2)
Dmy = SCPI.IEEE4882.OPC

Dim Trll(1) As Variant
Dim Dmy As Long
Trll(0) = 1
Trll(1) = 2
SCPI.SENSe(1).CORRection.COLLect.ACQuire.TRLLine = Trll
Dmy = SCPI.IEEE4882.OPC
```

SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRLReflect
SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRLThru

Cal > **Calibrate** > **2-Port TRL Cal** > **Line/Match** > **x-y Line/Match|x-y Fwd (Syx)|x-y Rvs (Sxy)**

> > 2TRL > / > x-y/|x-ySyx|x-ySxy

Cal > **Calibrate** > **3-Port TRL Cal** > **Line/Match** > **x-y Line/Match|x-y Fwd (Syx)|x-y Rvs (Sxy)|x-z Line/Match|x-z Fwd (Szx)|x-z Rvs (Sxz)|y-z Line/Match|y-z Fwd (Szy)|y-z Rvs (Syz)**

> > 3TRL > / > x-y/|x-ySyx|x-ySxy|x-z/|x-z Szx|x-zSxz|y-z/|y-zSzy|y-zSyz

Cal > **Calibrate** > **4-Port TRL Cal** > **Line/Match** > **x-y Line/Match|x-y Fwd (Syx)|x-y Rvs (Sxy)** > **x-z Line/Match |x-z Fwd (Szx)|x-z Rvs (Sxz)** > **x-w Line/Match|x-w Fwd (Swx)|x-w Rvs (Sxw)** > **y-z Line/Match|y-z Fwd (Szy)|y-z Rvs (Syz)** > **y-w Line/Match|y-w Fwd (Swy)|y-w Rvs (Syw)** > **z-w Line/Match|z-w Fwd (Swz)|z-w Rvs (Szw)**

> > 4TRL > / > x-y/|x-ySyx|x-ySxy>x-z/|x-z Szx|x-zSxz> x-w/|x-wSwx|x-wSxw> y-z/|y-z Szy|y-zSyz> y-w/|y-wSwy|y-wSyw> z-w/|z-w Swz|z-wSzw

SCPI

:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:TRLLine <numeric 1>, <numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:TRLL 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.TRLReflect

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.TRLReflect = *Port*

(*Ch*)TRL

	Port
	Long
	14
	1

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.TRLReflect = 1
Dmy = SCPI.IEEE4882.OPC
```

SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRLLine
SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRLThru

Cal > **Calibrate** > **2-Port TRL Cal** > **Reflect** > **Portx Reflect|Porty Reflect**

> > **2TRL** > > **x|y**

Cal > **Calibrate** > **3-Port TRL Cal** > **Reflect** > **Portx Reflect|Porty Reflect|Portz Reflect**

> > **3TRL** > > **x|y|z**

Cal > **Calibrate** > **4-Port TRL Cal** > **Reflect** > **Portx Reflect|Porty Reflect|Portz Reflect|Portw Reflect**

> >**4TRL** > > **x** **|y|z|w**

SCPI

:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:TRLReflect <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COLL:TRLR 1"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.TRLThru

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.TRLThru = *Ports*

(*Ch*)TRL

	Ports
	2 • *Ports(0)* • *Ports(1)* 0
	Variant
	14
	1

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.TRLThru = Array(1,2)
Dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(1).CORRection.COLLect.ACQuire.TRLThru = Array(2,1)
Dmy = SCPI.IEEE4882.OPC

Trlt1(0) = 1
Trlt1(1) = 2
Trlt2(0) = 2
Trlt2(1) = 1
SCPI.SENSe(1).CORRection.COLLect.ACQuire.TRLThru = Trlt1
Dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(1).CORRection.COLLect.ACQuire.TRLThru = Trlt2
Dmy = SCPI.IEEE4882.OPC
```

SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRLLine
SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.TRLReflect

Cal > **Calibrate** > **2-Port TRL Cal** > **Thru/Line** > **Port x-y Thru**

>>**2TRL**>/>**x-y**

Cal > **Calibrate** > **3-Port TRL Cal** > **Thru/Line** > **Port x-y Thru|Port x-z Thru|Port y-z Thru**

>>**3TRL**>/>**x-y|x-z|y-z**

Cal > **Calibrate** > **4-Port TRL Cal** > **Thru/Line** > **Port x-y Thru|Port x-z Thru|Port x-w Thru|Port y-z Thru|Port y-w Thru|Port z-w Thru**

>>**4TRL**>/>**x-y|x-z|x-w|y-z|y-w|z-w**

SCPI

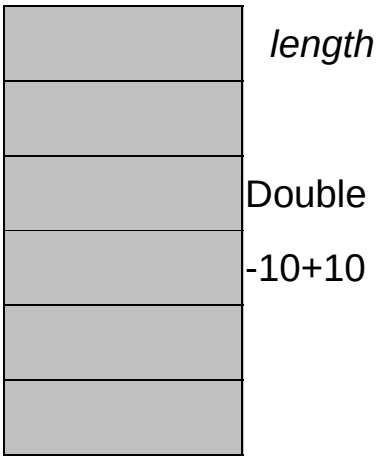
:SENSe{[1]-36}:CORRection:COLLect[:ACQuire]:TRLThru <numeric 1>, <numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:TRLT 1,2"
20 OUTPUT 717;"*OPC?"
30 OUTPUT 717;":SENS1:CORR:COLL:TRLT 2,1"
40 OUTPUT 717;"*OPC?"
50 ENTER 717;A
```

SCPI.SENSe(Ch).CORRection.COLLect.ADAPter(Pt).LENGth

SCPI.SENSe(Ch).CORRection.COLLect.ADAPter(Pt).LENGth=*length*

(Ch)/


length
Double
-10+10

SCPI.SENSe(1).CORRection.COLLect.ADAPter(2).LENGth = 0.01

SCPI.SENSe(Ch).CORRection.COLLect.METHod.ADAPter.REMoval

Cal > **Calibrate** > **Adapter Removal** > **Adapter Length**

> > >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ADAPter{[1]-4}:LENGth <+ or ->
<value of length>

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ADAP2:LENG 0.01"
20 OUTPUT 717;":SENS1:CORR:COLL:ADAP2:LENG?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.LABel

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.LABel = *Lbl*

Lbl = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.LABel

136*(Ch)*/

	Lbl
	String
	254
	• 1“85033E” • 2“85033D” • 3“85052D” • 4“85032F” • 5“85032B” • 6“85036B/E” • 7“85031B” • 8“85050C/D” • 9“85052C” • 1020““

```
Dim CalLbl As String
SCPI.SENSe(1).CORRection.COLLect.CKIT.LABel = "User 1"
CalLbl = SCPI.SENSe(1).CORRection.COLLect.CKIT.LABel
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Label Kit**

> >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:LABel <string>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:LABel?

{string}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:LAB ""MY_KIT"""
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:LAB?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.LOAD(C

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.LOAD(*Cpt*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.LOAD(*Cpt*)

(*Ch*)/ (*Cpt*)

	Cpt
	Long
	14

Cpt *Cpt*

	Value
	Long
	021

```
Dim StanLoad As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.LOAD(1) = 10
StanLoad =
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.LOAD(1)
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLLine(Cpt_m,Cp

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLReflect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLThru(Cpt_m,Cp

Cal > **Modify Cal Kit** > **Specify CLSs** > **Load** > **Set All|Port 1|Port 2|Port 3|Port 4**

> > **CLS** > > **|1|2|3|4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:LOAD <numeric 1>, <numeric 2>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:LOAD? <numeric 1>

{numeric 2}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:LOAD 1,9"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:LOAD? 1"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.OPEN(C

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.OPEN(*Cpt*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.OPEN(*Cpt*)

(*Ch*) / (*Cpt*)

	Value
	Long
	021

Cpt *Cpt*

```
Dim StanOpen As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.OPEN(1) = 10
StanOpen =
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.OPEN(1)
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLLine(Cpt_m,Cp

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLReflect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLThru(Cpt_m,Cp

Cal > **Modify Cal Kit** > **Specify CLSs** > **Open** > **Port 1|Port 2|Port 3|Port 4**

> > **CLS** > > **1 |2|3|4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:OPEN <numeric 1>, <numeric 2>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:OPEN? <numeric 1>

{numeric 2}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:OPEN 1,2"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:OPEN? 1"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.SELect

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.SELect = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.SELect

(*Ch*) / (*Cpt*)

	Value
	Long
	18
	1

```
Dim CKitSel As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.SELect = 3
CKitSel = SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.SELect
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLLine(Cpt_m,Cp

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLReflect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLThru(Cpt_m,Cp

Cal > **Modify Cal Kit** > **Specify CLSs** > **Sub Class** > **Sub Class 1| ... |Sub Class 8**

> > **CLS** > > **1**| … |**8**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:SELect <numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:SELect?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:SEL 1"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:SEL?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.SHORt(

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.SHORt(*Cpt*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.SHORt(*Cpt*)

(*Ch*) / (*Cpt*)

	Value
	Long
	121

Cpt *Cpt*

```
Dim StanShor As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.SHORt(1) = 10
StanShor =
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.SHORt(1)
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLLine(Cpt_m,Cp

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLReflect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLThru(Cpt_m,Cp

Cal > **Modify Cal Kit** > **Specify CLSs** > **Short** > **Port 1|Port 2|Port 3|Port 4**

> > **CLS** > > **1|2|3|4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:SHORt <numeric 1>, <numeric 2>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:SHORt? <numeric 1>

{numeric 2}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:SHOR 1,1"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:SHOR? 1"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.THRU(*C*

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.THRU(*Cpt_m*,*Cpt_*
= *Value*

Value =
SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.THRU(*Cpt_m*,*Cpt_*

(*Ch*)/2 (*Cpt_mCpt_n*)

	Cpt_m, *Cpt_n*
	Long
	14

Cpt_mCpt_n *Cpt_mCpt_n*2 *Cpt_mCpt_n*2
2

	Value
	Long
	021

```
Dim StanThru As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.THRU(1,2) = 10
StanThru =
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.THRU(1,2)
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SELect
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLLine(Cpt_m,Cp
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLReflect
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLThru(Cpt_m,Cp

Cal > **Modify Cal Kit** > **Specify CLSs** > **Thru** > **Port 1-2|Port 1-3|Port 1-4|Port 2-3|Port 2-4|Port 3-4**

> > **CLS** > > **1-2|1-3|1-4|2-3|2-4|3-4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:THRU <numeric 1>, <numeric 2>, <numeric 3>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:THRU? <numeric 1>, <numeric 2>

{numeric 3}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:THRU 1,2,11"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:THRU? 1,2"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.TRLLine

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.TRLLine(*Cpt_m*,*Cp*
= *Value*

Value =
SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.TRLLine(*Cpt_m*,*Cp*

(*Ch*) /2 (*Cpt_mCpt_n*)TRL

0

	Value
	Long
	021

```
Dim StanTrll As Long
SENSe(1).CORRection.COLLect.CKIT.ORDer.TRLLine(1,2) = 10
StanTrll = SENSe(1).CORRection.COLLect.CKIT.ORDer.TRLLine(1,2)
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLReflect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLThru(Cpt_m,Cp

Cal > **Modify Cal Kit** > **Specify CLSs** > **TRL Line/Match** > **Set All|Port 1-2|Port 1-3|Port 1-4|Port 2-3|Port 2-4|Port 3-4**

> > **CLS** > **TRL/>|1-2|1-3|1-4|2-3|2-4|3-4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:TRLLine <numeric 1>,<numeric 2>, <numeric 3>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:TRLLine? <numeric 1>,<numeric 2>

{numeric 3}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:TRLL 1,2,11"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:TRLL? 1,2"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.TRLRefl

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.TRLreflect = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.TRLreflect

(*Ch*)/2TRL

0

	Value
	Long
	021

```
Dim StanTrlr As Long
SENSe(1).CORRection.COLLect.CKIT.ORDer.TRLReflect = 5
StanTrlr = SENSe(1).CORRection.COLLect.CKIT.ORDer.TRLReflect
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLLine(Cpt_m,Cp

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLThru(Cpt_m,Cp

Cal > **Modify Cal Kit** > **Specify CLSs** > **TRL Reflect**

> > **CLS** > **TRL**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:TRLReflect <numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:TRLReflect?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:TRLR 11"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:TRLR?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.TRLThru

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.TRLThru(*Cpt_m*,*Cp*
= *Value*

Value =
SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.TRLThru(*Cpt_m*,*Cp*

/2 (*Cpt_mCpt_n*)TRL

0

	Value
	Long
	021

```
Dim StanTrlt As Long
SENSe(1).CORRection.COLLect.CKIT.ORDer.TRLThru(1,2) = 3
StanTrlt = SENSe(1).CORRection.COLLect.CKIT.ORDer.TRLThru(1,2)
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SELect
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLLine(Cpt_m,Cp
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLReflect

Cal > **Modify Cal Kit** > **Specify CLSs** > **TRL Thru** > **Set All|Port 1-2|Port 1-3|Port 1-4|Port 2-3|Port 2-4|Port 3-4**

> > **CLS** > **TRL** > **|1-2|1-3|1-4|2-3|2-4|3-4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:TRLThru <numeric 1>, <numeric 2>, <numeric 3>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:ORDer:TRLThru? <numeric 1>,<numeric 2>

{numeric 3}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:TRLT 1,2,5"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:TRLT? 1,2"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.RESet

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.RESet

(*Ch*)

SCPI.SENSe(1).CORRection.COLLect.CKIT.RESet

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:RESet

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:RES"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.SELect = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.SELect

/ (*Ch*)

	Value
	','Arial,8',10,10,00000000,0xc0ffff)">
	Long
	120
	1

```
Dim CalKit As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.SELect = 3
CalKit = SCPI.SENSe(1).CORRection.COLLect.CKIT.SELect
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SELect
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLLine(Cpt_m,Cp
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLReflect
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.TRLThru(Cpt_m,Cp
SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).DELay

Cal > **Cal Kit**

>

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT[:SELect] <numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT[:SELect]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT 3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).ARE

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).ARBitrary = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).ARBitrary

	Std
	Long
	121
	1

	Value
	Double
	-1E181E18

```
Dim StanArbt As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).ARBitrary = 50.5
StanArbt =
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).ARBitrary
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name**> **Arb. Impedance**
> > **STD** > **.** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:ARB 50.5"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:ARB?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C0

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C0 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C0

	Value
	C0
	Double
	-1E181E18
	fF ()1E-15 F

```
Dim StanC0 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C0 = 12.3
StanC0 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C0
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **C0**

> > **STD** > **.** > **C0**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C0 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C0?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C1

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C1 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C1

(*Ch*)/121 (*Std*) C1

	Value
	C1
	Double
	-1E181E18
	1E-27 F/Hz1E-27/

```
Dim StanC1 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C1 = 12.3
StanC1 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C1
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name**> **C1**

> > **STD** > **.** > **C1**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C1 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C2

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C2 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C2

(*Ch*)/121 (*Std*)C2

	Value
	C2
	Double
	-1E181E18
	1E-36 F/Hz2 1E-36/2

```
Dim StanC2 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C2 = 12.3
StanC2 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C2
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **C2**

> > > **.** > **C2**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C2 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C2?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C3

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C3 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C3

(*Ch*)/121 (*Std*)C3

	Value
	C3
	Double
	-1E181E18
	1E-45 F/Hz3 (1E-45/3)

```
Dim StanC3 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C3 = 12.3
StanC3 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C3
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **C3**

> > > **.** > **C3**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

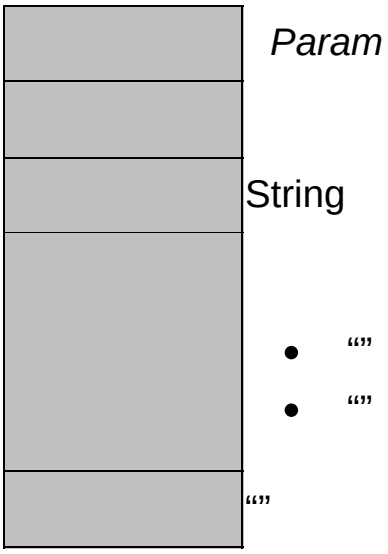
```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C3 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C3?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).CHA

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).CHARacter = *Param*

Param = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).CHARacter

(*Ch*)/121 (*Std*)C3


Param
String
“”
“”
“”

```
Dim StanChar As Double
SCPI.SENSe(1).SENSe.CORRection.COLLect.CKIT.STAN.CHARacter
= "WAVeguide"
StanChar =
SCPI.SENSe.CORRection.COLLect.CKIT.STAN.CHARacter
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).DELay

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **Media**

> > > . >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
{COAXial|WAVeguide}

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{COAXial|WAVeguide}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:CHAR WAV"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:CHAR?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).DEL

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).DELay = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).DELay

	Value
	Double
	-1E181E18
	s

```
Dim StanDel As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).DELay = 12.3
StanDel = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).DELay
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).CHARacter

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **Offset Delay**

> > > **.** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:DEL 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:DEL?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).FMA

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).FMAXimum = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).FMAXimum

(*Ch*)/121 (*Std*)

“”

	Value
	Double
	0999E9
	Hz

```
Dim StanFMax As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).FMAXimum =
13.2E9
StanFMax =
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).FMAXimum
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).FMINimum

[Cal] > **Modify Cal Kit** > **Define STDs** > **no. name** > **Max. Frequency**

[] > > > . >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:FMAX 5E9"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:FMAX?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).FMIN

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).FMINimum = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).FMINimum

“”

	Value
	Double
	0999E9
	Hz

```
Dim StanFMin As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).FMINimum =
600E6
StanFMax =
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).FMINimum
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).FMAXimum

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **Min. Frequency**

> > > **.** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:FMIN 1E9"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:FMIN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L0

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L0 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L0

	Value
	L0
	Double
	-1E181E18
	pH

```
Dim StanL0 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L0 = 12.3
StanL0 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L0
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **L0**

> > > **.** > **L0**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L0 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L0?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L1

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L1 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L1

	Value
	L1
	Double
	-1E181E18
	1E-24 H/Hz (1E-24/)

```
Dim StanL1 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L1 = 12.3
StanL1 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L1
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **L1**

> > > **.** > **L1**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L1 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L2

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L2 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L2

(*Ch*)/121 (*Std*)L2

	Value
	L2
	Double
	-1E181E18
	1E-33 H/Hz2 (1E-33/2)

```
Dim StanL2 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L2 = 12.3
StanL2 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L2
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **L2**

> > >**.** > **L2**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L2 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L2?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L3

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L3 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L3

	Value
	L3
	Double
	-1E181E18
	1E-42 H/Hz3 (1E-42/3)

```
Dim StanL3 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L3 = 12.3
StanL3 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L3
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **L3**

> > > **.** > **L3**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L3 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L3?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LAB

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LABel = *Lbl*

Lbl = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LABel

	Lbl
	(String)
	254

```
Dim StanLbl As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).LABel = "OPEN
3.5mm"
StanLbl = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).LABel
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **Label**

> > > . >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<string>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{string}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LAB ""OPEN"""
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LAB?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LOS

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LOSS = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LOSS

	Value
	(Double)
	-1E181E18
	/

```
Dim StanLoss As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).LOSS = 12.3
StanLoss =
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).LOSS
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > Modify Cal Kit > Define STDs > no. name > Offset Loss

> > > . >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]-21}:LOSS <numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]-21}:LOSS?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LOSS 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LOSS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).TYP

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).TYPE = *Param*

Param = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).TYPE

	Param
	(String)
	• “” • “” • “” • “” • “” • “” • “”DUT0

```
Dim StanType As String
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).TYPE = "OPEN"
StanType =
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).TYPE
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **STD Type**

> > > . >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]-21}:TYPE {OPEN|SHORt|LOAD|THRU|UTHRu|ARBI|NONE}

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]-21}:TYPE?

{OPEN|SHORt|LOAD|THRU|UTHRu|ARBI|NONE}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:TYPE OPEN"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:TYPE?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).Z0

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).Z0 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).Z0

(*Ch*)/121 (*Std*)Z0

	Value
	Z0
	(Double)
	-1E181E18

```
Dim StanZ0 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).Z0 = 50
StanZ0 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).Z0
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **Offset Z0**
> > > **.** > **Z0**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:
<numeric>

:SENSe{[1]-36}:CORRection:COLLect:CKIT:STAN{[1]|2|3|4|5|6|7|8|9|10|1:

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:Z0 50"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:Z0?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.TRLoption.IMP

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.TRLoption.IMPedance = *Param*

Param = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.TRLoption.IMPedance

(*Ch*)/TRL

	Param
	TRL
	(String)
	• “” • “”
	“”

Dim TrlImp As String
SCPI.SENSe(1).CORRection.COLLect.CKIT.TRLoption.IMPedance =
INE?br> TrlImp =
SCPI.SENSe(1).CORRection.COLLect.CKIT.TRLoption.IMPedance

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.TRLoption.RPLane

Cal > **Modify Cal Kit** > **TRL Option** > **Impedance**

> > **TRL** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:TRLoption:IMPedance {LINE|SYSTem}

:SENSe{[1]-36}:CORRection:COLLect:CKIT:TRLoption:IMPedance?

{LINE|SYST}<newline><^END>

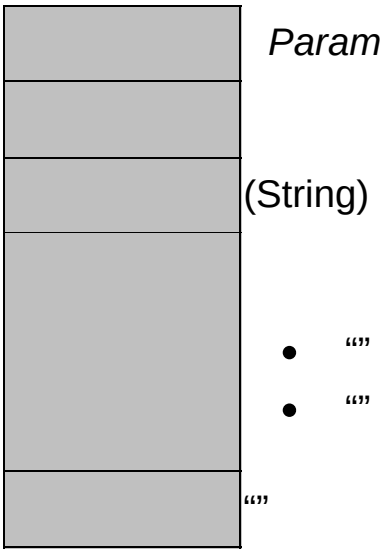
```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:TRL:IMP LINE"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:TRL:IMP?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.TRLoption.RPL

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.TRLoption.RPLane = *Param*

Param = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.TRLoption.RPLane

(Ch)/


Param
(String)
“”
“”
“”

```
Dim TrlRpl As String
SCPI.SENSe(1).SENSe(1).CORRection.COLLect.CKIT.TRLoption.RPLa
= EFLect?br>    TrlRpl =
SCPI.SENSe(1).CORRection.COLLect.CKIT.TRLoption.RPLane
```

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.TRLoption.IMPedance

Cal > **Modify Cal Kit** > **TRL Option** > **Reference Plane**

> > **TRL** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CKIT:TRLoption:RPLane {THRU|REFLect}

:SENSe{[1]-36}:CORRection:COLLect:CKIT:TRLoption:RPLANE?

{THRU|REFL}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:TRL:RPL REFL"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:TRL:RPL?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CLEar

SCPI.SENSe(*Ch*).CORRection.COLLect.CLEar

(*Ch*)

SCPI.SENSe(1).CORRection.COLLect.CLEar

SCPI.SENSe(Ch).OFFSet.STATe

Cal > **Calibrate** > **Responce(Open)** > **Cancel** > **OK**

> > > > **OK**

Cal > **Calibrate** > **Responce(Short)** > **Cancel** > **OK**

> > > > **OK**

Cal > **Calibrate** > **Responce(Thru)** > **Cancel** > **OK**

> > > > **OK**

Cal> **Calibrate** > **1-Port Cal** > **Cancel** > **OK**

> > **1** > > **OK**

Cal> **Calibrate** > **2-Port Cal** > **Cancel** > **OK**

> > **2** > > **OK**

Cal> **Calibrate** > **3-Port Cal** > **Cancel** > **OK**

> > **3** > > **OK**

Cal> **Calibrate** > **4-Port Cal** > **Cancel** > **OK**

> > **4** > > **O**K

SCPI

:SENSe{[1]-36}:CORRection:COLLect:CLEar

```
10 OUTPUT 717;":SENS1:CORR:COLL:CLE"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.CCHeck.ACQu

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.CCHeck.ACQuire

Ecal (*Ch*)Ecal

ECal
3.50

SCPI.SENSe(1).CORRection.COLLect.ECAL.CCHeck.ACQuire

Cal > **ECal** > **Confidence Check**

> **ECal** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:CCHeck[:ACQuire]

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:CCH"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.ERESponse

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.ERESponse = Eports

ECal

ECal

	Eports
	2 • *EPorts(0)* • *EPorts(1)* 0
	(Variant)
	14
	1

(*Ch*) Ch

```
Dim ERESPort(1) As Variant
ERESPort(0) = 1
ERESPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.ECAL.ERESponse = ERESPort
```

Cal > **ECal** > **Enhanced Response** > **2-1 (S21 S11)|3-1 (S31 S11)|…|3-4 (S34 S44)**

> **ECal** > > **2-1 S21 S11|3-1 S31 S11|...|3-4 S34 S44**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:ERESponse <numeric 1>, <numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ERES 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ISOLation.STA

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ISOLation.STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ISOLation.STATe

(*Ch*)/Ecal

	Status
	/ECal
	(Boolean)
	• •

```
Dim EcalIso As Boolean
SCPI.SENSe(1).CORRection.COLLect.ECAL.ISOLation.STATe = True
EcalIso =
SCPI.SENSe(1).CORRection.COLLect.ECAL.ISOLation.STATe
```

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT1
SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT2
SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT3
SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT4

Cal > **ECal** > **Isolation**
> **ECal** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:ISOLation[:STATe] {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:COLLect:ECAL:ISOLation[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ISOL ON"
20 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ISOL?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ORIentation.S

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ORIentation.STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ORIentation.STATe

/Ecal

	Status
	/ECal
	(Boolean)
	• •

(*Ch*) Ch

```
Dim EcalOri As Boolean
SCPI.SENSe(1).CORRection.COLLect.ECAL.ORIentation.STATe =
True
EcalOri =
SCPI.SENSe(1).CORRection.COLLect.ECAL.ORIentation.STATe
```

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(Cpt)

SCPI.CALCulate(Ch).FSIMulator.SENDed.DEEMbed.PORT(Pt).TYPE

Cal > **ECal** > **Orientation**

> **ECal** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:ORIentation[:STATe] {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:COLLect:ECAL:ORIentation[:STATe]?

{0|1}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ORI ON"
20 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ORI?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.PATH(*Cpt*)

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(*Cpt*) = *Value*

Value = SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(*Cpt*)

/ECaln

	Value
	ECal
	(Long)
	/

- 0
- 1A
- 2B
- 3C
- 4D

(*Cpt*) Cpt

```
Dim ECalPort As Long
SCPI.SENSe.CORRection.COLLect.ECAL.PATH(1) = 3
ECalPort = SCPI.SENSe.CORRection.COLLect.ECAL.PATH(1)
```

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.ORIentation.STATe

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:PATH <numeric 1>, <numeric 2>

:SENSe{[1]-36}:CORRection:COLLect:ECAL:PATH? <numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:PATH 1,2"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT1

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT1 = *Eport*

Ecal (*Ch*)1

ECal

	Eport
	(Long)
	14

SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT1 = 1

Cal > **ECal** > **1-Port Cal** > **Port 1|Port 2|Port 3|Port 4**
> **ECal** > **1** > **1|2|3|4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:SOLT1 <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:SOLT1 1"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT2

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT2 = *Eports*

ECal136 (*Ch*)22

ECa

	Eports
	2 • *EPorts(0)* *EPorts*(1) 2ECal 0
	(Variant)
	14
	1
	22

```
SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT2 = Array(1,2)
Dim EcalPort(1) As Variant
EcalPort(0) = 1
EcalPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT2 = EcalPort
```

Cal > **ECal** > **2-Port Cal** > **Port 1-2|Port 1-3|Port 1-4|Port 2-3|Port 2-4|Port 3-4**

> **ECal** > **2** > **1-2|1-3|1-4|2-3|2-4|3-4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:SOLT2 <numeric 1>, <numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:SOLT2 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT3

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT3 = *Eports*

ECal136 (*Ch*)33
4ECal

	Eports
	3 • *EPorts(0) EPorts*(1) *EPorts*(2) 3ECal 0
	(Variant)
	14
	1

```
SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT3 = Array(1,2,3)
```

Cal > **ECal** > **3-Port Cal** > **Port 1-2-3|Port 1-2-4|Port 1-3-4|Port 2-3-4**
> **ECal** > **3** > **1-2-3|1-2-4|1-3-4|2-3-4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:SOLT3 <numeric 1>, <numeric 2>,<numeric 3>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:SOLT3 1,2,3"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

```
Dim EcalPort(2) As Variant
EcalPort(0) = 1
EcalPort(1) = 2
EcalPort(2) = 3
SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT3 = EcalPort
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT4

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT4 = *Eports*

ECal (*Ch*)4

4ECal

	Eports
	4 • *EPorts(0) EPorts*(1) *EPorts*(2) *EPorts*(3) 4ECal 0
	(Variant)
	14
	1

```
SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT4 = Array(1,2,3,4)
Dim EcalPort(3) As Variant
EcalPort(0) = 1
EcalPort(1) = 2
EcalPort(2) = 3
EcalPort(3) = 4
SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT4 = EcalPort
```

Cal > **ECal** > **4-Port Cal**
> **ECal** > **4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:SOLT4 1,2,3,4

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:SOLT4 1,2,3,4"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.THRU

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.THRU = *Eports*

ECal (*Ch*)2

ECal

	Eports
	2 • *0* • *1* 0
	(Variant)
	14
	1
	2

```
SCPI.SENSe(1).CORRection.COLLect.ECAL.THRU = Array(1,2)
Dim EcalPort(1) As Variant
EcalPort(0) = 1
EcalPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.ECAL.THRU = EcalPort
```

Cal > **ECal** > **Thru Cal** > **2-1 (S21)|3-1 (S31)|4-1 (S41)|1-2 (S12)|3-2 (S32)| 4-2 (S42)| 1-3 (S13)|2-3 (S23)|4-3 (S43)|1-4 (S14)|2-4 (S24)|3-4 (S34)**

> **ECal** > > **2-1S21|3-1S31|4-1S41|1-2S12|3-2S32| 4-2S42| 1-3 S13|2-3S23|4-3S43|1-4S14|2-4S24|3-4S34**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:THRU <numeric 1>, <numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:SOLT2 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.UCHar

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.UCHar = *Param*

(*Ch*)EcalECal characteristic

EcalECalECal

ECal

	Param
	ECal
	(String)
	• “CHAR0”ECal • “CHAR1”ECal1 • “CHAR2”ECal2 • “CHAR3”ECal3 • “CHAR4”ECal4 • “CHAR5”ECal5
	“CHAR0”

```
Dim UserChar As String
SCPI.SENSe(1).CORRection.COLLect.ECAL.UCHar = "CHAR2"
UserChar = SCPI.SENSe(1).CORRection.COLLect.ECAL.UCHar
```

Cal > **ECal** > **Characterization** >
Factory|User1|User2|User3|User4|User5

> **ECal** > > **|1|2|3|4|5**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:UCHar {CHAR0| CHAR1 |CHAR2|CHAR3|CHAR4|CHAR5}

:SENSe{[1]-36}:CORRection:COLLect:ECAL:UCHar?

{CHAR0|CHAR1|CHAR2|CHAR3|CHAR4|CHAR5}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:UCH CHAR1"
20 OUTPUT 717;":SENS1:CORR:COLL:ECAL:UCH?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.UTHRU.STATe

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.UTHRU.STATe = *Status*

Ecal

	Status
	Ecal
	• •

```
Dim EcalPort As Boolean
EcalPort = 0
SCPI.SENSe(1).CORRection.COLLect.ECAL.UTHRU.STATe =
EcalPort
```

Cal > **ECal** > **Unknown Thru**
> **ECal** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:ECAL:UTHRU:STATe {ON|OFF|1|0}

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:UTHRU:STAT 1"
20 OUTPUT 717;":SENS1:CORR:COLL:ECAL:UTHRU:STAT?"
30 ENTER 717;A
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.ADAPter.R

SCPI.SENSe(Ch).CORRection.COLLect.METHod.ADAPter.REMoval = *Port*

(*Ch*)4

	Port
	/
	(Variant)
	1 4
	1

SCPI.SENSe(Ch).CORRection.COLLect.METHod.ADAPter.REMoval = 1

SCPI.SENSe(Ch).CORRection.COLLect.ADAPter(Pt).LENGth

Cal > **Calibrate** > **Adapter Removal** > **Select Port**

> > >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod:ADAPter:REMoval {1-4}

```
Call viVPrintf(vi, ":SENS1:CORR:COLL:METH:ADAPter:REMoval
 1"+vbcr, 0)
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.ERESpons

SCPI.SENSe(Ch).CORRection.COLLect.METHod.ERESponse = Ports

	Ports
	2 • *0* • *1* 0
	(Variant)
	14
	1

(*Ch*) Ch

```
Dim ERESPort(1) As Variant
ERESPort(0) = 1
ERESPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.METHod.ERESponse =
ERESPort
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Cal > **Calibrate** > **Enhanced Response** > **Select Ports** > **2-1 (S21 S11)|3-1 (S31 S11)**|...**|3-4 (S34 S44)**

> > > > **2-1S21 S11|3-1S31 S11|...|3-4S34 S44**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod:ERESponse <numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:ERES 1,2"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.OPEN = *Port*

(*Ch*)

SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.OPEN = 1

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Cal > **Calibrate** > **Response (Open) > Select Port**

> > >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod[:RESPonse]:OPEN <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:OPEN 1"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.SHORt = *Port*

(*Ch*)

SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.SHORt = 1

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Cal > **Calibrate** > **Response (Short) > Select Port**

> > >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod[:RESPonse]:SHORt <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:SHOR 1"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.THRU = *Ports*

(*Ch*)

(*Ch*)(*Port*) ChPort

```
SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.THRU =
Array(2,1)
Dim ThruPort(1) As Variant
ThruPort(0) = 2
ThruPort(1) = 1
SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.THRU =
ThruPort
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Cal > **Calibrate** > **Response (Thru) > Select Ports**

> > >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod[:RESPonse]:THRU
<numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:THRU 1,2"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT1

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT1 = *Port*

(*Ch*)1

SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT1 = 1

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Cal > **Calibrate** > **1-Port Cal > Select Port**

> > **1** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod:SOLT1 <numeric>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:SOLT1 1"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT2

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT2 = *Ports*

(*Ch*)22

	Ports
	2 • *0* 2 • *1* 0
	(Variant)
	14
	1
	22

```
SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT2 = Array(1,2)
Dim CalPort(1) As Variant
CalPort(0) = 1
CalPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT2 = CalPort
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Cal > **Calibrate** > **2-Port Cal > Select Ports**

> > **2** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod:SOLT2 <numeric 1>, <numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:SOLT2 1,2"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT3

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT3 = *Ports*

(*Ch*)33

	Ports
	3 • *0* 3 • *1* 3 • *2* 3 0
	(Variant)
	14
	1
	223

```
SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT3 = Array(1,2,3)
Dim CalPort(2) As Variant
CalPort(0) = 1
CalPort(1) = 2
CalPort(2) = 3
SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT3 = CalPort
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Cal > **Calibrate** > **3-Port Cal > Select Ports**

> > **3** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod:SOLT3 <numeric 1>, <numeric 2>,<numeric 3>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:SOLT3 1,2,3"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT4

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT4 = *Ports*

(*Ch*)4

	Ports
	4 • *04* • *14* • *24* • *34* 0
	(Variant)
	14
	1
	224

```
SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT4 =
Array(1,2,3,4)
Dim CalPort(3) As Variant
CalPort(0) = 1
CalPort(1) = 2
CalPort(2) = 3
CalPort(3) = 4
SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT4 = CalPort
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Cal > **Calibrate** > **4-Port Cal**

> > **4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod:SOLT4 1,2,3,4

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:SOLT3 1,2,3,4"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.TRL2

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.TRL2 = *Ports*

(*Ch*)2TRL

	Ports
	2 • *0* TRL2 • *1* TRL2 0
	(Variant)
	14
	1
	222

```
SCPI.SENSe(1).CORRection.COLLect.METHod.TRL2 = Array(1,2)
Dim TrlCalPort(1) As Variant
TrlCalPort(0) = 1
TrlCalPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.METHod.TRL2 = TrlCalPort
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL3
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL4
SCPI.SENSe(Ch).CORRection.COLLect.SAVE
SCPI.SENSe(Ch).CORRection.TYPE(Tr)

Cal > **Calibrate** > **2-Port TRL Cal** > **Select Ports** > **1-2**|**1-3**|**1-4**|**2-3**|**2-4**|**3-4**

> > **2TRL** > > **1-2**|**1-3**|**1-4**|**2-3**|**2-4**|**3-4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect
:METHod:TRL2 <numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:TRL2 1,2"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.TRL3

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.TRL3 = *Ports*

(*Ch*)3TRL

	Ports
	3 • *0* TRL3 • *1* TRL3 • *2* TRL3 0
	(Variant)
	14
	1
	223

```
SCPI.SENSe(1).CORRection.COLLect.METHod.TRL3 = Array(1,2,3)
Dim TrlCalPort(2) As Variant
TrlCalPort(0) = 1
TrlCalPort(1) = 2
TrlCalPort(2) = 3
SCPI.SENSe(1).CORRection.COLLect.METHod.TRL3 = TrlCalPort
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL2
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL4
SCPI.SENSe(Ch).CORRection.COLLect.SAVE
SCPI.SENSe(Ch).CORRection.TYPE(Tr)

Cal > **Calibrate** > **3-Port TRL Cal** > **Select Ports** > **1-2-3**|**1-2-4**|**1-3-4**|**2-3-4**

> > **3TRL** > > **1-2-3**|**1-2-4**|**1-3-4**|**2-3-4**

SCPI

:SENSe{[1]-36}:CORRection:COLLect
:METHod:TRL3 <numeric 1>,<numeric 2>,<numeric 3>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:TRL3 1,2,3"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.TRL4

SCPI.SENSe(*Ch*).CCORRection.COLLect.METHod.TRL4 = *Ports*

(*Ch*)4TRL

	Ports
	4 • *0* TRL4 • *1* TRL4 • *2* TRL4 • *3* TRL4 0
	(Variant)
	14
	1
	224

```
SCPI.SENSe(1).CORRection.COLLect.METHod.TRL4 = Array(1,2,3,4)
Dim TrlCalPort(3) As Variant
TrlCalPort(0) = 1
TrlCalPort(1) = 2
TrlCalPort(2) = 3
TrlCalPort(3) = 4
SCPI.SENSe(1).CORRection.COLLect.METHod.TRL4 = TrlCalPort
```

SCPI.SENSe(Ch).CORRection.COEFficient.DATA
SCPI.SENSe(Ch).CORRection.COEFficient.SAVE
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL2
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL3
SCPI.SENSe(Ch).CORRection.COLLect.SAVE
SCPI.SENSe(Ch).CORRection.TYPE(Tr)

Cal > **Calibrate** > **4-Port TRL Cal**

> > **4TRL**

SCPI

:SENSe{[1]-36}:CORRection:COLLect
:METHod:TRL4 <numeric 1>,<numeric 2>,<numeric 3>,<numeric 4>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:TRL4 1,2,3,4"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.TYPE

Param = SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.TYPE

(*Ch*)

SCPI.SENSe(Ch).CORRection.TYPE(Tr)

Param

AREM	
ERES	
NONE	
RESPO	
RESPS	
RESPT	
SOLT1	1
SOLT2	2
SOLT3	3
SOLT4	4
TRL2	TRL2
TRL3	TRL3
TRL4	TRL4

(String)

```
Dim CalType As String
CalType = SCPI.SENSe(1).CORRection.COLLect.METHod.TYPE
```

SCPI.SENSe(Ch).CORRection.COLLect.SAVE

SCPI.SENSe(Ch).CORRection.TYPE(Tr)

SCPI

:SENSe{[1]-36}:CORRection:COLLect:METHod:TYPE?

{AREM|ERES|NONE|RESPO|RESPS|RESPT|SOLT1|SOLT2|SOLT3|SOL
<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:TYPE?"
20 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.PARTial.SAVE

SCPI.SENSe(*Ch*).CORRection.COLLect.PARTial.SAVE

(*Ch*) Ch

SCPI.SENSe(1).CORRection.COLLect.PARTial.SAVE

Cal > **Calibrate** > **n-Port Cal** > **Overwrite**

> > **n** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:PARTial:SAVE

```
10 OUTPUT 717;":SENS1:CORR:COLL:PART:SAVE"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.SAVE

SCPI.SENSe(*Ch*).CORRection.COLLect.SAVE

(*Ch*) Ch

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.THRU =
Array(2,1)
SCPI.SENSe(1).CORRection.COLLect.ACQuire.THRU = Array(2,1)
Dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(1).CORRection.COLLect.SAVE
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.RESPonse.OPEN
SCPI.SENSe(Ch).CORRection.COLLect.METHod.RESPonse.SHORt
SCPI.SENSe(Ch).CORRection.COLLect.METHod.RESPonse.THRU
SCPI.SENSe(Ch).CORRection.COLLect.METHod.SOLT1
SCPI.SENSe(Ch).CORRection.COLLect.METHod.SOLT2
SCPI.SENSe(Ch).CORRection.COLLect.METHod.SOLT3
SCPI.SENSe(Ch).CORRection.COLLect.METHod.SOLT4
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL2
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL3
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL4

Cal > **Calibrate** > **Response|n-Port Cal** > **Done**

> > **|n** >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:SAVE

```
10 OUTPUT 717;":SENS1:CORR:COLL:SAVE"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.SIMPlified.SAVE

SCPI.SENSe(*Ch*).CORRection.COLLect.SIMPlified.SAVE

1/2 SCPI.SENSe(Ch).CORRection.COLLect.SAVE

34

3.50

(*Ch*) Ch

```
SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT3 = Array(1,2,3)
SCPI.SENSe(1).CORRection.COLLect.SIMPlified.SAVE
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.SOLT3
SCPI.SENSe(Ch).CORRection.COLLect.METHod.SOLT4
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL3
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL4
SCPI.SENSe(Ch).CORRection.COLLect.SAVE

Cal > **Calibrate** > **n-Port Cal|n-Port TRL** > **Done**

> > n|nTRL >

SCPI

:SENSe{[1]-36}:CORRection:COLLect:SIMPlified:SAVE

```
10 OUTPUT 717;":SENS1:CORR:COLL:SIMP:SAVE"
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.CONFig

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.CONFig = *Param*

Param = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.CONFig

(Ch)/

	Param
	(String)
	• “CSPN” • “AMKR” 12 • “USPN”
	“CSPN”

```
Dim Conf As String
SCPI.SENSe(1).CORRection.EXTension.AUTO.CONFig = "AMKR"
Conf = SCPI.SENSe(1).CORRection.EXTension.AUTO.CONFig
```

SCPI.SENSe(Ch).CORRection.EXTension.STATe
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Cal > **Port Extensions** > **Auto Port Extension** > **Method** > **Current Span|Active Marker|User Span**

> > > > **||**

SCPI

:SENSe{[1]-36}:CORRection:EXTension:AUTO:CONFig {CSPN|AMKR|USPN}

:SENSe{[1]-36}:CORRection:EXTension:AUTO:CONFig?

{CSPN|AMKR|USPN}<newline><^END>

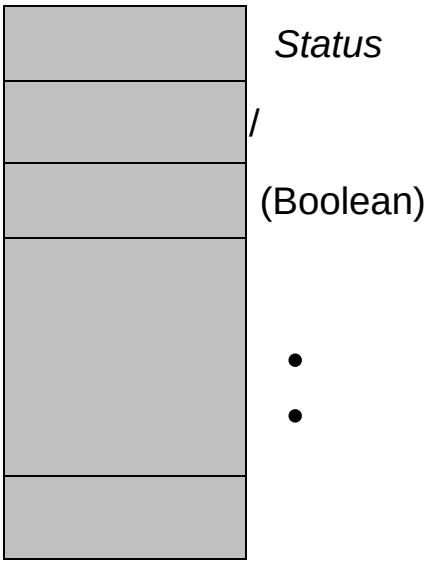
```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:CONF CSPN"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:CONF?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.DCOFfset

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.DCOFfset = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.DCOFfset

(Ch)/


Status
/
(Boolean)

```
Dim Dcof As Boolean
SCPI.SENSe(1).CORRection.EXTension.AUTO.DCOFfset = True
Dcof = SCPI.SENSe(1).CORRection.EXTension.AUTO.DCOFfset
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect
SCPI.CALCulate(Ch).SELected.CONVersion.FUNCtion
SCPI.SENSe(Ch).CORRection.EXTension.STATe
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Cal > **Port Extensions** > **Auto Port Extension** > **Adjust Mismatch**

> > >

SCPI

:SENSe{[1]-36}:CORRection:EXTension:AUTO:DCOFfset {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:EXTension:AUTO:DCOFfset?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:DCOF ON"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:DCOF?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.LOSS

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.LOSS = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.LOSS

(Ch)/

	Status
	(Boolean)
	• •

```
Dim AutoLoss As Boolean
SCPI.SENSe(1).CORRection.EXTension.AUTO.LOSS = True
AutoLoss = SCPI.SENSe(1).CORRection.EXTension.AUTO.LOSS
```

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Cal > **Port Extensions** > **Auto Port Extension** > **Include Loss**

> > >

SCPI

:SENSe{[1]-36}:CORRection:EXTension:AUTO:LOSS {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:EXTension:AUTO:LOSS?

{1|0}<newline><^END>

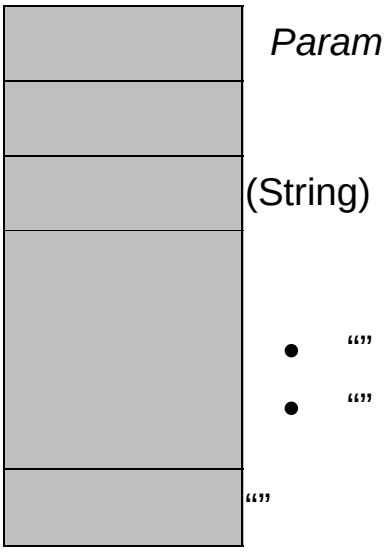
```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:LOSS ON"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:LOSS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.MEASure

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.MEASure = *Param*

Param = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.MEASure

(*Ch*)


Param
(String)
“”
“”
“”

```
Dim AutoMeas As String
SCPI.SENSe(1).CORRection.EXTension.AUTO.MEASure = "OPEN"
AutoLoss = SCPI.SENSe(1).CORRection.EXTension.AUTO.LOSS
```

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Cal > **Port Extensions** > **Auto Port Extension** > **Measure OPEN|Measure Short- All|Port 1|Port 2|Port 3|Port 4**

> > > **|-|1|2|3|4**

SCPI

:SENSe{[1]-36}:CORRection:EXTension:AUTO:MEASure {OPEN|SHORt}

:SENSe{[1]-36}:CORRection:EXTension:AUTO:MEASure?

{OPEN|SHOR}<newline><^END>

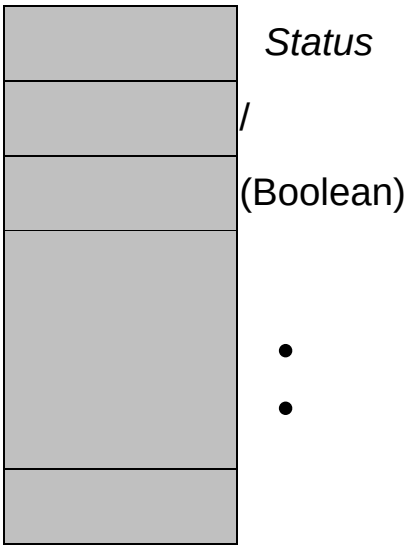
```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:MEAS OPEN"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:MEAS?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.PORT(*Pt*)

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.PORT(*Pt*) = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.PORT(*Pt*)

(Ch)/



(*Ch*) ChPt

```
Dim APort As Boolean
SCPI.SENSe(1).CORRection.EXTension.AUTO.PORT(1) = True
APort = SCPI.SENSe(1).CORRection.EXTension.AUTO.PORT(1)
```

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Cal > **Port Extensions** > **Auto Port Extension** > **Select Ports** > **Port 1|Port 2|Port 3|Port 4**

> > > >**1|2|3|4**

SCPI

:SENSe{[1]-36}:CORRection:EXTension:AUTO:PORT{[1]|2|3|4}{ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:EXTension:AUTO:PORT{[1]|2|3|4}?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:PORT1 ON"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:PORT1?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.RESet

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.RESet

(*Ch*)

SCPI.SENSe(1).CORRection.EXTension.AUTO.RESet

SCPI.SENSe(Ch).CORRection.EXTension.STATe
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

SCPI

:SENSe{[1]-36}:CORRection:EXTension:AUTO:RESet

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:RESet"
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STARt

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STARt = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STARt

(*Ch*) /

	Value
	(Double)
	9E33E9230235430435 1E58.5E9280285480485
	3E5
	Hz

```
Dim AStart As Double
SCPI.SENSe(1).CORRection.EXTension.AUTO.STARt = 1E9
AStart = SCPI.SENSe(1).CORRection.EXTension.AUTO.STARt
```

SCPI.SENSe(Ch).CORRection.EXTension.STATe
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Cal > Port Extensions > Auto Port Extension > Method > User Span Start

> > > >

SCPI

:SENSe{[1]-36}:CORRection:EXTension:AUTO:STARt <numeric>

:SENSe{[1]-36}:CORRection:EXTension:AUTO:STARt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:STAR 1.2E9"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:STAR?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STOP

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STOP = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STOP

(Ch)/

	Value
	(Double)
	9E33E9230235430435 1E58.5E9280285480485
	8500000000
	Hz

```
Dim AStop As Double
SCPI.SENSe(1).CORRection.EXTension.AUTO.STOP = 250E8
AStop = SCPI.SENSe(1).CORRection.EXTension.STOP.STARt
```

SCPI.SENSe(Ch).CORRection.EXTension.STATe
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt

Cal > **Port Extensions** > **Auto Port Extension** > **Method** > **User Span Stop**

> > > >

SCPI

:SENSe{[1]-36}:CORRection:EXTension:AUTO:STOP <numeric>

:SENSe{[1]-36}:CORRection:EXTension:AUTO:STOP?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:STOP 1.5E9"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:STOP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).FREQue

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).FREQuency(*Fq*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).FREQuency(*Fq*)

(*Ch*))/

	Fq
	(Long)
	12
	1

	Value
	(Double)
	9E33E9230430 9E38.5E9280480 1E53E9235435 1E58.5E9285485
	0
	Hz

(*Ch*)(*Pt*) ChPt

```
Dim PortFreq As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).FREQuency(1) =
500E6
PortFreq =
SCPI.SENSe(1).CORRection.EXTension.PORT(1).FREQuency(1)
```

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)

Cal > **Port Extensions** > **Loss** > **Freq1|Freq2**

> > > **1|2**

SCPI

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:FREQuency{[1]|
<numeric>

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:FREQuency{[1]|

<numeric><newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:FREQ1 10E6"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:FREQ1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).INCLude

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).INCLude(*Il*).STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).INCLude(*Il*).STATe

(*Ch*)14 (*Pt*)12(*Il*)

	ll
	(Long)
	12
	1

	Status
	(Boolean)
	• •

```
Dim PortIncl As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).INCLude(1).STATe =
500E6
PortIncl =
SCPI.SENSe(1).CORRection.EXTension.PORT(1).INCLude(1).STATe
```

SCPI.SENSe(Ch).CORRection.EXTension.STATe
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME

Cal > **Port Extensions** > **Loss** > **Loss1|Loss2**

> > > **1|2**

SCPI

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:INCLude{[1]|2}[:STATe] {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:INCLude{[1]|2}[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:INCL1 ON"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:INCL1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LDC

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LDC = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LDC

	Value
	(Double)
	-9090
	0
	dBm

```
Dim PLdc As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).LDC = 45
PLdc = SCPI.SENSe(1).CORRection.EXTension.PORT(1).LDC
```

SCPI.SENSe(Ch).CORRection.EXTension.STATe
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME

Cal > **Port Extensions** > **Loss** > **Loss at DC**

> > >

SCPI

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:LDC <numeric>

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:LDC?

<numeric><newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:LDC 1.2"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:LDC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LOSS(Lo

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LOSS(*Loss*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LOSS(*Loss*)

	Loss
	(Long)
	12
	1

	Value
	(Double)
	-9090
	0
	dBm

```
Dim PLoss As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).LOSS(1) = -45
PLoss = SCPI.SENSe(1).CORRection.EXTension.PORT(1).LOSS(1)
```

SCPI.SENSe(Ch).CORRection.EXTension.STATe
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME

Cal > **Port Extensions** > **Loss** > **Loss1|Loss2**

> > > **1|2**

SCPI

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:LOSS{[1]|2}
<numeric>

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:LOSS{[1]|2}?

<numeric><newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:LOSS1 0.8"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:LOSS1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).TIME

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).TIME = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).TIME

	Value
	(Double)
	-1010
	0
	s

```
Dim PortExt As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).TIME = 1E-3
PortExt = SCPI.SENSe(1).CORRection.EXTension.PORT(1).TIME
```

SCPI.SENSe(Ch).CORRection.EXTension.STATe
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)

Cal > **Port Extensions** > **Extension Port N**

> > **N**

SCPI

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:TIME <numeric>

:SENSe{[1]-36}:CORRection:EXTension:PORT{[1]|2|3|4}:TIME?

<numeric><newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:TIME 1e-3"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:TIME?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.STATe

SCPI.SENSe(*Ch*).CORRection.EXTension.STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim Ext As Boolean
SCPI.SENSe(1).CORRection.EXTension.STATe = True
Ext = SCPI.SENSe(1).CORRection.EXTension.STATe
```

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)
SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME

Cal > **Port Extensions** > **Extensions**

> >

SCPI

:SENSe{[1]-36}:CORRection:EXTension[:STATe] {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:EXTension[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:EXT ON"
20 OUTPUT 717;":SENS1:CORR:EXT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.IMPedance.INPut.MAGNitude

SCPI.SENSe(Ch).CORRection.IMPedance.INPut.MAGNitude = *Value*

Value = SCPI.SENSe(Ch).CORRection.IMPedance.INPut.MAGNitude

/Z0

3.01

	Value
	Z0
	(Double)
	1E-31000
	50
	0.001

```
Dim SysZ0 As Double
SCPI.SENSe.CORRection.IMPedance.INPut.MAGNitude = 75
SysZ0 = SCPI.SENSe.CORRection.IMPedance.INPut.MAGNitude
```

Equivalent key

Cal > **Set Z0**

> **Z0**

SCPI

:SENSe:CORRection:IMPedance[:INPut][:MAGNitude] <numeric>

:SENSe:CORRection:IMPedance[:INPut][:MAGNitude]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS:CORR:IMP 75"
20 OUTPUT 717;":SENS:CORR:IMP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.CLEar

SCPI.SENSe(*Ch*).CORRection.OFFSet.CLEar

(*Ch*)

SCPI.SENSe(1).CORRection.OFFSet.CLEar

SCPI.SENSe(Ch).OFFSet.STATe

SCPI.SENSe(Ch).CORRection.CLEar

Cal > **Mixer/Converter Calibration** > **Clear** > **OK**

> **/** > > **OK**

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:CLEar

```
10 OUTPUT 717;":SENS1:CORR:OFFS:CLE"
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.LO

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.LOAD = *Port*s

(*Ch*)

	Ports
	2 0 1 0
	(Variant)
	14
	“SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency”

1

```
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.LOAD =
Array(1,2)
```

2

```
Dim Port As Variant
Port(0) = 1
Port(1) = 2
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.LOAD = Port
```

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.OPEN
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.SHORt

Cal > **Mixer/Converter Caribration** > **Scalar Cal(Manual)** > **Reflection** > **PortX@FreqY Broadband**

> **/** > > > **X@Y**

XY

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect[:ACQuire]:LOAD
<numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:LOAD 1,4"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.OP

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.OPEN = *Port*s

(*Ch*)

	Ports
	2 0 1 0
	(Variant)
	14
	“SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency”

1

SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.OPEN = Array(1,2)

2

```
Dim Port As Variant
Port(0) = 1
Port(1) = 2
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.OPEN = Port
```

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.LOAD
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.SHORt

Cal > **Mixer/Converter Caribration** > **Scalar Cal(Manual)** > **Reflection** > **PortX@FreqY Open**

> **/** > > > **X@Y**

XY

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect[:ACQuire]:OPEN
<numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:OPEN 1,4"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.PM

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.PMETer = *Params*

(*Ch*)

	Params
	3 0 1 2 0
	(Variant)
	/14 • “ASENsor”A • “BSENsor”B
	“SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency”

```
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.PMETer =
Array(1,2,"ASENsor")

Dim Params As Variant
Params(0) = 1
Params(1) = 2
Params(2) = "ASENsor"
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.PMETer =
Params
```

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.LOAD
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.OPEN
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.SHORt

Cl > **Mixer/Converter Caribration** > **Scalar Cal(Manual)** > **Power Meter** > **Use Sensor A|Use Sensor B**

> **/** > > **A|B**

Cal > **Mixer/Converter Caribration** > **Scalar Cal(Manual)** > **Power Meter** > **PortX@FreqY**

> **/** > > **X@Y**

XY

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect[:ACQuire]:PMETer
<numeric 1>,<numeric 2>,{ASENsor|BSENsor}

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:PMET 1,4,ASEN"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.SH

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.SHORt = *Ports*

(*Ch*)

	Ports
	2 0 1 0
	(Variant)
	14
	“SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency”

```
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.SHORt =
Array(1,2)

Dim Port As Variant
Port(0) = 1
Port(1) = 2
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.SHORt = Port
```

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.LOAD
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.OPEN

Cal > **Mixer/Converter Caribration** > **Scalar Cal(Manual)** > **Reflection** > **PortX@FreqY Short**

> **/** > > > **X@Y**

XY

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect[:ACQuire]:SHORt
<numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:SHOR 1,4"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.TH

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ACQuire.THRU = *Port*s

(*Ch*)

	Ports
	2 0 1 0
	(Variant)
	14
	12S22S1221S11S21 2

```
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.THRU =
Array(1,2)

Dim Port As Variant
Port(0) = 1
Port(1) = 2
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ACQuire.THRU = Port
```

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.LOAD
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.OPEN
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.SHORt

Cal > **Mixer/Converter Caribration** > **Scalar Cal(Manual)** > **Transmission** > **PortX-Y@FreqZ Thru**

> **/** > > > **X-Y@Z**

X-YZ

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect[:ACQuire]:THRU
<numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:THRU 1,4"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.CLEar

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.CLEar

(*Ch*)

SCPI.SENSe(1).CORRection.OFFSet.COLLect.CLEar

SCPI.SENSe(Ch).OFFSet.STATe

SCPI.SENSe(Ch).CORRection.OFFSet.CLEar

Cal > **Mixer/Converter Calibration** > **Scalar Cal(Manual)** > **Cancel** > **OK**

> **/** > > > **OK**

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect:CLEar

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:CLE"
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ECAL.SMIX2

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ECAL.SMIX2 = *Port*s

ECal (*Ch*)2

ECal

	Ports
	2 01 12 //12 0
	(Variant)
	14
	“SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.PMETer” 2

```
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ECAL.SMIX2 =
Array(1,2)

Dim Port As Variant
Port(0) = 1
Port(1) = 2
SCPI.SENSe(1).CORRection.OFFSet.COLLect.ECAL.SMIX2 = Port
```

SCPI.SENSe(Ch).OFFSet.STATe

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.PMETer

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ECAL.SOLT1

Cal > **Mixer/Converter Caribration** > **Scalar Cal(ECal)** > **ECal & Done**

> **/** > > **ECal &**

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect:ECAL:SMIX2 <numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:ECAL:SMIX2 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ECAL.SOLT

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.ECAL.SOLT1 = *EPort*

ECal (*Ch*)1

ECal

	EPort
	1
	(Long)
	14

1

SCPI.SENSe(1).CORRection.OFFSet.COLLect.ECAL.SOLT1 = 1

SCPI.SENSe(Ch).OFFSet.STATe

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ECAL.SMIX2

Cal > **Mixer/Converter Caribration** > **Scalar Cal(ECal)** > **Select Ports** > **Port n**

> **/** > > > **n**

Cal > **Mixer/Converter Caribration** > **Scalar Cal(ECal)** > **ECal & Done**

> **/** > > **ECal &**

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect:ECAL:SOLT1 <numeric>

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:ECAL:SOLT1 2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.METHod.SM

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.METHod.SMIX2 = *Ports*

(*Ch*)22

Ports
2 01 12 the “SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.LOAD”,“SC “SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.THRU”,12 0
(Variant)
14
2

```
SCPI.SENSe(1).CORRection.OFFSet.COLLect.METHod.SMIX2 =
Array(1,2)

Dim Port As Variant
Port(0) = 1
Port(1) = 2
SCPI.SENSe(1).CORRection.OFFSet.COLLect.METHod.SMIX2 = Port
```

SCPI.SENSe(Ch).OFFSet.STATe
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.LOAD
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.OPEN
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.SHORt
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.THRU

Cal > **Mixer/Converter Caribration** > **Scalar Cal(Manual)** > **Select Ports** > **2-1(fwd)|1-2(rev)|1-2(both**

> **/** > > > **2-1|1-2|1-2**

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect:METHod:SMIX2 <numeric 1>,<numeric 2>

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:METH:SMIX2 1,2"
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.METHod.SO

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.METHod.SOLT1 = *Port*

(*Ch*)1

Port

1

(Long)

14

1

SCPI.SENSe(1).CORRection.OFFSet.COLLect.METHod.SOLT1 = 2

SCPI.SENSe(Ch).OFFSet.STATe

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.METHod.SMIX2

Cal > **Mixer/Converter Caribration** > **Scalar Cal(Manual)** > **Select Ports** > **Port n**

> **/** > > > **n**

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect:METHod:SOLT1 <numeric>

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:METH:SOLT1 2"
```

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.SAVE

SCPI.SENSe(*Ch*).CORRection.OFFSet.COLLect.SAVE

(*Ch*)

SCPI.SENSe(1).CORRection.OFFSet.COLLect.SAVE

SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.METHod.SMIX2
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.METHod.SOLT1
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.LOAD
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.OPEN
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.SHORt
SCPI.SENSe(Ch).CORRection.OFFSet.COLLect.ACQuire.THRU

Cal > **Mixer/Converter Caribration** > **Scalar Cal(Manual)** > **Done**

> **/** > >

SCPI

:SENSe{[1]-36}:CORRection:OFFSet:COLLect:SAVE

```
10 OUTPUT 717;":SENS1:CORR:OFFS:COLL:SAVE"
```

SCPI.SENSe(*Ch*).CORRection.PROPerty

SCPI.SENSe(*Ch*).CORRection.PROPerty = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.PROPerty

(Ch)/

	Status
	(Boolean)
	• •

```
Dim CalProp As Boolean
SCPI.SENSe(1).CORRection.PROPerty = True
CalProp = SCPI.SENSe(1).CORRection.PROPerty
```

Cal > **Property**

>

SCPI

:SENSe{[1]-36}:CORRection:PROPerty {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:PROPerty?

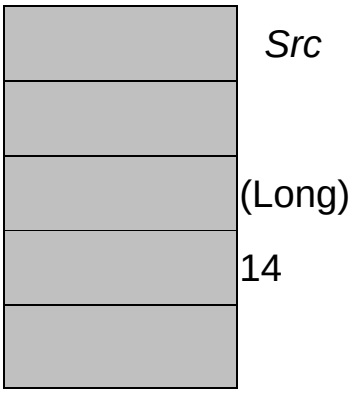
{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:PROP ON"
20 OUTPUT 717;":SENS1:CORR:PROP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.RECeiver(*Pt*).COLLect.ACQui

SCPI.SENSe(*Ch*).CORRection.RECeiver(*Pt*).COLLect.ACQuire = *Src*

(*Ch*)(*Pt*)This command executes receiver calibration, measurement the selected port of selected channel .


Src
(Long)
14

SCPI.SENSe(1).CORRection.RECeiver(2).COLLect.ACQuire = 4

SCPI.SENSe(Ch).CORRection.RECeiver(Pt).STATe

Cal > **Receiver Calibration** > **Select Port 1|2|3|4**

> > **1|2|3|4**

Cal > **Receiver Calibration** > **Source Port 1|2|3|4**

> > **1|2|3|4**

Cal > **Receiver Calibration** > **Take Cal Sweep**

> >

SCPI

:SENSe{[1]-36}:CORRection:RECeiver{[1]|2|3|4}:COLLect:ACQuire <numeric>

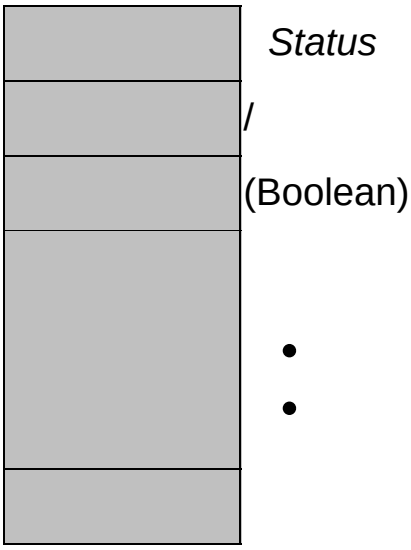
```
10 OUTPUT 717;":SENS1:CORR:REC1:COLL:ACQ 2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.RECeiver(*Pt*).STATe

SCPI.SENSe(*Ch*).CORRection.RECeiver(*Pt*).STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.RECeiver(*Pt*).STATe

(*Ch*)(*Pt*)/


Status
/
(Boolean)

```
Dim Corr As Boolean
SCPI.SENSe(1).CORRection.RECeiver(1).STATe = True
Corr = SCPI.SENSe(1).CORRection.RECeiver(1).STATe
```

SCPI.SENSe(Ch).CORRection.RECeiver(Pt).COLLect.ACQuire

Cal > **Receiver Calibration** > **Correction**

> >

SCPI

:SENSe{[1]-36}:CORRection:RECeiver{[1]|2|3|4}[:STATe] {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:RECeiver{[1]|2|3|4}[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:REC1 ON"
20 OUTPUT 717;":SENS1:CORR:REC1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.RVELocity.COAX

SCPI.SENSe(*Ch*).CORRection.RVELocity.COAX = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.RVELocity.COAX

(Ch)/

	Value
	(Double)
	010
	1

```
Dim Vel As Double
SCPI.SENSe(1).CORRection.RVELocity.COAX = 0.5
Vel = SCPI.SENSe(1).CORRection.RVELocity.COAX
```

SCPI

:SENSe{[1]-36}:CORRection:RVELocity:COAX <numeric>

:SENSe{[1]-36}:CORRection:RVELocity:COAX?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:RVEL:COAX 0.7"
20 OUTPUT 717;":SENS1:CORR:RVEL:COAX?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.STATe

SCPI.SENSe(*Ch*).CORRection.STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim Corr As Boolean
SCPI.SENSe(1).CORRection.STATe = True
Corr = SCPI.SENSe(1).CORRection.STATe
```

Cal > **Correction**

>

SCPI

:SENSe{[1]-36}:CORRection:STATe {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:STAT ON"
20 OUTPUT 717;":SENS1:CORR:STAT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.TRIGger.FREE.STATe

SCPI.SENSe(*Ch*).CORRection.TRIGger.FREE.STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.TRIGger.FREE.STATe

SCPI.TRIGger.SEQuence.SOURce

Status

/

(Boolean)

- “”
- “”

```
Dim TrigSour As Boolean
SCPI.SENSe(1).CORRection.TRIGger.FREE.STATe = False
TrigSour = SCPI.SENSe(1).CORRection.TRIGger.FREE.STATe
```

Cal > **Cal Trigger Source** > **Internal|System**

> > **|**

SCPI

:SENSe{[1]-36}:CORRection:TRIGger:FREE[:STATe] {ON|OFF|1|0}

:SENSe{[1]-36}:CORRection:TRIGger:FREE[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:CORR:TRIG:FREE OFF"
20 OUTPUT 717;":SENS1:CORR:TRIG:FREE?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.TYPE(*Tr*)

Data = SCPI.SENSe(*Ch*).CORRection.TYPE(*Tr*)

(*Ch*)(*Tr*)

	Data
	5 • *(0)* • *(1)* 0 • *(2)* SOLT2SOLT3SOLT40 • *(3)* SOLT3SOLT40 • *(4)* SOLT40 0
	(0) • “ERES” • “” • “RESPO” • “RESPS” • “RESPT” • “SMIX2” • “SOLT1”1 • “SOLT2”2 • “SOLT3”3 • “SOLT4”4
	(Variant)

```
Dim CalType As Variant
CalType = SCPI.SENSe(1).CORRection.TYPE(1)
```

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL2
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL3
SCPI.SENSe(Ch).CORRection.COLLect.METHod.TRL4

SCPI

:SENSe{[1]-36}:CORRection:TYPE{[1]-36}?

{ERES|NONE|RESPO|RESPS|RESPT|SOLT1|SOLT2|SOLT3|SOLT4|SMI
{numeric 1},{numeric 2}, {numeric 3},{numeric 4}<newline><^END>

{ERES|NONE|RESPO|RESPS|RESPT|SOLT1|SOLT2|SOLT3

ERES	
NONE	
RESPO	
RESPS	
RESPT	
SMIX2	
SOLT1	1
SOLT2	2
SOLT3	3
SOLT4	4

{1}

(0)
SMIX2

{2}

ERESRESPTSOLT2SOLT3SOLT4SMIX20

SMIX2

{3}

(SOLT3SOLT40)

{4}

(SOLT40)

```
10 OUTPUT 717;":SENS1:CORR:TYPE1?"
20 ENTER 717;A$
```

SCPI.SENSe(*Ch*).FREQuency.CENTer

SCPI.SENSe(*Ch*).FREQuency.CENTer = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.CENTer

/ (*Ch*)

	Value
	(Double)
	9E33E9230430 9E38.5E9280480 1E53E9235435 1E58.5E9285485
	4.25015E9
	Hz
	0.51

```
Dim Cntr As Double
SCPI.SENSe(1).FREQuency.CENTer = 2E9
Cntr = SCPI.SENSe(1).FREQuency.CENTer
```

SCPI.SENSe(Ch).FREQuency.SPAN

Center

SCPI

:SENSe{[1]-36}:FREQuency:CENTer <numeric>

:SENSe{[1]-36}:FREQuency:CENTer?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:FREQ:CENT 2E9"
20 OUTPUT 717;":SENS1:FREQ:CENT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.CW

SCPI.SENSe(*Ch*).FREQuency.CW = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.CW

/136 (*Ch*)CW

SCPI.SENSe(Ch).FREQuency.FIXed

	Value
	(Double)
	9E33E9230235430435 1E58.5E9280285480485
	1E5
	Hz
	1

```
Dim CwFreq As Double
SCPI.SENSe(1).FREQuency.CW = 1E9
CwFreq = SCPI.SENSe(1).FREQuency.CW
```

SCPI.SENSe(Ch).FREQuency.FIXed

SCPI.SENSe(Ch).SWEep.TYPE

Sweep Setup > **Power** > **CW Freq**

> > **CW**

SCPI

:SENSe{[1]-36}:FREQuency[:CW|FIXed] <numeric>

:SENSe{[1]-36}:FREQuency[:CW|FIXed]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:FREQ 1E9"
20 OUTPUT 717;":SENS1:FREQ?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.DATA

Data = SCPI.SENSe(Ch).FREQuency.DATA

136(Ch)

	Data
	NOPn1NOP • *n-1* n 0
	(Variant)

```
Dim FreqData As Variant
SCPI.SENSe(1).SWEep.POINts = 201
FreqData = SCPI.SENSe(1).FREQuency.DATA
```

SCPI.SENSe(Ch).SWEep.POINts

SCPI

:SENSe{[1]-36}:FREQuency:DATA?

{numeric 1},?E/span> ,{numeric NOP}<newline><^END>

{n} n

NOPn1NOP

```
10 DIM A(1:201)
20 OUTPUT 717;":SENS1:FREQ:DATA?"
30 ENTER 717;A(*)
```

SCPI.SENSe(*Ch*).FREQuency.FIXed

SCPI.SENSe(*Ch*).FREQuency.FIXed = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.FIXed

/136 (*Ch*)CW

SCPI.SENSe(Ch).FREQuency.CW

	Value
	(Double)
	9E33E9230430 9E38.5E9280480 1E53E9235435 1E58.5E9285485
	1E5
	Hz
	1

```
Dim CwFreq As Double
SCPI.SENSe(1).FREQuency.FIXed = 1E9
CwFreq = SCPI.SENSe(1).FREQuency.FIXed
```

SCPI.SENSe(Ch).FREQuency.CW
SCPI.SENSe(Ch).SWEep.TYPE

Sweep Setup > **Power** > **CW Freq**

> > **CW**

SCPI

:SENSe{[1]-36}:FREQuency[:CW|FIXed] <numeric>

:SENSe{[1]-36}:FREQuency[:CW|FIXed]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:FREQ 1E9"
20 OUTPUT 717;":SENS1:FREQ?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.SPAN

SCPI.SENSe(*Ch*).FREQuency.SPAN = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.SPAN

/ (*Ch*)

	Value
	(Double)
	08.4997E9
	8.4997E9
	Hz
	1

```
Dim Span As Double
SCPI.SENSe(1).FREQuency.SPAN = 1E9
Span = SCPI.SENSe(1).FREQuency.SPAN
```

SCPI.SENSe(Ch).FREQuency.CENTer

Span

SCPI

:SENSe{[1]-36}:FREQuency:SPAN <numeric>

:SENSe{[1]-36}:FREQuency:SPAN?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:FREQ:SPAN 1E9"
20 OUTPUT 717;":SENS1:FREQ:SPAN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.STARt

SCPI.SENSe(*Ch*).FREQuency.STARt = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.STARt

/ (*Ch*)

	Value
	(Double)
	9E33E9230430 9E38.5E9280480 1E53E9235435 1E58.5E9285485
	1E5
	Hz
	1

```
Dim Start As Double
SCPI.SENSe(1).FREQuency.STARt = 100E6
Start = SCPI.SENSe(1).FREQuency.STARt
```

SCPI.SENSe(Ch).FREQuency.STOP

Start

SCPI

:SENSe{[1]-36}:FREQuency:STARt <numeric>

:SENSe{[1]-36}:FREQuency:STARt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:FREQ:STAR 100E6"
20 OUTPUT 717;":SENS1:FREQ:STAR?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.STOP

SCPI.SENSe(*Ch*).FREQuency.STOP = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.STOP

/ (*Ch*)

	Value
	(Double)
	9E33E9230430 9E38.5E9280480 1E53E9235435 1E58.5E9285485
	8.5E9
	Hz
	1

```
Dim Stp As Double
SCPI.SENSe(1).FREQuency.STOP = 3E9
Stp = SCPI.SENSe(1).FREQuency.STOP
```

SCPI.SENSe(Ch).FREQuency.STARt

Stop

SCPI

:SENSe{[1]-36}:FREQuency:STOP <numeric>

:SENSe{[1]-36}:FREQuency:STOP?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:FREQ:STOP 100E6"
20 OUTPUT 717;":SENS1:FREQ:STOP?"
30 ENTER 717;A
```

SCPI.SENSe(Ch).MULTiplexer(*Id*).COUNt

Value = SCPI.SENSe(Ch).MULTiplexer(*Id*).COUNt

IDIdE5091A916

	Id
	E5091AID
	(Long)
	12
	1

	Value
	(Long)
	E5091A0

```
Dim NPort As Long
NPort = SCPI.SENSe.MULTiplexer(1).COUNt
```

SCPI

:SENSe:MULTiplexer{[1]|2}:COUNt?

{numeric}<newline><^END>

0 is read when the E5091A is not connected.

```
10 OUTPUT 717;":SENS:MULT1:COUN?"
20 ENTER 717;A
```

SCPI.SENSe(Ch).MULTiplexer(*Id*).DISPlay.STATe

SCPI.SENSe(Ch).MULTiplexer(*Id*).DISPlay.STATe = *Status*

Status = SCPI.SENSe(Ch).MULTiplexer(*Id*).DISPlay.STATe

/ID *Id*E5091A

	Status
	/
	(Boolean)
	• •

```
Dim Prop As Boolean
SCPI.SENSe.MULTiplexer(1).DISPlay.STATe = True
Prop = SCPI.SENSe.MULTiplexer(1).DISPlay.STATe
```

SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT1
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT2
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT3
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT4
SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).CATalog
SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).SELect

[System] > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Property**

[] > > **1|2** >

SCPI

:SENSe:MULTiplexer{[1]|2}:DISPlay[:STATe] {ON|OFF|1|0}

:SENSe:MULTiplexer{[1]|2}:DISPlay[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS:MULT1:DISP ON"
20 OUTPUT 717;":SENS:MULT1:DISP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).OUTPut.DATA

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).OUTPut.DATA = *Value*

Value = SCPI.SENSe(*Ch*).MULTiplexer(*Id*).OUTPut.DATA

E5091A(*Ch*)/ID12 (*Id*)E5091A/

1/081LSB8MSB

	Value
	/
	(Long)
	0255
	0

```
Dim OData As Long
SCPI.SENSe.MULTiplexer(1).OUTPut.DATA = "E5091_16"
MultName = SCPI.SENSe.MULTiplexer(1).OUTPut.DATA
```

SCPI.SENSe(Ch).MULTiplexer(Id).STATe

System > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Control Lines**

> > **1|2** >

SCPI

:SENSe{[1]-36}:MULTiplexer{[1]|2}:OUTPut[:DATA] <numeric>

:SENSe{[1]-36}:MULTiplexer{[1]|2}:OUTPut[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:MULT1:OUTP 5"
20 OUTPUT 717;":SENS1:MULT1:OUTP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).PORT(*Pt*).CATalog

String = SCPI.SENSe.MULTiplexer(*Id*).PORT(*Pt*).CATalog

E5091A(*Ch*)ID2 (*Id*)E5091A120(*Pt*)

```
Dim PortCat As String
PortCat = SCPI.SENSe.MULTiplexer(1).PORT(4).CATalog
```

SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT1
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT2
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT3
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT4
SCPI.SENSe(Ch).MULTiplexer(Id).DISPlay.STATe
SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).SELect

SCPI

:SENSe{[1]-36}:MULTiplexer{[1]|2}:PORT{[1]|2|3|4|5|6|7|8|9|10|11|12|13|1

{string}<newline><^END>

```
10 OUTPUT 717;":SENS1:MULT1:PORT1:CAT?"
20 ENTER 717;A$
```

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).PORT(*Pt*).SELect

SCPI.SENSe.MULTiplexer(*Id*).PORT(*Pt*).SELect = *String*

String = SCPI.SENSe.MULTiplexer(*Id*).PORT(*Pt*).SELect

/E5091AID *Id*

E5091A-009

	Status			
	/E5091A			
	(Boolean)			
	1	2	3	4
	AT1	**T1T2**	**R1+ R2+ R3+**	**R1- R2-R3-**
	A	T1	R1+	R1-

E5091A-013

	Status			
	/E5091A			
	(Boolean)			
	1	2	3	4
	A1T1 T2T3	**T1 T2 T3T4**	**R1+ R2+ R3+ R4+**	**R1- R2- R3-R4-**
	A	T1	R1+	R1-

E5091A-016

	Status
	/E5091A
	(Boolean)

	1	2	3	4	5	6	7
	A1(A) A2 A3 A4T1 T2T3	**B1(T4) B2B3 B4T1 T2T3**	**R1+ R2+ R3+ R4+**	**R1- R2- R3-R4-**	**X1 X2**	**Y1 Y2**	**Z1Z2**
	A1	B1	R1+	R1-	X1	Y1	Z1

(*Ch*)(*Id*)(*Pt*) ChIdPt

```
Dim PortSel As String
SCPI.SENSe.MULTiplexer(1).PORT(4).SELect = "R2"
PortSel = SCPI.SENSe.MULTiplexer(1).PORT(4).SELect
```

SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT1
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT2
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT3
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT4
SCPI.SENSe(Ch).MULTiplexer(Id).DISPlay.STATe
SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).CATalog
SCPI.SENSe(Ch).MULTiplexer(Id).STATe

System > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Port 1|Port 2|Port 3|Port 4|Port 5|Port 6|Port 7**

> > **1|2** > **1|2|3|4|5|6|7**

SCPI

:SENSe{[1]-36}:MULTiplexer{[1]|2}:PORT{[1]|2|3|4|5|6|7|8|9|10|11|12|13|1
[:SELect] <string>

:SENSe{[1]-36}:MULTiplexer{[1]|2}:PORT{[1]|2|3|4|5|6|7|8|9|10|11|12|13|1
[:SELect]?

{string}<newline><^END>

```
10 OUTPUT 717;":SENS1:MULT1:PORT1 A"
20 OUTPUT 717;":SENS1:MULT1:PORT1?"
30 ENTER 717;A$
```

SCPI.SENSe(Ch).MULTiplexer(*Id*).STATe

SCPI.SENSe(Ch).MULTiplexer(*Id*).STATe = *Status*

Status = SCPI.SENSe(Ch).MULTiplexer(*Id*).STATe

/E5091AID *Id*

	Status
	/E5091A
	(Boolean)
	• E5091A • 5091A

```
Dim Cont As Boolean
SCPI.SENSe.MULTiplexer(1).STATe = True
Cont = SCPI.SENSe.MULTiplexer(1).STATe
```

SCPI.SENSe(Ch).MULTiplexer(Id).OUTPut.DATA
SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).SELect
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT1
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT2
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT3
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT4
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.OUTPut.DATA

System > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Control**
> > **1|2** >

SCPI

:SENSe:MULTiplexer{[1]|2}:STATe {ON|OFF|1|0}

:SENSe:MULTiplexer{[1]|2}:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS:MULT1:STAT ON"
20 OUTPUT 717;":SENS:MULT1:STAT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.OUTPut.DATA

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.OUTPut.DATA = *Value*

Value = SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.OUTPut.DATA

E5091A(*Ch*)/E5091AID *Id* /

1/081LSB2MSB

	Value
	(Long)
	0255
	0
	1

```
Dim C_line As Long
SCPI.SENSe(1).MULTiplexer(1).TSET9.OUTPut.DATA = 5
C_line = SCPI.SENSe(1).MULTiplexer(1).TSET9.OUTPut.DATA
```

SCPI.SENSe(Ch).MULTiplexer(Id).STATe

System > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Control Lines** > **Line 1|...|Line 8**

> > **1|2** > > **1|___|8**

SCPI

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:OUTPut[:DATA] <numeric>

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:OUTPut[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:MULT1:TSET9:OUTP 5"
20 OUTPUT 717;":SENS1:MULT1:TSET9:OUTP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT1

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT1 = *Param*

Param = SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT1

E5091A(*Ch*)/E5091AID *Id*1

2T1T112T2

	Param
	1
	(String)
	• “A”A1 • “T1”T11
	A

```
Dim Port As String
SCPI.SENSe(1).MULTiplexer(1).TSET9.PORT1 = "T1"
Port = SCPI.SENSe(1).MULTiplexer(1).TSET9.PORT1
```

SCPI.SENSe(Ch).MULTiplexer(Id).STATe
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT2
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT3
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT4
SCPI.SENSe(Ch).MULTiplexer(Id).DISPlay.STATe
SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).CATalog

System > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Port 1** > **A|T1**

> > **1|2** > **1** > **A|T1**

SCPI

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:PORT1 {A|T1}

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:PORT1?

{A|T1}<newline><^END>

```
10 OUTPUT 717;":SENS1:MULT1:TSET9:PORT1 T1"
20 OUTPUT 717;":SENS1:MULT1:TSET9:PORT1?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT2

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT2 = *Param*

Param = SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT2

E5091A(*Ch*)/E5091AID *Id*2

1T1T121A

	Param
	2
	(String)
	• “T1”T12 • “T2”T22
	T1

```
Dim Port As String
SCPI.SENSe(1).MULTiplexer(1).TSET9.PORT2 = "T2"
Port = SCPI.SENSe(1).MULTiplexer(1).TSET9.PORT2
```

SCPI.SENSe(Ch).MULTiplexer(Id).STATe
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT1
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT3
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT4
SCPI.SENSe(Ch).MULTiplexer(Id).DISPlay.STATe
SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).CATalog

System > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Port2** > **T1|T2**

> > **1|2** > **2** >**T1|T2**

SCPI

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:PORT2 {T1|T2}

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:PORT2?

{A|T1}<newline><^END>

```
10 OUTPUT 717;":SENS1:MULT1:TSET9:PORT2 T2"
20 OUTPUT 717;":SENS1:MULT1:TSET9:PORT2?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT3

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT3 = *Param*

Param = SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT3

E5091A*(Ch)*/E5091AID *Id*3

	Param
	3
	(String)
	• “R1”R1+3 • “R2”R2+3 • “R3” R3+3
	R1

```
Dim Port As String
SCPI.SENSe(1).MULTiplexer(1).TSET9.PORT3 = "R2"
Port = SCPI.SENSe(1).MULTiplexer(1).TSET9.PORT3
```

SCPI.SENSe(Ch).MULTiplexer(Id).STATe
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT1
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT2
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT4
SCPI.SENSe(Ch).MULTiplexer(Id).DISPlay.STATe
SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).CATalog

System > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Port3** > **R1+|R2+|R3+**

> > **1|2** > **3** > **R1+|R2+|R3+**

SCPI

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:PORT3 {R1|R2}R3}

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:PORT3?

{R1|R2}R3}<newline><^END>

```
10 OUTPUT 717;":SENS1:MULT1:TSET9:PORT3 R2"
20 OUTPUT 717;":SENS1:MULT1:TSET9:PORT3?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT4

SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT3 = *Param*

Param = SCPI.SENSe(*Ch*).MULTiplexer(*Id*).TSET9.PORT3

E5091A*(Ch)*/E5091AID *Id*4

	Param
	4
	(String)
	• “R1”R1- 4 • “R2”R2- 4 • “R3” R3-4
	R1

```
Dim Port As String
SCPI.SENSe(1).MULTiplexer(1).TSET9.PORT4 = "R2"
Port = SCPI.SENSe(1).MULTiplexer(1).TSET9.PORT4
```

SCPI.SENSe(Ch).MULTiplexer(Id).STATe
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT1
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT2
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT3
SCPI.SENSe(Ch).MULTiplexer(Id).DISPlay.STATe
SCPI.SENSe(Ch).MULTiplexer(Id).PORT(Pt).CATalog

System > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Port4** > **R1-|R2-|R3-**

> > **1|2** > **4** > **R1-|R2-|R3-**

SCPI

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:PORT4 {R1|R2|R3}

:SENSe{[1]-36}:MULTiplexer{[1]|2}:TSET9:PORT4?

{R1|R2}R3}<newline><^END>

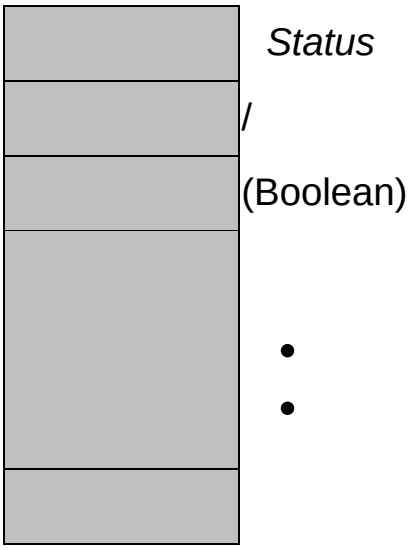
```
10 OUTPUT 717;":SENS1:MULT1:TSET9:PORT4 R2"
20 OUTPUT 717;":SENS1:MULT1:TSET9:PORT4?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.CONTrol.STATe

SCPI.SENSe(*Ch*).OFFSet.LOCal.CONTrol.STATe = *Status*

Status = SCPI.SENSe(*Ch*).OFFSet.LOCal.CONTrol.STATe

(Ch)/


Status
/
(Boolean)

```
Dim State As Boolean
SCPI.SENSe(1).OFFSet.LOCal.CONTrol.STATe = False
State = SCPI.SENSe(1).OFFSet.LOCal.CONTrol.STATe
```

Sweep Setup > **Frequency Offset** > **External Source** > **Control**

> > >

SCPI

:SENSe{[1]-36}:OFFSet:LOCal:CONTrol[:STATe] {ON|OFF|1|0}

:SENSe{[1]-36}:OFFSet:LOCal:CONTrol[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:LOC:CONT ON"
20 OUTPUT 717;":SENS1:OFFS:LOC:CONT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.DATA

Data = SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.DATA

(*Ch*)

	Data
	(Variant)

```
Dim LoData As Variant
LoData = SCPI.SENSe(1).OFFSet.LOCal.FREQuency.DATA
```

SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STARt
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STOP

SCPI

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:DATA?

{value 1},?E/span> ,{value NOP}<newline><^END>

NOTE Where, NOP is the number of points.

```
10 DIM A(1:201)
20 OUTPUT 717;":SENS1:OFFS:LOC:DATA?"
30 ENTER 717;A(*)
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.DIVisor

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.DIVisor = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.DIVisor

(Ch)/

	Value
	(Double)
	1100
	1

```
Dim Divisor As Double
SCPI.SENSe(1).OFFSet.LOCal.FREQuency.DIVisor = 50
Divisor = SCPI.SENSe(1).OFFSet.LOCal.FREQuency.DIVisor
```

SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STARt
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STOP

Sweep Setup > **Frequency Offset** > **External Source** > **Divisor**

> > >

SCPI.SENSe(Ch).FREQuency.STARt
SCPI.SENSe(Ch).FREQuency.STOP

SCPI

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:DIVisor <numeric>

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:DIVisor?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:LOC:DIV 1E1"
20 OUTPUT 717;":SENS1:OFFS:LOC:DIV?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.MULTiplier

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.MULTiplier = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.MULTiplier

(Ch)/

	Value
	(Double)
	-100100
	0

```
Dim Multiplier As Double
SCPI.SENSe(1).OFFSet.LOCal.FREQuency.MULTiplier = -10
Multiplier = SCPI.SENSe(1).OFFSet.LOCal.FREQuency.MULTiplier
```

SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STARt
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STOP

Sweep Setup > **Frequency Offset** > **External Source** > **Multiplier**

> > >

“SCPI.SENSe(Ch).FREQuency.STARt”“SCPI.SENSe(Ch).FRE

SCPI

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:MULTiplier <numeric>

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:MULTiplier?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:LOC:MULT -1E1"
20 OUTPUT 717;":SENS1:OFFS:LOC:MULT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.OFFSet

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.OFFSet = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.OFFSet

(Ch)/

	Value
	(Double)
	-1E121E12
	0
	Hz
	1

```
Dim Offset As Double
SCPI.SENSe(1).OFFSet.LOCal.FREQuency.OFFSet = -10
Offset = SCPI.SENSe(1).OFFSet.LOCal.FREQuency.OFFSet
```

SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STARt
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STOP

Sweep Setup > **Frequency Offset** > **External Source** > **Offset**

> > >

“SCPI.SENSe(Ch).FREQuency.STARt”“SCPI.SENSe(Ch).FRE

SCPI

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:OFFSet <numeric>

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:OFFSet?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:LOC:OFFS 5E9"
20 OUTPUT 717;":SENS1:OFFS:LOC:OFFS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.STARt

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.STARt = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.STARt

(Ch)/

	Value
	(Double)
	01E12
	0
	Hz
	1

```
Dim Start As Double
SCPI.SENSe(1).OFFSet.LOCal.FREQuency.STARt = 100E6
Start = SCPI.SENSe(1).OFFSet.LOCal.FREQuency.STARt
```

SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STOP

Sweep Setup > **Frequency Offset** > **External Source** > **Start**

> > >

SCPI

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:STARt <numeric>

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:STARt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:LOC:STAR 5E9"
20 OUTPUT 717;":SENS1:OFFS:LOC:STAR?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.STOP

SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.STOP = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.LOCal.FREQuency.STOP

(Ch)/

	Value
	(Double)
	01E12
	0
	Hz
	1

```
Dim Stop As Double
SCPI.SENSe(1).OFFSet.LOCal.FREQuency.STOP = 100E6
Stop = SCPI.SENSe(1).OFFSet.LOCal.FREQuency.STOP
```

SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.LOCal.FREQuency.STARt

Sweep Setup > **Frequency Offset** > **External Source** > **Stop**

> > >

SCPI

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:STOP <numeric>

:SENSe{[1]-36}:OFFSet:LOCal[:FREQuency]:STOP?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:LOC:STOP 5E9"
20 OUTPUT 717;":SENS1:OFFS:LOC:STOP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.POWer.LEVel.IMMediate.AM

SCPI.SENSe(*Ch*).OFFSet.LOCal.POWer.LEVel.IMMediate.AMPLitude = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.LOCal.POWer.LEVel.IMMediate.AMPLitude

(Ch)/

	Value
	(Double)
	-15030
	-10
	dBm
	0.01

```
Dim Power As Double
SCPI.SENSe(1).OFFSet.LOCal.POWer.LEVel.IMMediate.AMPLitude =
-10
Power =
SCPI.SENSe(1).OFFSet.LOCal.POWer.LEVel.IMMediate.AMPLitude
```

SCPI.SENSe(Ch).OFFSet.LOCal.POWer.LEVel.SLOPe.DATA
SCPI.SENSe(Ch).OFFSet.LOCal.POWer.LEVel.SLOPe.STATe

Sweep Setup > **Frequency Offset** > **External Source** > **Power**

> > >

SCPI

:SENSe{[1]-36}:OFFSet:LOCal:POWer[:LEVel][:IMMediate] [:AMPLitude] <numeric>

:SENSe{[1]-36}:OFFSet:LOCal:POWer[:LEVel][:IMMediate] [:AMPLitude]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:LOC:POW -12.5"
20 OUTPUT 717;":SENS1:OFFS:LOC:POW?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.POWer.LEVel.SLOPe.DATA

SCPI.SENSe(*Ch*).OFFSet.LOCal.POWer.LEVel.SLOPe.DATA = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.LOCal.POWer.LEVel.SLOPe.DATA

(Ch)/

	Value
	(Double)
	-22
	0
	dB/GHz
	0.01

```
Dim Slope As Double
SCPI.SENSe(1).OFFSet.LOCal.POWer.LEVel.SLOPe.DATA = -1
Slope = SCPI.SENSe(1).OFFSet.LOCal.POWer.LEVel.SLOPe.DATA
```

SCPI.SENSe(Ch).OFFSet.LOCal.POWer.LEVel.IMMediate.AMPLitude
SCPI.SENSe(Ch).OFFSet.LOCal.POWer.LEVel.SLOPe.STATe

Sweep Setup > **Frequency Offset** > **External Source** > **Slope**

> > >

SCPI

:SENSe{[1]-36}:OFFSet:LOCal:POWer[:LEVel]:SLOPe[:DATA] <numeric>

:SENSe{[1]-36}:OFFSet:LOCal:POWer[:LEVel]:SLOPe[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:LOC:POW:SLOP 1.0"
20 OUTPUT 717;":SENS1:OFFS:LOC:POW:SLOP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.POWer.LEVel.SLOPe.STATe

SCPI.SENSe(*Ch*).OFFSet.LOCal.POWer.LEVel.SLOPe.STATe = *Status*

Status = SCPI.SENSe(*Ch*).OFFSet.LOCal.POWer.LEVel.SLOPe.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim State As Boolean
SCPI.SENSe(1).OFFSet.LOCal.POWer.LEVel.SLOPe.STATe = False
State = SCPI.SENSe(1).OFFSet.LOCal.POWer.LEVel.SLOPe.STATe
```

SCPI.SENSe(Ch).OFFSet.LOCal.POWer.LEVel.IMMediate.AMPLitude
SCPI.SENSe(Ch).OFFSet.LOCal.POWer.LEVel.SLOPe.DATA

Sweep Setup > **Frequency Offset** > **External Source** > **Slope**

> > >

SCPI

:SENSe{[1]-36}:OFFSet:LOCal:POWer[:LEVel]:SLOPe:STATe {ON|OFF|1|0}

:SENSe{[1]-36}:OFFSet:LOCal:POWer[:LEVel]:SLOPe:STATe?

{1|0}<newline><^END>

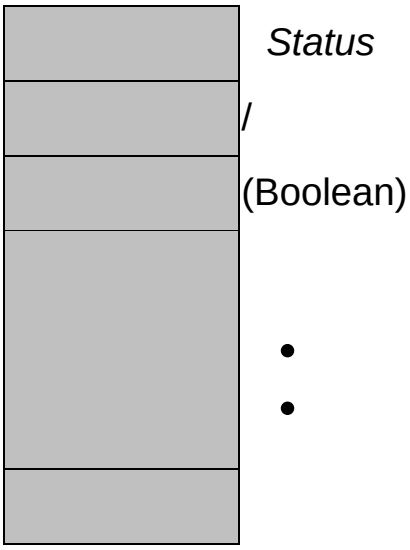
```
10 OUTPUT 717;":SENS1:OFFS:LOC:POW:SLOP:STAT ON"
20 OUTPUT 717;":SENS1:OFFS:LOC:POW:SLOP:STAT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.LOCal.STATe

SCPI.SENSe(*Ch*).OFFSet.LOCal.STATe = *Status*

Status = SCPI.SENSe(*Ch*).OFFSet.LOCal.STATe

(Ch)/


Status
/
(Boolean)

```
Dim State As Boolean
SCPI.SENSe(1).OFFSet.LOCal.STATe = False
State = SCPI.SENSe(1).OFFSet.LOCal.STATe
```

Setup > **Frequency Offset** > **External Source** > **LO Frequency**

> > >

SCPI

:SENSe{[1]-36}:OFFSet:LOCal:STATe {ON|OFF|1|0}

:SENSe{[1]-36}:OFFSet:LOCal:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:LOC:STAT ON"
20 OUTPUT 717;":SENS1:OFFS:LOC:STAT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.DATA

Data = SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.DATA

(*Ch*)(*Pt*)

	Data
	(Variant)

```
Dim Freqdata As Variant
Freqdata = SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.DATA
```

SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STARt
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STOP

SCPI

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency]:DATA?

{value 1},...,{value NOP}<newline><^END>

Where, NOP is the number of points.

```
10 DIM A(1:201)
20 OUTPUT 717;":SENS1:OFFS:PORT2:DATA?"
30 ENTER 717;A(*)
```

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.DIVisor

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.DIVisor = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.DIVisor

(*Ch*)/ (*Pt*).

	Value
	(Double)
	1100
	1

```
Dim Divisor As Double
SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.DIVisor = 50
Divisor = SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.DIVisor
```

SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STARt
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STOP

Sweep Setup > **Frequency Offset** > **Port n** > **Divisor**

> > >

“SCPI.SENSe(Ch).FREQuency.STARt”“SCPI.SENSe(Ch).FRE

SCPI

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency]:DIVisor <numeric>

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency]:DIVisor?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:PORT2:DIV 1E1"
20 OUTPUT 717;":SENS1:OFFS:PORT2:DIV?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.MULTiplier

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.MULTiplier = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.MULTiplier

(*Ch*)/ (*Pt*)

	Value
	-100100
	1

```
Dim Multiplier As Double
SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.MULTiplier = -10
Multiplier = SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.MULTiplier
```

SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STARt
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STOP

Sweep Setup > **Frequency Offset** > **Port n** > **Multiplier**

> > >

"SCPI.SENSe(Ch).FREQuency.STARt""SCPI.SENSe(Ch).FRE

SCPI

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency] :MULTiplier <numeric>

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency] :MULTiplier?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:PORT2:MULT 1E1"
20 OUTPUT 717;":SENS1:OFFS:PORT2:MULT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.OFFSet

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.OFFSet = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.OFFSet

(*Ch*)/ (*Pt*)

	Value
	(Double)
	-1E121E12
	0
	Hz
	1

```
Dim Offset As Double
SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.OFFSet = 1E9
Offset = SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.OFFSet
```

SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STARt
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STOP

Sweep Setup > **Frequency Offset** > **Port n** > **Offset**

> > **n** >

“SCPI.SENSe(Ch).FREQuency.STARt”“SCPI.SENSe(Ch).FRE

SCPI

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency]:OFFSet <numeric>

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency]:OFFSet?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:PORT2:OFFS 1E9"
20 OUTPUT 717;":SENS1:OFFS:PORT2:OFFS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.STARt

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.STARt = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.STARt

(*Ch*)/ (*Pt*)

	Value
	(Double)
	9E33E9230430 9E38.5E9280480 1E53E9235435 1E58.5E9285485
	1E5
	Hz
	1

```
Dim Start As Double
SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.STARt = 100E6
Start = SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.STARt
```

SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STOP

Sweep Setup > Frequency Offset > Port n > Start

> > **n** >

SCPI

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency]:STARt <numeric>

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency]:STARt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:PORT2:STAR 300000"
20 OUTPUT 717;":SENS1:OFFS:PORT2:STAR?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.STOP

SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.STOP = *Value*

Value = SCPI.SENSe(*Ch*).OFFSet.PORT(Pt).FREQuency.STOP

(*Ch*)/ (*Pt*)

	Value
	(Double)
	9E33E9230430 9E38.5E9280480 1E53E9235435 1E58.5E9285485
	8.5E9 (3E9)
	Hz
	1

```
Dim Stop As Double
SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.STOP = 100E6
Stop = SCPI.SENSe(1).OFFSet.PORT(1).FREQuency.STOP
```

SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.DIVisor
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.MULTiplier
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.OFFSet
SCPI.SENSe(Ch).OFFSet.PORT(Pt).FREQuency.STARt

Sweep Setup > **Frequency Offset** > **Port n** > **Stop**

> > **n** >

SCPI

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency]:STOP <numeric>

:SENSe{[1]-36}:OFFSet:PORT{[1]|2|3|4}[:FREQuency]:STOP?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS:PORT2:STOP 1e+009"
20 OUTPUT 717;":SENS1:OFFS:PORT2:STOP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).OFFSet.STATe

SCPI.SENSe(*Ch*).OFFSet.STATe = *Status*

Status = SCPI.SENSe(*Ch*).OFFSet.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim State As Boolean
SCPI.SENSe(1).OFFSet.STATe = False
State = SCPI.SENSe(1).OFFSet.STATe
```

Sweep Setup > **Frequency Offset** > **Frequency Offset**

> >

SCPI

:SENSe{[1]-36}:OFFSet[:STATe] {ON|OFF|1|0}

:SENSe{[1]-36}:OFFSet[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:OFFS ON"
20 OUTPUT 717;":SENS1:OFFS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).ROSCillator.SOURce

Param = SCPI.SENSe(*Ch*).ROSCillator.SOURce

Ref In connector

	Param
	(String)
	• “” • “”

(*Ch*) Ch

```
Dim Ref As String
Ref = SCPI.SENSe(1).ROSCillator.SOURce
```

Displayed on the instrument status bar (at the bottom of the LCD display).

SCPI

:SENSe{[1]-36}:ROSCillator:SOURce?

{INTernal|EXTernal}<newline><^END>


```
10 OUTPUT 717;":SENS1:ROSC:SOUR?"
20 ENTER 717;A$
```

SCPI.SENSe(*Ch*).SEGMent.DATA

SCPI.SENSe(*Ch*).SEGMent.DATA = *Data*

Data = SCPI.SENSe(*Ch*).SEGMent.DATA

(*Ch*)

	Data
	N<segm>n1N = {<buf><stim><ifbw><pow><del><swp><time> <segm> <star 1><stop 1><nop 1><ifbw 1> <pow 1><del 1><swp 1><time 1>... <star n> <stop n><nop n><ifbw n><pow n><del n><swp n> <time n> .... <star N><stop N><nop N><ifbw N> <pow N><del N><swp N><time N>} • <buf>566 • <stim> 0/ 1/ • <ifbw>/IF01 • <pow>/01 • <del>/01 • <swp>/01 <buf>5 • <time>/01 • <segm> 1201 • <star n>n/ • <stop n>n/ • <nop n>n • <ifbw n>IF<ifbw>0nIF • <pow n><pow>0n • <del n><del>0n • <swp n> n 0Stepped mode

注意	1Swept mode 2Stepped mode 3Swept mode <buf>5<del>0<buf>23E5070B/E5071B • <time n> <time>0n
	(Variant)
	<stim><ifbw><pow><del><swp><time> <star n><stop n><nop n><ifbw n><pow n> <time n>

```
Dim SegmData As Variant
SCPI.SENSe(1).SEGMent.DATA = Array(5,0,0,1,0,0,2, _
100E6,1E9,31,0,2E9,3E9,51,-10)
SegmData = SCPI.SENSe(1).SEGMent.DATA
Dim SegmData(14) As Variant
Dim Ref As Variant
SegmData(0) = 5
SegmData(1) = 0
SegmData(2) = 0
SegmData(3) = 1
SegmData(4) = 0
SegmData(5) = 0
SegmData(6) = 2
SegmData(7) = 100E6
SegmData(8) = 1E9
SegmData(9) = 31
SegmData(10) = 0
SegmData(11) = 2E9
SegmData(12) = 3E9
SegmData(13) = 51
SegmData(14) = -10
SCPI.SENSe(1).SEGMent.DATA = SegmData
Ref = SCPI.SENSe(1).SEGMent.DATA
```

SCPI.SENSe(Ch).SWEep.TYPE

SCPI

:SENSe{[1]-36}:SEGMent:DATA 5,<mode>,<ifbw>,<pow>,<del>,<time>,
<segm>,
<star 1>,<stop 1>,<nop 1>,<ifbw 1>,<pow 1>,<del 1>,<time 1>,?
E/span> ,
<star n>,<stop n>,<nop n>,<ifbw n>,<pow n>,<del n>,<time n>,?
E/span> ,
<star N>,<stop N>,<nop N>,<ifbw N>,<pow N>,<del N>,<time N>

:SENSe{[1]-36}:SEGMent:DATA 6,<mode>,<ifbw>,<pow>,<del>,<swp>,
<time>,<segm>,
<star 1>,<stop 1>,<nop 1>,<ifbw 1>,<pow 1>,<del 1>,<swp 1>,<time
1>,?E/span> ,
<star n>,<stop n>,<nop n>,<ifbw n>,<pow n>,<del n>,<swp n>,<time
n>,?E/span> ,
<star N>,<stop N>,<nop N>,<ifbw N>,<pow N>,<del N>,<swp N>,<time
N>

:SENSe{[1]-36}:SEGMent:DATA?

N<segm>n1N

OFF:

5,{mode},{ifbw},{pow},{del},{time},{segm},
{star 1},{stop 1},{nop 1},{pow 1},{del 1},{time 1},?E/span> ,
{star n},{stop n},{nop n},{pow n},{del n},{time n},?E/span> ,
{star N},{stop N},{nop N},{pow N},{del N},{time N}<newline><^END>

ON:

6,{mode},{ifbw},{pow},{del},{swp},{time},{segm},
{star 1},{stop 1},{nop 1},{pow 1},{del 1},{swp 1},{time 1},?E/span> ,
{star n},{stop n},{nop n},{pow n},{del n},{swp n},{time n},?E/span> ,
{star N},{stop N},{nop N},{pow N},{del N},{swp N},{time N}<newline>
<^END>

```
10 DIM H(1:3,1:4)
20 OUTPUT 717;":SENS1:SEGM:DATA 5,0,1,0,0,0,3,";
30 OUTPUT 717;"1E9,3E9,11,70e3,";
40 OUTPUT 717;"3E9,4E9,51,7e3,";
50 OUTPUT 717;"4E9,6E9,11,70e3"
60 OUTPUT 717;":SENS1:SEGM:DATA?"
70 ENTER 717;A,B,C,D,E,F,G,H(*)
```

```
10 DIM H(1:3,1:5)
20 OUTPUT 717;":SENS1:SEGM:DATA 6,0,1,0,0,1,0,3,";
30 OUTPUT 717;"1E9,3E9,11,70e3,3,";
40 OUTPUT 717;"3E9,4E9,51,7e3,2,";
50 OUTPUT 717;"4E9,6E9,11,70e3,3"
60 OUTPUT 717;":SENS1:SEGM:DATA?"
70 ENTER 717;A,B,C,D,E,F,G,H(*)
```

SCPI.SENSe(*Ch*).SEGMent.SWEep.POINts

Value = SCPI.SENSe(*Ch*).SEGMent.SWEep.POINts

(*Ch*)

	Value
	(Long)
	2

```
Dim SegmPoin As Long
SegmPoin = SCPI.SENSe(1).SEGMent.SWEep.POINts
```

SCPI.SENSe(Ch).SEGMent.DATA

SCPI

:SENSe{[1]-36}:SEGMent:SWEep:POINts?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:SEGM:SWE:POIN?"
20 ENTER 717;A
```

SCPI.SENSe(*Ch*).SEGMent.SWEep.TIME.DATA

Value = SCPI.SENSe(*Ch*).SEGMent.SWEep.TIME.DATA

(*Ch*)

	Value
	(Double)

```
Dim SegmTime As Double
SegmTime = SCPI.SENSe(1).SEGMent.SWEep.TIME.DATA
```

SCPI.SENSe(Ch).SEGMent.DATA

SCPI

:SENSe{[1]-36}:SEGMent:SWEep:TIME[:DATA]?

{numeric}<newline><^END>

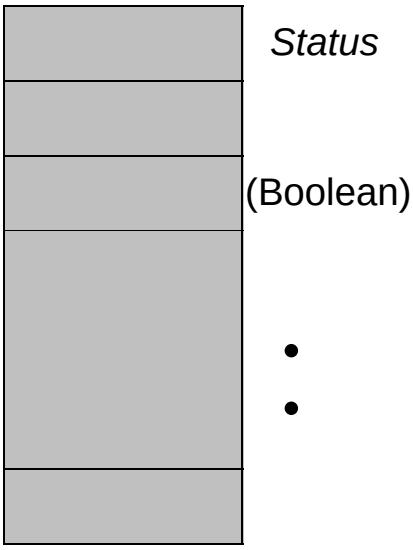
```
10 OUTPUT 717;":SENS1:SEGM:SWE:TIME?"
20 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.ASPurious

SCPI.SENSe(*Ch*).SWEep.ASPurious = *Status*

Status = SCPI.SENSe(*Ch*).SWEep.ASPurious

E5070B/E5071B


Status
(Boolean)

```
Dim ASpurious As Boolean
SCPI.SENSe(1).SWEep.ASPurious = False
ASpurious = SCPI.SENSe(1).SWEep.ASPurious
```

SCPI

:SENSe{[1]-36}:SWEep:ASPurious {ON|OFF|1|0}

:SENSe{[1]-36}:SWEep:ASPurious?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:SWE:ASP OFF"
20 OUTPUT 717;":SENS1:SWE:ASP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.DELay

SCPI.SENSe(*Ch*).SWEep.DELay = *Value*

Value = SCPI.SENSe(*Ch*).SWEep.DELay

/ (*Ch*)

	Value
	(Double)
	01
	0
	s
	0.001

```
Dim SweDel As Double
SCPI.SENSe(1).SWEep.DELay = 0.05
SweDel = SCPI.SENSe(1).SWEep.DELay
```

>

SCPI

:SENSe{[1]-36}:SWEep:DELay <numeric>

:SENSe{[1]-36}:SWEep:DELay?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:SWE:DEL 0.05"
20 OUTPUT 717;":SENS1:SWE:DEL?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.GENeration

SCPI.SENSe(*Ch*).SWEep.GENeration = *Param*

Param = SCPI.SENSe(*Ch*).SWEep.GENeration

/ (*Ch*)

POW SCPI.SENSe(Ch).SWEep.TYPEON
SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection.STATe0
ON SCPI.SOURce(Ch).POWer.LEVel.SLOPe.STATe

	Param
	(String)
	• “STEPped” • “ANALog” • “FSTepped”E5070B/E5071B • “FANalog”E5070B/E5071B
	“STEPped”

```
Dim SwptMode As String
SCPI.SENSe(1).SWEep.GENeration = "anal"
SwptMode = SCPI.SENSe(1).SWEep.GENeration
```

SCPI.SENSe(Ch).SWEep.TYPE
SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection.STATe
SCPI.SOURce(Ch).POWer.LEVel.SLOPe.STATe

Sweep Setup > **Sweep Mode** > **Stepped|Swept**

> > **|**

SCPI

:SENSe{[1]-36}:SWEep:GENeration
{STEPped|ANALog|FSTepped|FANalog}

:SENSe{[1]-36}:SWEep:GENeration?

{STEP|ANAL|FST|FAN}<newline><^END>

```
10 OUTPUT 717;":SENS1:SWE:GEN ANAL"
20 OUTPUT 717;":SENS1:SWE:GEN?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).SWEep.POINts

SCPI.SENSe(*Ch*).SWEep.POINts = *Value*

Value = SCPI.SENSe(*Ch*).SWEep.POINts

/ (*Ch*)

	Value
	(Long)
	220001
	201

```
Dim Nop As Long
SCPI.SENSe(1).SWEep.POINts = 801
Nop = SCPI.SENSe(1).SWEep.POINts
```

Sweep Setup > **Points**

>

SCPI

:SENSe{[1]-36}:SWEep:POINts <numeric>

:SENSe{[1]-36}:SWEep:POINts?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:SWE:POIN 801"
20 OUTPUT 717;":SENS1:SWE:POIN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.TIME.AUTO

SCPI.SENSe(*Ch*).SWEep.TIME.AUTO = *Status*

Status = SCPI.SENSe(*Ch*).SWEep.TIME.AUTO

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim SweAuto As Boolean
SCPI.SENSe(1).SWEep.TIME.AUTO = False
SweAuto = SCPI.SENSe(1).SWEep.TIME.AUTO
```

SCPI.SENSe(Ch).SWEep.TIME.DATA

Sweep Setup > **Sweep Time**

>

0s

SCPI

:SENSe{[1]-36}:SWEep:TIME:AUTO {ON|OFF|1|0}

:SENSe{[1]-36}:SWEep:TIME:AUTO?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SENS1:SWE:TIME:AUTO ON"
20 OUTPUT 717;":SENS1:SWE:TIME:AUTO?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.TIME.DATA

e

SCPI.SENSe(*Ch*).SWEep.TIME.DATA = *Value*

Value = SCPI.SENSe(*Ch*).SWEep.TIME.DATA

/ (*Ch*)

SCPI.SENSe(Ch).SWEep.TIME.AUTO

	Value
	(Double)
	s

```
Dim SweTime As Double
SCPI.SENSe(1).SWEep.TIME.AUTO = False
SCPI.SENSe(1).SWEep.TIME.DATA = 1.5
SweTime = SCPI.SENSe(1).SWEep.TIME.DATA
```

SCPI.SENSe(Ch).SWEep.TIME.AUTO

SCPI

:SENSe{[1]-36}:SWEep:TIME[:DATA] <numeric>

:SENSe{[1]-36}:SWEep:TIME[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS1:SWE:TIME 1.5"
20 OUTPUT 717;":SENS1:SWE:TIME?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.TYPE

SCPI.SENSe(*Ch*).SWEep.TYPE = *Param*

Param = SCPI.SENSe(*Ch*).SWEep.TYPE

/ (*Ch*)

	Param
	(String)
	• “LINear” • “LOGarithmic” • “SEGMent” • “POWer”
	“LINear”

```
Dim SweType As String
SCPI.SENSe(1).SWEep.TYPE = "segm"
SweType = SCPI.SENSe(1).SWEep.TYPE
```

Sweep Setup > **Sweep Type** > **Lin Freq|Log Freq|Segment**

> > **||**

SCPI

:SENSe{[1]-36}:SWEep:TYPE {LINear|LOGarithmic|SEGMent|POWer}

:SENSe{[1]-36}:SWEep:TYPE?

{LIN|LOG|SEGM|POW}<newline><^END>

```
10 OUTPUT 717;":SENS1:SWE:TYPE SEGM"
20 OUTPUT 717;":SENS1:SWE:TYPE?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).VOLTave(*Auxpt*).DC.RANGe.UPPer

SCPI.SENSe(*Ch*).VOLTave(Auxpt).DC.RANGe.UPPer=Param

Param=SCPI.SENSe(*Ch*).VOLTave(Auxpt).DC.RANGe.UPPer

/ (*Ch*)

	Param
	AUX
	(Double)
	(V)
	110
	10

	Auxpt
	AUX
	(Long)
	12

```
Dim DcRange As variant
SCPI.SENSe(1).VOLTave(2).DC.RANGe.UPPer=1
DcRange = SCPI.SENSe(1).VOLTave(1).DC.RANGe.UPPer
```

Meas > **Aux Input** (1 or 2) > **Range**
> **Aux** (12) >

SCPI

:SENSe{[1]-36}:VOLTave{[1]|2}:DC.RANGe:UPPer

:SENSe{[1]-36}:VOLTave{[1]|2}:DC.RANGe:UPPer?

{1|10}<newline><^END>

```
10 OUTPUT 717;":SENS1:VOLT1:DC.RANG:UPP 10"
20 OUTPUT 717;":SENS1:VOLT1:DC.RANG:UPP?"
30 ENTER 717;A
```

SCPI.SENSe.MULTiplexer(/*d*).INCount

Value = SCPI.SENSe.MULTiplexer(*Id*).INCount

E5091AID 12(*Id*)

```
Dim INCount As Long
INCount = SCPI.SENSe(1).MULTiplexer(1).INCount
```

SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT1
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT2
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT3
SCPI.SENSe(Ch).MULTiplexer(Id).TSET9.PORT4

SCPI

:SENSe:MULTiplexer{[1]|2}:INCount?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SENS:MULT1:INC?"
20 ENTER 717;A
```

SCPI.SENSe.MULTiplexer(*id*).NAME

SCPI.SENSe.MULTiplexer(*Id*).NAME = *Param*

Param = SCPI.SENSe.MULTiplexer(*Id*).NAME

/E5091A

	Param
	E5091A
	(String)
	• “E5091_9”E5091A009 • “E5091_13”13-E5091A016 • “E5091_16”16E5091A016
	E5091_9

Id Id

```
Dim MultName As String
SCPI.SENSe.MULTiplexer(1).NAME = "E5091_16"
MultName = SCPI.SENSe.MULTiplexer(1).NAME
```

S

SCPI.SENSe.MULTiplexer.CATalog
SCPI.SENSe.MULTiplexer(Id).INCount

System > **Multiport Test Set Setup** > **Test Set 1|Test Set 2** > **Select Test Set** > **E5091_9|E5091_13|E5091_16**

> > **1|2** > > **E5091_9|E5091_13|E5091_16**

SCPI

:SENSe:MULTiplexer{[1]|2}:NAME {E5091_9|E5091_13|E5091_16}

:SENSe:MULTiplexer{[1]|2}:NAME?

{E5091_9|E5091_13|E5091_16}<newline><^END>

```
10 OUTPUT 717;":SENS:MULT1:NAME E5091_9"
20 OUTPUT 717;":SENS:MULT1:NAME?"
30 ENTER 717;A$
```

SCPI.SENSe.MULTiplexer.CATalog

Param = SCPI.SENSe.MULTiplexer.CATalog

/E5091A

```
Dim MultCat As String
MultCat = SCPI.SENSe.MULTiplexer.CATalog
```

SCPI.SENSe.MULTiplexer(Id).NAME

SCPI

:SENSe:MULTiplexer{[1]|2}:CATalog?

{string}<newline><^END>

```
10 OUTPUT 717;":SENS:MULT:CAT?"
20 ENTER 717;A$
```

SCPI.SERVice.CHANnel(*Ch*).TRACe.ACTive

Value = SCPI.SERVice.CHANnel(*Ch*).TRACe.ACTive

(*Ch*)

	Value
	(Long)

```
Dim ActTrac As Long
ActTrac = SCPI.SERVice.CHANnel(1).TRACe.ACTive
```

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI

:SERVice:CHANnel{[1]-36}:TRACe:ACTive?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SERV:CHAN1:TRAC:ACT?"
20 ENTER 717;A
```

SCPI.SERVice.CHANnel.ACTive

Value = SCPI.SERVice.CHANnel.ACTive

	Value
	(Long)

```
Dim ActChan As Long
ActChan = SCPI.SERVice.CHANnel.ACTive
```

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI

:SERVice:CHANnel:ACTive?

{numeric}<newline><^END>

SCPI.SERVice.CHANnel.COUNt

Value = SCPI.SERVice.CHANnel.COUNt

E5071C

	Value
	(Long)

```
Dim MaxChan As Long
MaxChan = SCPI.SERVice.CHANnel.COUNt
```

SCPI

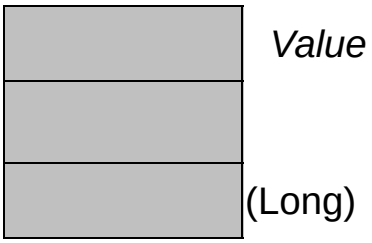
:SERVice:CHANnel:COUNt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SERV:CHAN:COUN?"
20 ENTER 717;A
```

SCPI.SERVice.CHANnel.TRACe.COUNt

Value = SCPI.SERVice.CHANnel.TRACe.COUNt


Value
(Long)

```
Dim MaxTrac As Long
MaxTrac = SCPI.SERVice.CHANnel.TRACe.COUNt
```

SCPI

:SERVice:CHANnel:TRACe:COUNt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SERV:CHAN:TRAC:COUN?"
20 ENTER 717;A
```

SCPI.SERVice.PORT.COUNt

Value = SCPI.SERVice.PORT.COUNt

E5071C

	Value
	(Long)

```
Dim MaxPort As Long
MaxPort = SCPI.SERVice.PORT.COUNt
```

SCPI

:SERVice:PORT:COUNt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SERV:PORT:COUN?"
20 ENTER 717;A
```

SCPI.SERVice.SREVision

Value = SCPI.SERVice.SREVision

E5071C

	Value
	1
	0
	(Long)

```
Dim SystemRevision As Long
SystemRevision = SCPI.SERVice.PORT.COUNt
```

SCPI

:SERVice:SREVision?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SERV:SREV?"
20 ENTER 717;A
```

SCPI.SERVice.SWEep.FREQency.MAXimum

Value = SCPI.SERVice.SWEep.FREQency.MAXimum

E5071C

	Value
	(Double)

```
Dim MaxFreq As Double
MaxFreq = SCPI.SERVice.SWEep.FREQency.MAXimum
```

SCPI

:SERVice:SWEep:FREQency:MAXimum?

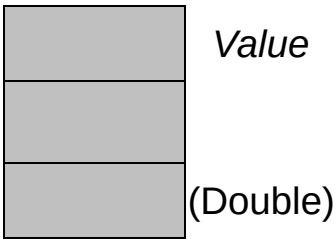
{numeric}<newline><^END>

```
10 OUTPUT 717;":SERV:SWE:FREQ:MAX?"
20 ENTER 717;A
```

SCPI.SERVice.SWEep.FREQency.MINimum

Value = SCPI.SERVice.SWEep.FREQency.MINimum

E5071C


Value
(Double)

```
Dim MinFreq As Double
MinFreq = SCPI.SERVice.SWEep.FREQency.MINimum
```

SCPI

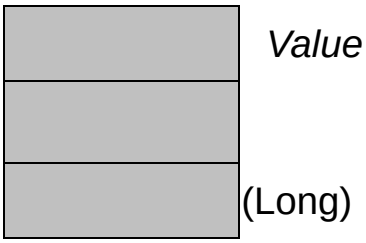
:SERVice:SWEep:FREQency:MINimum?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SERV:SWE:FREQ:MIN?"
20 ENTER 717;A
```

SCPI.SERVice.SWEep.POINts

Value = SCPI.SERVice.SWEep.POINts


Value
(Long)

```
Dim MaxPoin As Long
MaxPoin = SCPI.SERVice.SWEep.POINts
```

SCPI

:SERVice:SWEep:POINts?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SERV:SWE:POIN?"
20 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.ATTenuation.AUTO

()

SCPI.SOURce(*Ch*).POWer.ATTenuation.AUTO = *Status*
Status = SSCPI.SOURce(*Ch*).POWer.ATTenuation.AUTO

E5071CE5070B/E5071B

	Status
	(Boolean)
	• •

SCPI.SOURce(Ch).POWer.ATTenuation.DATA

SCPI

:SOURce{[1]-36}:POWer:ATTenuation:AUTO {ON|OFF|1|0}

:SOURce{[1]-36}:POWer:ATTenuation:AUTO?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:ATT:AUTO ON"
20 OUTPUT 717;":SOUR1:POW:ATT:AUTO?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.ATTenuation.DATA

()

SCPI.SOURce(*Ch*).POWer.ATTenuation.DATA = *Value*

Value = SCPI.SOURce(*Ch*).POWer.ATTenuation.DATA

E5071CE5070B/E5071B

	Value
	(Long)
	035
	0
	dB
	5

```
Dim Att As Long
SCPI.SOURce(1).POWer.ATTenuation.DATA = 10
Att = SCPI.SOURce(1).POWer.ATTenuation.DATA
```

SCPI.SOURce(Ch).POWer.ATTenuation.AUTO

SCPI

:SOURce{[1]-36}:POWer:ATTenuation[:DATA] <numeric>

:SOURce{[1]-36}:POWer:ATTenuation[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:ATT 15"
20 OUTPUT 717;":SOUR1:POW:ATT?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.CENTer

()

SCPI.SOURce(*Ch*).POWer.CENTer = *Value*

Value = SCPI.SOURce(*Ch*).POWer.CENTer

136*(Ch)*/

	Value
	(Double)
	-7.5
	dBm
	0.050.025

```
Dim Pcntr As Double
SCPI.SOURce(1).POWer.CENTer = 0
Pcntr = SCPI.SOURce(1).POWer.CENTer
```

SCPI.SENSe(Ch).SWEep.TYPE
SCPI.SOURce(Ch).POWer.ATTenuation.DATA
SCPI.SOURce(Ch).POWer.SPAN

Center

SCPI

:SOURce{[1]-36}:POWer:CENTer <numeric>

:SOURce{[1]-36}:POWer:CENTer?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:CENT 0"
20 OUTPUT 717;":SOUR1:POW:CENT?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.LEVel.IMMediate.AMPLitude

()

SCPI.SOURce(*Ch*).POWer.LEVel.IMMediate.AMPLitude = *Value*

Value = SCPI.SOURce(*Ch*).POWer.LEVel.IMMediate.AMPLitude

/ (*Ch*)

	Value
	(Double)
	0
	dBm
	0.05

```
Dim PowLev As Double
SCPI.SOURce(1).POWer.LEVel.IMMediate.AMPLitude = -10
PowLev = SCPI.SOURce(1).POWer.LEVel.IMMediate.AMPLitude
```

SCPI.SOURce(Ch).POWer.ATTenuation.DATA

SCPI.SOURce(Ch).POWer.ATTenuation.AUTO

Sweep Setup > Power

>

SCPI

:SOURce{[1]-36}:POWer[:LEVel][:IMMediate]
[:AMPLitude] <numeric>

:SOURce{[1]-36}:POWer[:LEVel][:IMMediate]
[:AMPLitude]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW -12.5"
20 OUTPUT 717;":SOUR1:POW?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.DATA

()

SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.DATA = *Value*

Value = SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.DATA

/136 *(Ch)*

	Value
	(Double)
	-22
	0
	dB/GHz
	0.01

```
Dim SlopLev As Double
SCPI.SOURce(1).POWer.LEVel.SLOPe.DATA = 0.1
SlopLev = SCPI.SOURce(1).POWer.LEVel.SLOPe.DATA
```

SCPI.SOURce(Ch).POWer.LEVel.SLOPe.STATe

Sweep Setup > **Power** > **Slop [xxx dB/GHz]**

> > **[xxx dB/GHz]**

SCPI

:SOURce{[1]-36}:POWer[:LEVel]:SLOPe[:DATA] <numeric>

:SOURce{[1]-36}:POWer[:LEVel]:SLOPe[:DATA]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:SLOP 0.1"
20 OUTPUT 717;":SOUR1:POW:SLOP?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.STATe

()

SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.STATe = *Status*

Status = SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.STATe

(Ch)/

	Status
	/
	(Boolean)
	• •

```
Dim Slop As Boolean
SCPI.SOURce(1).POWer.LEVel.SLOPe.STATe = True
Slop = SCPI.SOURce(1).POWer.LEVel.SLOPe.STATe
```

SCPI.SOURce(Ch).POWer.LEVel.SLOPe.DATA

SCPI.SENSe(Ch).SWEep.GENeration

Sweep Setup > **Power** > **Slop [ON/OFF]**

> > **[/]**

SCPI

:SOURce{[1]-36}:POWer[:LEVel]:SLOPe:STATe {ON|OFF|1|0}

:SOURce{[1]-36}:POWer[:LEVel]:SLOPe:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:SLOP:STAT ON"
20 OUTPUT 717;":SOUR1:POW:SLOP:STAT?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.A

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.ACQuire = *Param*

(*Ch*)(*Pt*)

	Param
	(String)
	• “ASENsor”A • “BSENsor”B

```
Dim Dmy As Long
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.ACQuire =
"asen"
Dmy = SCPI.IEEE4882.OPC
```

SCPI.IEEE4882.OPC

Cal > **Power Calibration** > **Take Cal Sweep**

> >

SCPI

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection:COLLect
[:ACQuire] {ASENsor|BSENsor}

```
10 OUTPUT 717;":SOUR1:POW:PORT1:CORR:COLL ASEN"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.A

()

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.AVERage.C
= *Value*

Value =
SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.AVERage.C

/ (*Ch*)(*Pt*)

	Value
	(Long)
	1100
	1

```
Dim AvgCnt As Long
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.AVERage.CO
= 6
AvgCnt =
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.AVERage.CO
```

SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection.COLLect.ACQuire

Cal > **Power Calibration** > **Num of Readings**

> >

SCPI

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection:COLLect
:AVERage[:COUNt] <numeric>

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection:COLLect
:AVERage[:COUNt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:PORT1:CORR:COLL:AVER 6"
20 OUTPUT 717;":SOUR1:POW:PORT1:CORR:COLL:AVER?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.T

()

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.TABLe.LOSS
= *Data*

Data =
SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.TABLe.LOSS

(*Ch*)(*Pt*)/

	Data
	1+NumX2n1Num • *(0)* 01000(0) • *(nX2-1)* n(1 kHz500 GHz) • *(n*X2*)* n(-100 dB100 dB) 0
	(Variant)
	1+Num ()X2 (*n*X2-1)(*n* X2)

```
Dim Loss As Variant
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.TABLe.LOSS
= Array(2,1e8,0.5 ,1e9,0.8)
Loss =
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.TABLe.LOSS
'''Clear Loss Table
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.TABLe.LOSS
= Array(0)
Dim Loss(4) As Variant
Dim Ref As Variant
Loss(0) = 2
Loss(1) = 1e8
Loss(2) = 0.5
Loss(3) = 1e9
Loss(4) = 0.8
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.TABLe.LOSS
= Loss
Ref =
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.TABLe.LOSS
'''Clear Loss Table
Dim Loss(0) As Variant
Loss(0) = 0
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.TABLe.LOSS
= Loss
```

SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection.COLLect.TABLe.LOSS

Cal > **Power Calibration** > **Loss Compen** > **Delete | Add | Clear Loss Table**

> > > ||

SCPI

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection:COLLect
:TABLe:LOSS:DATA <numeric 1>,...,<numeric 1+(Nx2)>

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection:COLLect
:TABLe:LOSS:DATA?

{numeric 1},...,{numeric 1+(Nx2)}<newline><^END>

```
10 DIM B(1:2,1:2)
20 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:TABL:LOSS:DATA
2,";
30 OUTPUT 717;"1E8,0.5,1E9,0.8"
40 OUTPUT
717;":SOUR:POW:PORT:CORR:COLL:TABL:LOSS:DATA?"
50 ENTER 717;A,B(*)
```

```
10 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:TABL:LOSS:DATA
0" ! Clear Table
```

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.T

()

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.TABLe.LOSS
= *Status*

Status =
SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.COLLect.TABLe.LOSS

/ (*Ch*)(*Pt*)

	Status
	/
	(Boolean)
	• •

```
Dim LComp As Boolean
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.TABLe.LOSS
= True
LComp =
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.TABLe.LOSS
```

Cal > **Power Calibration** > **Loss Compen** > **Compensation**

> > >

SCPI

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection:COLLect
:TABLe:LOSS[:STATe] {ON|OFF|1|0}

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection:COLLect
:TABLe:LOSS[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:PORT1:CORR:COLL:TABL:LOSS
ON"
20 OUTPUT 717;":SOUR1:POW:PORT1:CORR:COLL:TABL:LOSS?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.DATA

()

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.DATA = *Data*

Data = SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.DATA

(*Ch*)(*Pt*)/

	Data
	NOPn1NOP *(n-1)* n 0
	(Variant)
	NOP}2

```
Dim FreqData As Variant
SCPI.SENSe(1).SWEep.POINts = 201
FreqData = SCPI.SENSe(1).FREQuency.DATA
Dim CorData As Variant
SCPI.SENSe(1).SWEep.POINts = 201
CorData = SCPI.SOURce(1).POWer.PORT(1).CORRection.DATA
SCPI.SOURce(1).POWer.PORT(2).CORRection.DATA = CorData
```

SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection.STATe

SCPI.SENSe(Ch).SWEep.POINts

SCPI

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection:DATA <numeric 1>, ..,<numeric NOP>

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection:DATA?

{numeric 1},...,{numeric NOP}<newline><^END>

```
10 DIM A(1:201)
20 OUTPUT 717;":SOUR1:POW:PORT1:CORR:DATA?"
30 ENTER 717;A(*)
```

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.STATe

()

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.STATe = *Status*

Status = SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).CORRection.STATe

(*Ch*)(*Pt*)/

	Status
	/
	(Boolean)
	• •

```
Dim PowCorr As Boolean
SCPI.SOURce(1).POWer.PORT(1).CORRection.STATe = True
PowCorr = SCPI.SOURce(1).POWer.PORT(1).CORRection.STATe
```

Cal > **Power Calibration** > **Correction**

> >

SCPI

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection[:STATe] {ON|OFF|1|0}

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}:CORRection[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:PORT1:CORR ON"
20 OUTPUT 717;":SOUR1:POW:PORT1:CORR?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).LEVel.IMMediate.AMPLi

()

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).LEVel.IMMediate.AMPLitude = *Value*

Value = SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).LEVel.IMMediate.AMPLitude

(*Ch*)(*Pt*)/

	Value
	(Double)
	0
	dBm
	0.05

```
Dim PowLev As Double
SCPI.SOURce(1).POWer.PORT.COUPle = False
SCPI.SOURce(1).POWer.PORT(1).LEVel.IMMediate.AMPLitude =
-12.5
PowLev =
SCPI.SOURce(1).POWer.PORT(1).LEVel.IMMediate.AMPLitude
```

SCPI.SOURce(Ch).POWer.PORT.COUPle

SCPI.SOURce(Ch).POWer.ATTenuation.DATA

Sweep Setup > **Power** > **Port Power** > **Port 1 Power | Port 2 Power | Port 3 Power |Port 4 Power**

> > > **1|2|3|4**

SCPI

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}[:LEVel][:IMMediate][:AMPLitude] <numeric>

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}[:LEVel][:IMMediate][:AMPLitude]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:PORT1 -12.5"
20 OUTPUT 717;":SOUR1:POW:PORT1?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.PORT.COUPle

()

SCPI.SOURce(*Ch*).POWer.PORT.COUPle = *Status*

Status = SCPI.SOURce(*Ch*).POWer.PORT.COUPle

/136 (*Ch*)

	Status
	/
	(Boolean)
	• •

```
Dim OutCpl As Boolean
SCPI.SOURce(1).POWer.PORT.COUPle = False
OutCpl = SCPI.SOURce(1).POWer.PORT.COUPle
```

SCPI.SOURce(Ch).POWer.PORT(Pt).LEVel.IMMediate.AMPLitude

Sweep Setup > **Power** > **Port Couple**

> >

SCPI

:SOURce{[1]-36}:POWer:PORT:COUPle {ON|OFF|1|0}

:SOURce{[1]-36}:POWer:PORT:COUPle?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:PORT:COUP OFF"
20 OUTPUT 717;":SOUR1:POW:PORT:COUP?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.SPAN

()

SCPI.SOURce(*Ch*).POWer.SPAN = *Value*

Value = SCPI.SOURce(*Ch*).POWer.SPAN

136*(Ch)*/

	Value
	(Double)
	(0-65 dBm)
	15
	dBm
	0.05

```
Dim Pspan As Double
SCPI.SOURce(1).POWer.SPAN = 10
Pspan = SCPI.SOURce(1).POWer.SPAN
```

SCPI.SENSe(Ch).SWEep.TYPE
SCPI.SOURce(Ch).POWer.ATTenuation.DATA
SCPI.SOURce(Ch).POWer.CENTer

Span

SCPI

:SOURce{[1]-36}:POWer:SPAN <numeric>

:SOURce{[1]-36}:POWer:SPAN?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:SPAN 10"
20 OUTPUT 717;":SOUR1:POW:SPAN?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.STARt

()

SCPI.SOURce(*Ch*).POWer.STARt = *Value*

Value = SCPI.SOURce(*Ch*).POWer.STARt

136*(Ch)*/

	Value
	(Double)
	-15
	dBm
	0.05

```
Dim Pstart As Double
SCPI.SOURce(1).POWer.STARt = -10
Pstart = SCPI.SOURce(1).POWer.STARt
```

SCPI.SENSe(Ch).SWEep.TYPE

SCPI.SOURce(Ch).POWer.STOP

Start

SCPI

:SOURce{[1]-36}:POWer:STARt <numeric>

:SOURce{[1]-36}:POWer:STARt?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:STAR -10"
20 OUTPUT 717;":SOUR1:POW:STAR?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.STOP

()

SCPI.SOURce(*Ch*).POWer.STOP = *Value*

Value = SCPI.SOURce(*Ch*).POWer.STOP

136*(Ch)*/

	Value
	(Double)
	0
	dBm
	0.05

```
Dim Pstop As Double
SCPI.SOURce(1).POWer.STOP = 10
Pstop = SCPI.SOURce(1).POWer.STOP
```

SCPI.SENSe(Ch).SWEep.TYPE

SCPI.SOURce(Ch).POWer.ATTenuation.DATA

SCPI.SOURce(Ch).POWer.STARt

Stop

SCPI

:SOURce{[1]-36}:POWer:STOP <numeric>

:SOURce{[1]-36}:POWer:STOP?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:STOP 10"
20 OUTPUT 717;":SOUR1:POW:STOP?"
30 ENTER 717;A
```

SCPI.SOURce.POWer.PORT.CORRection.COLLect.ASENsor

()

SCPI.SOURce.POWer.PORT.CORRection.COLLect.ASENsor.RCFactor = *Value*

Value = SCPI.SOURce.POWer.PORT.CORRection.COLLect.ASENsor.RCFactor

/A50 MHz

	Value
	(Double)
	1150
	100
	()
	0.01

```
Dim CalRef As Double
SCPI.SOURce.POWer.PORT.CORRection.COLLect.ASENsor.RCFactor
= 99.5
CalRef =
SCPI.SOURce.POWer.PORT.CORRection.COLLect.ASENsor.RCFactor
```

SCPI.SOURce.POWer.PORT.CORRection.COLLect.BSENsor.RCFactor

Cal > **Power Calibration** > **Sensor A Settings** > **Ref Cal Factor**

> > **A** >

SCPI

:SOURce:POWer:PORT:CORRection:COLLect:ASENsor:RCFactor <numeric>

:SOURce:POWer:PORT:CORRection:COLLect:ASENsor:RCFactor?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:ASEN:RCF 99.5"
20 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:ASEN:RCF?"
30 ENTER 717;A
```

SCPI.SOURce.POWer.PORT.CORRection.COLLect.BSENsor

()

SCPI.SOURce.POWer.PORT.CORRection.COLLect.BSENsor.RCFactor = *Value*

Value = SCPI.SOURce.POWer.PORT.CORRection.COLLect.BSENsor.RCFactor

/B50 MHz

	Value
	(Double)
	1150
	100
	()
	0.01

```
Dim CalRef As Double
SCPI.SOURce.POWer.PORT.CORRection.COLLect.BSENsor.RCFactor
= 99
CalRef =
SCPI.SOURce.POWer.PORT.CORRection.COLLect.BSENsor.RCFactor
```

SCPI.SOURce.POWer.PORT.CORRection.COLLect.ASENsor.RCFactor

Cal > **Power Calibration** > **Sensor B Settings** > **Ref Cal Factor**

> > **B** >

SCPI

:SOURce:POWer:PORT:CORRection:COLLect:BSENsor:RCFactor <numeric>

:SOURce:POWer:PORT:CORRection:COLLect:BSENsor:RCFactor?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:BSEN:RCF 99.5"
20 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:BSEN:RCF?"
30 ENTER 717;A
```

SCPI.SOURce.POWer.PORT.CORRection.COLLect.NTOLera

()

SCPI.SOURce.POWer.PORT.CORRection.COLLect.NTOLerance= *Value*

Value =
SCPI.SOURce.POWer.PORT.CORRection.COLLect.NTOLerance

(*Ch*)(*Pt*)/

	Value
	(Double)
	0100
	5
	dB
	0.001

```
Dim TorVal As Long
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.NTOLerance
= 10
TorVal =
SCPI.SOURce(1).POWer.PORT(1).CORRection.COLLect.NTOLerance
```

SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection.COLLect.ACQuire

Cal > **Power Calibration** > **Tolerance**

> >

SCPI

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}
:CORRection:COLLect:NTOLerance <numeric>

:SOURce{[1]-36}:POWer:PORT{[1]|2|3|4}
:CORRection:COLLect:NTOLerance?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SOUR1:POW:PORT1:CORR:COLL:NTOL 6"
20 OUTPUT 717;":SOUR1:POW:PORT1:CORR:COLL:NTOL?"
30 ENTER 717;A
```

SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.AS

()

SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.ASENsor.DAT
= *Data*

Data =
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.ASENsor.DAT

/A

	Data
	1+NumX2n1Num • *(0)* 01000 *(0)* • *(nX2-1)* n(1 kHz500 GHz) • *(nX2)* n 1150 • 0
	(Variant)
	1+Num ()/2 (n\2-1)(n\2)

```
Dim CalFac As Variant
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.ASENsor.DA
= Array(2,1e7,99.8, 1e9,98.7)
CalFac =
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.ASENsor.DA
'''Clear Cal Factor Table
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.ASENsor.DA
= Array(0)
Dim CalFac(4) As Variant
Dim Ref As Variant
CalFac(0) = 2
CalFac(1) = 1e7
CalFac(2) = 99.8
CalFac(3) = 1e9
CalFac(4) = 98.7
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.ASENsor.DA
= CalFac
Ref =
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.ASENsor.DA
'''Clear Cal Factor Table
Dim CalFac(0) As Variant
CalFac(0) = 0
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.ASENsor.DA
= CalFac
```

SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BSENsor.DAT

Cal > Power Calibration > Sensor A Settings > Delete | Add | Clear Cal Factor Table

> > **A** > ||

SCPI

:SOURce:POWer:PORT:CORRection:COLLect:TABLe:ASENsor:DATA <numeric 1>,...,<numeric 1+(Nx2)>

:SOURce:POWer:PORT:CORRection:COLLect:TABLe:ASENsor:DATA?

{numeric 1},...,{numeric 1+(Nx2)}<newline><^END>

```
10 DIM B(1:3,1:2)
20 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:TABL:ASEN:DATA
3,";
30 OUTPUT 717;"1E6,98.5,1E7,99,1E8,99.5"
40 OUTPUT
717;":SOUR:POW:PORT:CORR:COLL:TABL:ASEN:DATA?"
50 ENTER 717;A,B(*)
```

```
10 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:TABL:ASEN:DATA
0" !Clear Table
```

SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BS

()

SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BSENsor.DAT
= *Data*

Data =
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BSENsor.DAT

/B

	Data
	1+NumX2n1Num • *(0)* 01000 *(0)* • *(nX2-1)* n(1 kHz500 GHz) • *(n*X*2)* n1150 0
	(Variant)
	1+Num ()X2 (nX2-1)(n X2)

```
Dim CalFac As Variant
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BSENsor.DA
= Array(2,1e7,99.8, 1e9,98.7)
CalFac =
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BSENsor.DA
'''Clear Cal Factor Table
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BSENsor.DA
= Array(0)
```

```
Dim CalFac(4) As Variant
Dim Ref As Variant
CalFac(0) = 2
CalFac(1) = 1e7
CalFac(2) = 99.8
CalFac(3) = 1e9
CalFac(4) = 98.7
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BSENsor.DA
= CalFac
Ref =
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BSENsor.DA
'''Clear Cal Factor Table
Dim CalFac(0) As Variant
CalFac(0) = 0
SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.BSENsor.DA
= CalFac
```

SCPI.SOURce.POWer.PORT.CORRection.COLLect.TABLe.ASENsor.DAT

Cal > Power Calibration > Sensor B Settings > Delete | Add | Clear Cal Factor Table

> > **B** > ||

SCPI

:SOURce:POWer:PORT:CORRection:COLLect:TABLe:BSENsor:DATA <numeric 1>,...,<numeric 1+(Nx2)>

:SOURce:POWer:PORT:CORRection:COLLect:TABLe:BSENsor:DATA? <numeric 1> parameter.

{numeric 1},...,{numeric 1+(Nx2)}<newline><^END>

```
10 DIM B(1:3,1:2)
20 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:TABL:BSEN:DATA
3,";
30 OUTPUT 717;"1E6,98.5,1E7,99,1E8,99.5"
40 OUTPUT
717;":SOUR:POW:PORT:CORR:COLL:TABL:BSEN:DATA?"
50 ENTER 717;A,B(*)
```

```
10 OUTPUT 717;":SOUR:POW:PORT:CORR:COLL:TABL:BSEN:DATA
0" !Clear Table
```

SCPI.STATus.OPERation.CONDition

()

Value = SCPI.STATus.OPERation.CONDition

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.OPERation.CONDition
```

SCPI.STATus.OPERation.NTRansition

SCPI.STATus.OPERation.PTRansition

SCPI

:STATus:OPERation:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:OPER:COND?"
20 ENTER 717;A
```

SCPI.STATus.OPERation.ENABle

()

SCPI.STATus.OPERation.ENABle = *Value*

Value = SCPI.STATus.OPERation.ENABle

/

	Value
	(Long)
	065535
	0
	03613151

```
Dim Stat As Long
SCPI.STATus.OPERation.ENABle = 16
Stat = SCPI.STATus.OPERation.ENABle
```

SCPI.IEEE4882.SRE

SCPI

:STATus:OPERation:ENABle <numeric>

:STATus:OPERation:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:OPER:ENAB 16"
20 OUTPUT 717;":STAT:OPER:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.OPERation.EVENt

()

Value = SCPI.STATus.OPERation.EVENt

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.OPERation.EVENt
```

SCPI.IEEE4882.CLS

SCPI.STATus.OPERation.NTRansition

SCPI.STATus.OPERation.PTRansition

SCPI

:STATus:OPERation[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:OPER?"
20 ENTER 717;A
```

SCPI.STATus.OPERation.NTRansition

()

SCPI.STATus.OPERation.NTRansition = *Value*

Value = SCPI.STATus.OPERation.NTRansition

/

	Value
	(Long)
	065535
	0
	03613151

```
Dim Stat As Long
SCPI.STATus.OPERation.NTRansition = 16
Stat = SCPI.STATus.OPERation.NTRansition
```

SCPI.STATus.OPERation.EVENt

SCPI.STATus.OPERation.PTRansition

SCPI

:STATus:OPERation:NTRansition <numeric>

:STATus:OPERation:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:OPER:NTR 16"
20 OUTPUT 717;":STAT:OPER:NTR?"
30 ENTER 717;A
```

SCPI.STATus.OPERation.PTRansition

()

SCPI.STATus.OPERation.PTRansition = *Value*

Value = SCPI.STATus.OPERation.PTRansition

/

	Value
	(Long)
	065535
	16432
	03613151

```
Dim Stat As Long
SCPI.STATus.OPERation.PTRansition = 0
Stat = SCPI.STATus.OPERation.PTRansition
```

SCPI.STATus.OPERation.EVENt

SCPI.STATus.OPERation.NTRansition

SCPI

:STATus:OPERation:PTRansition <numeric>

:STATus:OPERation:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:OPER:PTR 16"
20 OUTPUT 717;":STAT:OPER:PTR?"
30 ENTER 717;A
```

SCPI.STATus.PRESet

SCPI.STATus.PRESet

{1-36}{1-36}

SCPI.STATus.PRESet

SCPI

:STATus:PRESet

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).CONDition

()

Value = SCPI.STATUS.QUEStionable.BLIMit.CHANnel(Ch).CONDition

	Value
	(Long)

(*Ch*) Ch

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).CONDition
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).NTRansition
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).PTRansition

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.

()

Value =
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.CONDition

“”

	Value
	(Long)

(*Ch*) Ch

```
Dim Stat As Long
Stat =
SCPI.STATus.QUStionable.BLIMit.CHANnel(1).ECHannel.CONDition
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.NTRansition
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.PTRansition

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ECHannel:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ECH:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.

()

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.ENABle = Value

Value = SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.ENABle

/

	Value
	(Long)
	065535
	/
	03151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ECHannel.ENABle = 6
Stat =
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ECHannel.ENABle
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ENABle

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ECHannel:ENABle <numeric>

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ECHannel:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ECH:ENAB 6"
20 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ECH:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.

()

Value =
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.EVENt

“”

	Value
	(Long)

(*Ch*) Ch

```
Dim Stat As Long
Stat =
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ECHannel.EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ECHannel[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ECH?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.

()

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.NTRansitior
= Value

Value =
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.NTRansitior

/“”

	Value
	(Long)
	065535
	0
	03151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ECHannel.NTRansitior
= 6
Stat =
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ECHannel.NTRansitior
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.EVENt
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.PTRansitior

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ECHannel:NTRansition <numeric>

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ECHannel:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ECH:NTR 6"
20 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ECH:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.

()

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.PTRansitior
= Value

Value =
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.PTRansitior

/“”

	Value
	(Long)
	065535
	/
	03151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ECHannel.PTRansition
= 6
Stat =
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ECHannel.PTRansition
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.EVENt
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ECHannel.NTRansitior

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ECHannel:PTRansition <numeric>

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ECHannel:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ECH:PTR 6"
20 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ECH:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ENABle

()

```
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ENABle = Value
Value = SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).ENABle
```

/“”

	Value
	(Long)
	065535
	/
	151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ENABle = 16
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ENABle
```

SCPI.STATus.QUEStionable.BLIMit.ENABle

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ENABle <numeric>

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:ENABle?

{numeric}<newline><^END>

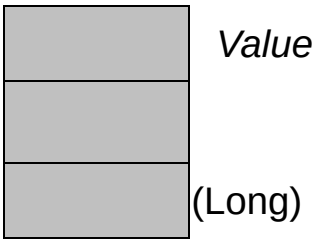
```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ENAB 16"
20 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).EVENt

()

Value = SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).EVENt

“”



(*Ch*) Ch

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).NTRansitio

()

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).NTRansition = Value
Value = SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).NTRansition

/“”

	Value
	(Long)
	065535
	0
	151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).NTRansition = 16
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).NTRansition
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).PTRansition

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:NTRansition <numeric>

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:NTR 16"
20 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).PTRansitic

()

```
SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).PTRansition = Value
Value = SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).PTRansition
```

/“”

	Value
	(Long)
	065535
	/
	151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).PTRansition = 0
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).PTRansition
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).NTRansition

SCPI

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:PTRansition <numeric>

:STATus:QUEStionable:BLIMit:CHANnel{[1]-16}:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:PTR 16"
20 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CONDition

()

Value = SCPI.STATus.QUEStionable.BLIMit.CONDition

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.CONDition
```

SCPI.STATus.QUEStionable.BLIMit.NTRansition
SCPI.STATus.QUEStionable.BLIMit.PTRansition

SCPI

:STATus:QUEStionable:BLIMit:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.ELIMit.CONDition

()

Value = SCPI.STATus.QUEStionable.BLIMit.ELIMit.CONDition

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.ELIMit.CONDition
```

SCPI.STATus.QUEStionable.BLIMit.ELIMit.NTRansition
SCPI.STATus.QUEStionable.BLIMit.ELIMit.PTRansition

SCPI

:STATus:QUEStionable:BLIMit:ELIMit:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:ELIM:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.ELIMit.ENABle

()

SCPI.STATus.QUEStionable.BLIMit.ELIMit.ENABle = Value

Value = SCPI.STATus.QUEStionable.BLIMit.ELIMit.ENABle

/

	Value
	(Long)
	065535
	/
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.ELIMit.ENABle = 6
Stat = SCPI.STATus.QUEStionable.BLIMit.ELIMit.ENABle
```

SCPI.STATus.QUEStionable.BLIMit.ENABle

SCPI

:STATus:QUEStionable:BLIMit:ELIMit:ENABle <numeric>

:STATus:QUEStionable:BLIMit:ELIMit:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:ELIM:ENAB 6"
20 OUTPUT 717;":STAT:QUES:BLIM:ELIM:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.ELIMit.EVENt

()

Value = SCPI.STATus.QUEStionable.BLIMit.ELIMit.EVENt

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.ELIMit.EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:BLIMit:ELIMit[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:ELIM?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.ELIMit.NTRansition

()

SCPI.STATus.QUEStionable.BLIMit.ELIMit.NTRansition = Value

Value = SCPI.STATus.QUEStionable.BLIMit.ELIMit.NTRansition

/

	Value
	(Long)
	065535
	0
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.ELIMit.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.BLIMit.ELIMit.NTRansition
```

SCPI.STATus.QUEStionable.BLIMit.ELIMit.EVENt

SCPI.STATus.QUEStionable.BLIMit.ELIMit.PTRansition

SCPI

:STATus:QUEStionable:BLIMit:ELIMit:NTRansition <numeric>

:STATus:QUEStionable:BLIMit:ELIMit:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:ELIM:NTR 6"
20 OUTPUT 717;":STAT:QUES:BLIM:ELIM:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.ELIMit.PTRansition

()

SCPI.STATus.QUEStionable.BLIMit.ELIMit.PTRansition = Value
Value = SCPI.STATus.QUEStionable.BLIMit.ELIMit.PTRansition

/

	Value
	(Long)
	065535
	/
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.ELIMit.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.BLIMit.ELIMit.PTRansition
```

SCPI.STATus.QUEStionable.BLIMit.ELIMit.EVENt

SCPI.STATus.QUEStionable.BLIMit.ELIMit.NTRansition

SCPI

:STATus:QUEStionable:BLIMit:ELIMit:PTRansition <numeric>

:STATus:QUEStionable:BLIMit:ELIMit:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:ELIM:PTR 6"
20 OUTPUT 717;":STAT:QUES:BLIM:ELIM:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.ENABle

()

SCPI.STATus.QUEStionable.BLIMit.ENABle = Value

Value = SCPI.STATus.QUEStionable.BLIMit.ENABle

/

	Value
	(Long)
	065535
	/
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.ENABle = 16
Stat = SCPI.STATus.QUEStionable.BLIMit.ENABle
```

SCPI.STATus.QUEStionable.ENABle

SCPI

:STATus:QUEStionable:BLIMit:ENABle <numeric>

:STATus:QUEStionable:BLIMit:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:ENAB 16"
20 OUTPUT 717;":STAT:QUES:BLIM:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.EVENt

()

Value = SCPI.STATus.QUEStionable.BLIMit.EVENt

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:BLIMit[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.NTRansition

()

SCPI.STATus.QUEStionable.BLIMit.NTRansition = Value

Value = SCPI.STATus.QUEStionable.BLIMit.NTRansition

/

	Value
	(Long)
	065535
	0
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.BLIMit.NTRansition
```

SCPI.STATus.QUEStionable.BLIMit.EVENt

SCPI.STATus.QUEStionable.BLIMit.PTRansition

SCPI

:STATus:QUEStionable:BLIMit:NTRansition <numeric>

:STATus:QUEStionable:BLIMit:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:NTR 16"
20 OUTPUT 717;":STAT:QUES:BLIM:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.PTRansition

()

SCPI.STATus.QUEStionable.BLIMit.PTRansition = Value

Value = SCPI.STATus.QUEStionable.BLIMit.PTRansition

/

	Value
	(Long)
	065535
	/
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.BLIMit.PTRansition
```

SCPI.STATus.QUEStionable.BLIMit.EVENt

SCPI.STATus.QUEStionable.BLIMit.NTRansition

SCPI

:STATus:QUEStionable:BLIMit:PTRansition <numeric>

:STATus:QUEStionable:BLIMit:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:BLIM:PTR 16"
20 OUTPUT 717;":STAT:QUES:BLIM:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.CONDition

()

Value = SCPI.STATus.QUEStionable.CONDition

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.CONDition
```

SCPI.STATus.QUEStionable.NTRansition
SCPI.STATus.QUEStionable.PTRansition

SCPI

:STATus:QUEStionable:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.ENABle

()

SCPI.STATus.QUEStionable.ENABle = *Value*

Value = SCPI.STATus.QUEStionable.ENABle

/

	Value
	(Long)
	065535
	0
	0911151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.ENABle = 6
Stat = SCPI.STATus.QUEStionable.ENABle
```

SCPI.IEEE4882.SRE

SCPI

:STATus:QUEStionable:ENABle <numeric>

:STATus:QUEStionable:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:ENAB 16"
20 OUTPUT 717;":STAT:QUES:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.EVENt

()

Value = SCPI.STATus.QUEStionable.EVENt

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.EVENt
```

SCPI.IEEE4882.CLS
SCPI.STATus.QUEStionable.NTRansition
SCPI.STATus.QUEStionable.PTRansition

SCPI

:STATus:QUEStionable[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).CONDition

()

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).CONDition

(*Ch*)

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).CONDition
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).NTRansition
SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).PTRansition

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:CONDition?

{numeric}<newline><^END>

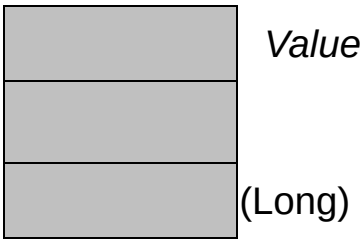
```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.C

()

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).CONDition

(*Ch*)


Value
(Long)

```
Dim Stat As Long
Stat =
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ECHannel.CONDition
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).ECHannel.NTRansition
SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).ECHannel.PTRansition

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ECHannel:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ECH:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.E

()

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.ENABle = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.ENABle

/ (*Ch*)

	Value
	(Long)
	065535
	/
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ECHannel.ENABle = 6
Stat =
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ECHannel.ENABle
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).ENABle

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ECHannel:ENABle <numeric>

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ECHannel:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ECH:ENAB 6"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ECH:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.E

()

Value =
SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.EVENt

(*Ch*)

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ECHannel.EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ECHannel[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ECH?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.N

()

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.NTRansition

/ (*Ch*)

	Value
	(Long)
	065535
	0
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ECHannel.NTRansition
= 6
Stat =
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ECHannel.NTRansition
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).ECHannel.EVENt
SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).ECHannel.PTRansition

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ECHannel:NTRansition <numeric>

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ECHannel:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ECH:NTR 6"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ECH:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.P

()

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ECHannel.PTRansition

/ (*Ch*)

	Value
	(Long)
	065535
	/
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ECHannel.PTRansition
= 6
Stat =
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ECHannel.PTRansition
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).ECHannel.EVENt
SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).ECHannel.NTRansition

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ECHannel:PTRansition <numeric>

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ECHannel:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ECH:PTR 6"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ECH:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ENABle

()

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ENABle = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ENABle

/ (*Ch*)

	Value
	(Long)
	065535
	/
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ENABle = 16
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ENABle
```

SCPI.STATus.QUEStionable.LIMit.ENABle

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ENABle <numeric>

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ENAB 16"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).EVENt

()

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).EVENt

(*Ch*)

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).NTRansition

()

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).NTRansition

/ (*Ch*)

	Value
	(Long)
	065535
	0
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).NTRansition = 16
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).NTRansition
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).PTRansition

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:NTRansition <numeric>

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:NTR 16"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).PTRansition

()

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).PTRansition

/ (*Ch*)

	Value
	(Long)
	065535
	/
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).PTRansition = 0
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).PTRansition
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).NTRansition

SCPI

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:PTRansition <numeric>

:STATus:QUEStionable:LIMit:CHANnel{[1]-16}:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:PTR 16"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CONDition

()

Value = SCPI.STATus.QUEStionable.LIMit.CONDition

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.CONDition
```

SCPI.STATus.QUEStionable.LIMit.NTRansition

SCPI.STATus.QUEStionable.LIMit.PTRansition

SCPI

:STATus:QUEStionable:LIMit:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.ELIMit.CONDition

()

Value = SCPI.STATus.QUEStionable.LIMit.ELIMit.CONDition

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.ELIMit.CONDition
```

SCPI.STATus.QUEStionable.LIMit.ELIMit.NTRansition
SCPI.STATus.QUEStionable.LIMit.ELIMit.PTRansition

SCPI

:STATus:QUEStionable:LIMit:ELIMit:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:ELIM:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.ELIMit.ENABle

()

SCPI.STATus.QUEStionable.LIMit.ELIMit.ENABle = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.ELIMit.ENABle

/

	Value
	(Long)
	065535
	/
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.ELIMit.ENABle = 6
Stat = SCPI.STATus.QUEStionable.LIMit.ELIMit.ENABle
```

SCPI.STATus.QUEStionable.LIMit.ENABle

SCPI

:STATus:QUEStionable:LIMit:ELIMit:ENABle <numeric>

:STATus:QUEStionable:LIMit:ELIMit:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:ELIM:ENAB 6"
20 OUTPUT 717;":STAT:QUES:LIM:ELIM:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.ELIMit.EVENt

()

Value = SCPI.STATus.QUEStionable.LIMit.ELIMit.EVENt

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.ELIMit.EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:LIMit:ELIMit[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:ELIM?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.ELIMit.NTRansition

()

SCPI.STATus.QUEStionable.LIMit.ELIMit.NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.ELIMit.NTRansition

/

	Value
	(Long)
	065535
	0
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.ELIMit.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.LIMit.ELIMit.NTRansition
```

SCPI.STATus.QUEStionable.LIMit.ELIMit.EVENt

SCPI.STATus.QUEStionable.LIMit.ELIMit.PTRansition

SCPI

:STATus:QUEStionable:LIMit:ELIMit:NTRansition <numeric>

:STATus:QUEStionable:LIMit:ELIMit:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:ELIM:NTR 6"
20 OUTPUT 717;":STAT:QUES:LIM:ELIM:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.ELIMit.PTRansition

()

SCPI.STATus.QUEStionable.LIMit.ELIMit.PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.ELIMit.PTRansition

/

	Value
	(Long)
	065535
	/
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.ELIMit.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.LIMit.ELIMit.PTRansition
```

SCPI.STATus.QUEStionable.LIMit.ELIMit.EVENt

SCPI.STATus.QUEStionable.LIMit.ELIMit.NTRansition

SCPI

:STATus:QUEStionable:LIMit:ELIMit:PTRansition <numeric>

:STATus:QUEStionable:LIMit:ELIMit:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:ELIM:PTR 6"
20 OUTPUT 717;":STAT:QUES:LIM:ELIM:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.ENABle

()

SCPI.STATus.QUEStionable.LIMit.ENABle = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.ENABle

/

	Value
	(Long)
	065535
	/
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.ENABle = 16
Stat = SCPI.STATus.QUEStionable.LIMit.ENABle
```

SCPI.STATus.QUEStionable.ENABle

SCPI

:STATus:QUEStionable:LIMit:ENABle <numeric>

:STATus:QUEStionable:LIMit:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:ENAB 16"
20 OUTPUT 717;":STAT:QUES:LIM:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.EVENt

()

Value = SCPI.STATus.QUEStionable.LIMit.EVENt

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:LIMit[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.NTRansition

()

SCPI.STATus.QUEStionable.LIMit.NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.NTRansition

/

	Value
	(Long)
	065535
	0
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.LIMit.NTRansition
```

SCPI.STATus.QUEStionable.LIMit.EVENt

SCPI.STATus.QUEStionable.LIMit.PTRansition

SCPI

:STATus:QUEStionable:LIMit:NTRansition <numeric>

:STATus:QUEStionable:LIMit:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:NTR 16"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.PTRansition

()

SCPI.STATus.QUEStionable.LIMit.PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.PTRansition

/

	Value
	(Long)
	065535
	/
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.LIMit.PTRansition
```

SCPI.STATus.QUEStionable.LIMit.EVENt

SCPI.STATus.QUEStionable.LIMit.NTRansition

SCPI

:STATus:QUEStionable:LIMit:PTRansition <numeric>

:STATus:QUEStionable:LIMit:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:PTR 16"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.NTRansition

()

SCPI.STATus.QUEStionable.NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.NTRansition

/

	Value
	(Long)
	065535
	0
	0911 151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.NTRansition
```

SCPI.STATus.QUEStionable.EVENt

SCPI.STATus.QUEStionable.PTRansition

SCPI

:STATus:QUEStionable:NTRansition <numeric>

:STATus:QUEStionable:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:NTR 16"
20 OUTPUT 717;":STAT:QUES:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.PTRansition

()

SCPI.STATus.QUEStionable.PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.PTRansition

/

	Value
	(Long)
	065535
	1024
	0911151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.PTRansition
```

SCPI.STATus.QUEStionable.EVENt

SCPI.STATus.QUEStionable.NTRansition

SCPI

:STATus:QUEStionable:PTRansition <numeric>

:STATus:QUEStionable:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:PTR 16"
20 OUTPUT 717;":STAT:QUES:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).CONDition

()

Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).CONDition

	Value
	(Long)

(*Ch*) Ch

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).CONDition
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).NTRansition
SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).PTRansition

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.

()

Value =
SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.CONDition

	Value
	(Long)

(*Ch*) Ch

```
Dim Stat As Long
Stat =
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ECHannel.CONDition
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.NTRansitior
SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.PTRansitior

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ECHannel:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ECH:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.

()

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.ENABle = Value

Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.ENABle

/

	Value
	(Long)
	065535
	/
	03151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ECHannel.ENABle = 6
Stat =
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ECHannel.ENABle
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ENABle

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ECHannel:ENABle <numeric>

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ECHannel:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ECH:ENAB 6"
20 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ECH:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.

()

Value =
SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.EVENt

	Value
	(Long)

(*Ch*) Ch

```
Dim Stat As Long
Stat =
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ECHannel.EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ECHannel[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ECH?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.

()

SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).ECHannel.NTRansitior
= *Value*

Value =
SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).ECHannel.NTRansitior

/

	Value
	(Long)
	065535
	0
	0151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ECHannel.NTRansitior
= 6
Stat =
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ECHannel.NTRansitior
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.EVENt
SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.PTRansitior

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ECHannel:NTRansition <numeric>

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ECHannel:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ECH:NTR 6"
20 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ECH:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.

()

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.PTRansitior
= Value

Value =
SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.PTRansitior

/

	Value
	(Long)
	065535
	/
Note	03151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ECHannel.PTRansitior
= 6
Stat =
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ECHannel.PTRansitior
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.EVENt
SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ECHannel.NTRansitior

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ECHannel:PTRansition <numeric>

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ECHannel:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ECH:PTR 6"
20 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ECH:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ENABle

()

```
SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ENABle = Value
Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).ENABle
```

/

	Value
	(Long)
	065535
	/
	1

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ENABle = 16
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ENABle
```

SCPI.STATus.QUEStionable.RLIMit.ENABle

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ENABle <numeric>

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ENAB 16"
20 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).EVENt

()

Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).EVENt

	Value
	(Long)

(*Ch*) Ch

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).NTRansitio

()

```
SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).NTRansition = Value
Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).NTRansition
```

/

	Value
	(Long)
	065535
	0
	151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).NTRansition = 16
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).NTRansition
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).PTRansition

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:NTRansition <numeric>

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:NTR 16"
20 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).PTRansitic

()

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).PTRansition = Value
Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).PTRansition

/

	Value
	(Long)
	065535
	/
	151

(*Ch*) Ch

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).PTRansition = 0
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).PTRansition
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).NTRansition

SCPI

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:PTRansition <numeric>

:STATus:QUEStionable:RLIMit:CHANnel{[1]-16}:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:PTR 16"
20 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CONDition

()

Value = SCPI.STATus.QUEStionable.RLIMit.CONDition

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.RLIMit.CONDition
```

SCPI.STATus.QUEStionable.RLIMit.NTRansition
SCPI.STATus.QUEStionable.RLIMit.PTRansition

SCPI

:STATus:QUEStionable:RLIMit:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.ELIMit.CONDition

()

Value = SCPI.STATus.QUEStionable.RLIMit.ELIMit.CONDition

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.RLIMit.ELIMit.CONDition
```

SCPI.STATus.QUEStionable.RLIMit.ELIMit.NTRansition
SCPI.STATus.QUEStionable.RLIMit.ELIMit.PTRansition

SCPI

:STATus:QUEStionable:RLIMit:ELIMit:CONDition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:ELIM:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.ELIMit.ENABle

()

SCPI.STATus.QUEStionable.RLIMit.ELIMit.ENABle = Value

Value = SCPI.STATus.QUEStionable.RLIMit.ELIMit.ENABle

/

	Value
	(Long)
	065535
	/
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.ELIMit.ENABle = 6
Stat = SCPI.STATus.QUEStionable.RLIMit.ELIMit.ENABle
```

SCPI.STATus.QUEStionable.RLIMit.ENABle

SCPI

:STATus:QUEStionable:RLIMit:ELIMit:ENABle <numeric>

:STATus:QUEStionable:RLIMit:ELIMit:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:ELIM:ENAB 6"
20 OUTPUT 717;":STAT:QUES:RLIM:ELIM:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.ELIMit.EVENt

()

Value = SCPI.STATus.QUEStionable.RLIMit.ELIMit.EVENt

	Value
	(Long)

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.RLIMit.ELIMit.EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:RLIMit:ELIMit[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:ELIM?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.ELIMit.NTRansition

()

SCPI.STATus.QUEStionable.RLIMit.ELIMit.NTRansition = Value

Value = SCPI.STATus.QUEStionable.RLIMit.ELIMit.NTRansition

/

	Value
	(Long)
	065535
	0
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.ELIMit.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.RLIMit.ELIMit.NTRansition
```

SCPI.STATus.QUEStionable.RLIMit.ELIMit.EVENt

SCPI.STATus.QUEStionable.RLIMit.ELIMit.PTRansition

SCPI

:STATus:QUEStionable:RLIMit:ELIMit:NTRansition <numeric>

:STATus:QUEStionable:RLIMit:ELIMit:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:ELIM:NTR 6"
20 OUTPUT 717;":STAT:QUES:RLIM:ELIM:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.ELIMit.PTRansition

()

SCPI.STATus.QUEStionable.RLIMit.ELIMit.PTRansition = Value

Value = SCPI.STATus.QUEStionable.RLIMit.ELIMit.PTRansition

/

	Value
	(Long)
	065535
	/
	03151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.ELIMit.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.RLIMit.ELIMit.PTRansition
```

SCPI.STATus.QUEStionable.RLIMit.ELIMit.EVENt

SCPI.STATus.QUEStionable.RLIMit.ELIMit.NTRansition

SCPI

:STATus:QUEStionable:RLIMit:ELIMit:PTRansition <numeric>

:STATus:QUEStionable:RLIMit:ELIMit:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:ELIM:PTR 6"
20 OUTPUT 717;":STAT:QUES:RLIM:ELIM:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.ENABle

()

SCPI.STATus.QUEStionable.RLIMit.ENABle = Value

Value = SCPI.STATus.QUEStionable.RLIMit.ENABle

/

	Value
	(Long)
	065535
	/
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.ENABle = 16
Stat = SCPI.STATus.QUEStionable.RLIMit.ENABle
```

SCPI.STATus.QUEStionable.ENABle

SCPI

:STATus:QUEStionable:RLIMit:ENABle <numeric>

:STATus:QUEStionable:RLIMit:ENABle?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:ENAB 16"
20 OUTPUT 717;":STAT:QUES:RLIM:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.EVENt

()

Value = SCPI.STATus.QUEStionable.RLIMit.EVENt

	Value
	(Long)

```
Dim Stat As Long
Stat =SCPI.STATus.QUEStionable.RLIMit.EVENt
```

SCPI.IEEE4882.CLS

SCPI

:STATus:QUEStionable:RLIMit[:EVENt]?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.NTRansition

()

SCPI.STATus.QUEStionable.RLIMit.NTRansition = Value

Value = SCPI.STATus.QUEStionable.RLIMit.NTRansition

/

	Value
	(Long)
	065535
	0
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.RLIMit.NTRansition
```

SCPI.STATus.QUEStionable.RLIMit.EVENt

SCPI.STATus.QUEStionable.RLIMit.PTRansition

SCPI

:STATus:QUEStionable:RLIMit:NTRansition <numeric>

:STATus:QUEStionable:RLIMit:NTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:NTR 16"
20 OUTPUT 717;":STAT:QUES:RLIM:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.PTRansition

()

SCPI.STATus.QUEStionable.RLIMit.PTRansition = Value

Value = SCPI.STATus.QUEStionable.RLIMit.PTRansition

/

	Value
	(Long)
	0 to 65535
	/
	151

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.RLIMit.PTRansition
```

SCPI.STATus.QUEStionable.RLIMit.EVENt

SCPI.STATus.QUEStionable.RLIMit.NTRansition

SCPI

:STATus:QUEStionable:RLIMit:PTRansition <numeric>

:STATus:QUEStionable:RLIMit:PTRansition?

{numeric}<newline><^END>

```
10 OUTPUT 717;":STAT:QUES:RLIM:PTR 16"
20 OUTPUT 717;":STAT:QUES:RLIM:PTR?"
30 ENTER 717;A
```

SCPI.SYSTem.BACKlight

SCPI.SYSTem.BACKlight = *Status*

Status = SCPI.SYSTem.BACKlight

/LCD

	Status
	/
	(Boolean)
	• •

```
Dim BckLght As Boolean
SCPI.SYSTem.BACKlight = False
BckLght = SCPI.SYSTem.BACKlight
```

System > **Backlight**

>

SCPI

:SYSTem:BACKlight {ON|OFF|1|0}

:SYSTem:BACKlight?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SYST:BACK OFF"
20 OUTPUT 717;":SYST:BACK?"
30 ENTER 717;A
```

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

SCPI.SYSTem.BEEPer.COMPlete.STATe

SCPI.SYSTem.BEEPer.WARNing.IMMediate

System > **Misc Setup** > **Beeper** > **Test Beep Complete**

> **Misc** > >

SCPI

:SYSTem:BEEPer:COMPlete:IMMediate

```
10 OUTPUT 717;":SYST:BEEP:COMP:IMM"
```

SCPI.SYSTem.BEEPer.COMPlete.STATe

SCPI.SYSTem.BEEPer.COMPlete.STATe = *Status*

Status = SCPI.SYSTem.BEEPer.COMPlete.STATe

/

Status

/

(Boolean)

-
-

```
Dim BeepComp As Boolean
SCPI.SYSTem.BEEPer.COMPlete.STATe = False
BeepComp = SCPI.SYSTem.BEEPer.COMPlete.STATe
```

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

SCPI.SYSTem.BEEPer.WARNing.STATe

System > **Misc Setup** > **Beeper** > **Beep Complete**
> **Misc** > >

SCPI

:SYSTem:BEEPer:COMPlete:STATe {ON|OFF|1|0}

:SYSTem:BEEPer:COMPlete:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SYST:BEEP:COMP:STAT OFF"
20 OUTPUT 717;":SYST:BEEP:COMP:STAT?"
30 ENTER 717;A
```

SCPI.SYSTem.BEEPer.WARNing.IMMediate

SCPI.SYSTem.BEEPer.WARNing.IMMediate

/

SCPI.SYSTem.BEEPer.WARNing.IMMediate

SCPI.SYSTem.BEEPer.WARNing.STATe

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

System > **Misc Setup** > **Beeper** > **Test Beep Warning**

> **Misc** > >

SCPI

:SYSTem:BEEPer:WARNing:IMMediate

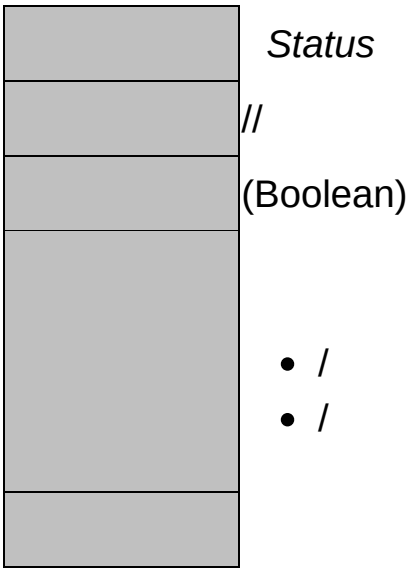
```
10 OUTPUT 717;":SYST:BEEP:WARN:IMM"
```

SCPI.SYSTem.BEEPer.WARNing.STATe

SCPI.SYSTem.BEEPer.WARNing.STATe = *Status*

Status = SCPI.SYSTem.BEEPer.WARNing.STATe

//


Status
//
(Boolean)
/
/

```
Dim BeepWarn As Boolean
SCPI.SYSTem.BEEPer.WARNing.STATe = False
BeepWarn = SCPI.SYSTem.BEEPer.WARNing.STATe
```

SCPI.SYSTem.BEEPer.WARNing.IMMediate

SCPI.SYSTem.BEEPer.COMPlete.STATe

System > **Misc Setup** > **Beeper** > **Beep Warning**
> **Misc** > >

SCPI

:SYSTem:BEEPer:WARNing:STATe {ON|OFF|1|0}

:SYSTem:BEEPer:WARNing:STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SYST:BEEP:WARN:STAT OFF"
20 OUTPUT 717;":SYST:BEEP:WARN:STAT?"
30 ENTER 717;A
```

SCPI.SYSTem.COMMunicate.GPIB.PMETer.ADDRess

SCPI.SYSTem.COMMunicate.GPIB.PMETer.ADDRess = *Value*

Value = SCPI.COMMunicate.GPIB.PMETer.ADDRess

/GPIB

	Value
	GPIB
	(Long)
	030
	13

```
Dim Paddr As Long
SCPI.SYSTem.COMMunicate.GPIB.PMETer.ADDRess = 15
Paddr = SCPI.SYSTem.COMMunicate.GPIB.PMETer.ADDRess
```

System > **Misc Setup** > **GPIB Setup** > **Power Meter Address**
> **Misc** > **GPIB** >

SCPI

:SYSTem:COMMunicate:GPIB:PMETer:ADDRess <numeric>

:SYSTem:COMMunicate:GPIB:PMETer:ADDRess?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SYST:COMM:GPIB:PMET:ADDR 10"
20 OUTPUT 717;":SYST:COMM:GPIB:PMET:ADDR?"
30 ENTER 717;A
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.ADDRess

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.ADDRess = *Value*

Value = SCPI.SYSTem.COMMunicate.GPIB.SGENerator.ADDRess

/GPIB

	Value
	GPIB
	(Long)
	030
	19

```
Dim SGaddr As Long
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.ADDRess = 20
SGaddr = SCPI.SYSTem.COMMunicate.GPIB.SGENerator.ADDRess
```

System > **Misc Setup** > **GPIB Setup** > **Signal Generator Address** > **Misc** > **GPIB** >

SCPI

:SYSTem:COMMunicate:GPIB:SGENerator:ADDRess <numeric>

:SYSTem:COMMunicate:GPIB:SGENerator:ADDRess?

{value}<newline><^END>

```
10 OUTPUT 717;":SYST:COMM:GPIB:SGEN:ADDR 19"
20 OUTPUT 717;":SYST:COMM:GPIB:SGEN:ADDR?"
30 ENTER 717;A
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.FREQuenc
= *Param*

Param =
SCPI.COMMunicate.GPIB.SGENerator.CCOMmand.FREQuency

/

1

	Value
	(String)
	254
	“FR %f% HZ”
	“%f%.”

```
Dim SGcomm As String
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.FREQuer
= "FREQ %f%HZ"
SGcomm =
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.FREQuer
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.POWer
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.PRESet
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.RFON
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWELl
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.TYPE

System > **Misc Setup** > **GPIB Setup** > **Signal Generator Address** > **Custom Commands** > **Set Frequency**

> **Misc** > **GPIB** > > >

SCPI

:SYSTem:COMMunicate:GPIB:SGENerator:CCOMmand:FREQuency <string>

:SYSTem:COMMunicate:GPIB:SGENerator:CCOMmand:FREQuency?

{string}<newline><^END>

```
10 OUTPUT 717;":SYST:COMM:GPIB:SGEN:CCOM:FREQ ""FREQ
%f%HZ"""
20 OUTPUT 717;":SYST:COMM:GPIB:SGEN:CCOM:FREQ?"
30 ENTER 717;A$
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.POWer = *Param*

Param = SCPI.COMMunicate.GPIB.SGENerator.CCOMmand.POWer

/

1

	Value
	(String)
	254
	“AP %p% DM”
	“%p%”

```
Dim SGcomm As String
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.POWer =
"AMPL %p%DBM"
SGcomm =
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.POWer
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.FREQuenc
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.PRESet
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.RFON
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWELl
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.TYPE

System> Misc Setup > GPIB Setup > Signal Generator Address > Custom Commands > Set Power Level

> Misc > GPIB > > >

SCPI

:SYSTem:COMMunicate:GPIB:SGENerator:CCOMmand:POWer <string>

:SYSTem:COMMunicate:GPIB:SGENerator:CCOMmand:POWer?

{string}<newline><^END>

```
10 OUTPUT 717;":SYST:COMM:GPIB:SGEN:CCOM:POW ""AMPL
%p%DBM"""
20 OUTPUT 717;":SYST:COMM:GPIB:SGEN:CCOM:POW?"
30 ENTER 717;A$
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.PRESet = *Param*

Param = SCPI.COMMunicate.GPIB.SGENerator.CCOMmand.POWer

/

1

	Value
	(String)
	254
	“”

```
Dim SGcomm As String
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.PRESet =
"*RST"
SGcomm =
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.PRESet
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.FREQuenc
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.POWer
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.RFON
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWELl
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.TYPE

System > **Misc Setup** > **GPIB Setup** > **Signal Generator Address** > **Custom Commands** > **Preset**

> **Misc** > **GPIB** > > >

SCPI

:SYSTem:COMMunicate:GPIB:SGENerator:CCOMmand:PRESet <string>

:SYSTem:COMMunicate:GPIB:SGENerator:CCOMmand:PRESet?

{string}<newline><^END>

```
10 OUTPUT 717;":SYST:COMM:GPIB:SGEN:CCOM:PRES ""*RST""""
20 OUTPUT 717;":SYST:COMM:GPIB:SGEN:CCOM:PRES?"
30 ENTER 717;A$
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.RFON = *Param*

Param = SCPI.COMMunicate.GPIB.SGENerator.CCOMmand.RFON

/RF

1

	Value
	RF
	(String)
	254
	“R3”

```
Dim SGcomm As String
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.RFON =
"AMPL:STATE ON"
SGcomm =
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.RFON
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.FREQuenc
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.POWer
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.PRESet
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWELl
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.TYPE

System > **Misc Setup** > **GPIB Setup** > **Signal Generator Address** > **Custom Commands** > **Turn RF Out On**

> **Misc** > **GPIB** > > > **RF**

SCPI

:SYSTem:COMMunicate:GPIB:SGENerator:CCOMmand:RFON <string>

:SYSTem:COMMunicate:GPIB:SGENerator:CCOMmand:RFON?

{string}<newline><^END>

```
10 OUTPUT 717;":SYST:COMM:GPIB:SGEN:CCOM:RFON
""AMPL:STATE ON""""
20 OUTPUT 717;":SYST:COMM:GPIB:SGEN:CCOM:RFON?"
30 ENTER 717;A$
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWELl

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWELl = *Value*

Value = SCPI.COMMunicate.GPIB.SGENerator.DWELl

/

	Value
	(Double)
	01
	0.1
	s
	0.001

```
Dim SGdelay As Double
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWELl = 0.2
SGdelay = SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWELl
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.FREQuenc
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.POWer
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.PRESet
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.RFON
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.TYPE

System > **Misc Setup** > **GPIB Setup** > **Signal Generator Address** > **Switching Time**

> **Misc** > **GPIB** > >

SCPI

:SYSTem:COMMunicate:GPIB:SGENerator:DWELl <numeric>

:SYSTem:COMMunicate:GPIB:SGENerator:DWELl?

{numeric}<newline><^END>

```
10 OUTPUT 717;":SYST:COMM:GPIB:SGEN:DWEL 0.2"
20 OUTPUT 717;":SYST:COMM:GPIB:SGEN:DWEL?"
30 ENTER 717;A
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.TYPE

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.TYPE = *Value*

Value = SCPI.COMMunicate.GPIB.SGENerator.TYPE

/

	Value
	1: 2: 8643A8644B8664A8665A/B 3: 8648A/B/C/DESGPSG
	(long)
	13
	3

```
Dim SGtype As long
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.TYPE = 2
SGtype = SCPI.SYSTem.COMMunicate.GPIB.SGENerator.TYPE
```

SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.FREQuenc
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.POWer
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.PRESet
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.CCOMmand.RFON
SCPI.SYSTem.COMMunicate.GPIB.SGENerator.DWELl

System > **Misc Setup** > **GPIB Setup** > **Signal Generator Address** > **Custom Commands|8643A,8644B,8664A,8665A/B|8648A/B/C/D,ESG Series,PSG Series**

> **Misc** > **GPIB** > > **|8643A8644B8664A8665A/B|8648A/B/C/D ESGPSG**

SCPI

:SYSTem:COMMunicate:GPIB:SGENerator:TYPE <numeric>

:SYSTem:COMMunicate:GPIB:SGENerator:TYPE?

{value}<newline><^END>

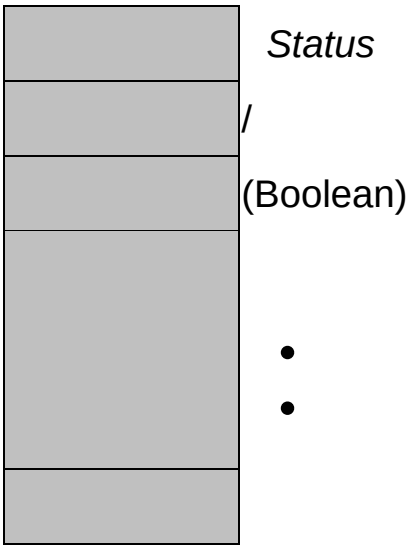
```
10 OUTPUT 717;":SYST:COMM:GPIB:SGEN:TYPE 2"
20 OUTPUT 717;":SYST:COMM:GPIB:SGEN:TYPE?"
30 ENTER 717;A
```

SCPI.SYSTem.CORRection.STATe

SCPI.SYSTem.CORRection.STATe = *Status*

Status = SCPI.SYSTem.CORRection.STATe

/


Status
/
(Boolean)

```
Dim SysCal As Boolean
SCPI.SYSTem.CORRection.STATe = False
SysCal = SCPI.SYSTem.CORRection.STATe
```

System > **Misc Setup** > **Service Menu** > **System Correction** > **Misc** > >

SCPI

:SYSTem:CORRection[:STATe] {ON|OFF|1|0}

:SYSTem:CORRection[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SYST:CORR OFF"
20 OUTPUT 717;":SYST:CORR?"
30 ENTER 717;A
```

SCPI.SYSTem.DATE

SCPI.SYSTem.DATE = *Data*

Data = SCPI.SYSTem.DATE

/E5071C

	Data
	3 • *0* • *1* • *2* 0
	(Variant)
	• *Data(0)* 1980 to 2099 • *Data(1)* 1 to 12 • *Data(2)* 1 to 31
	1

```
Dim Day As Variant
SCPI.SYSTem.DATE = Array(2001,12,24)
Day = SCPI.SYSTem.DATE
```

```
Dim Day(2) As Variant
Dim Ref As Variant
Day(0) = 2001
Day(1) = 12
Day(2) = 24
SCPI.SYSTem.DATE = Day
Ref = SCPI.SYSTem.DATE
```

SCPI.SYSTem.TIME

SCPI.DISPlay.CLOCk

System > **Misc Setup** > **Clock Setup** > **Set Date and Time**
> **Misc** > >

SCPI

:SYSTem:DATE <numeric 1>,<numeric 2>,<numeric 3>

:SYSTem:DATE?

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

```
10 OUTPUT 717;":SYST:DATE 2002,1,1"
20 OUTPUT 717;":SYST:DATE?"
30 ENTER 717;A,B,C
```

SCPI.SYSTem.ERRor

Err = SCPI.SYSTem.ERRor

E5071C100

SCPI.IEEE4882.CLS

E5071CSCPI

	Err
	2 • *Err(0)* • *Err(1)* 0
	(Variant)
	0“”

```
Dim Err As Variant
Err = SCPI.SYSTem.ERRor
```

SCPI.IEEE4882.CLS

SCPI

:SYSTem:ERRor?

{numeric},{string}<newline><^END>

{numeric}:

{string}:

"

0“”

```
10 OUTPUT 717;":SYST:ERR?"
20 ENTER 717;A,B$
```

SCPI.SYSTem.ISPC.PORT

SCPI.SYSTem.ISPControl.PORT = *Value*

Status = SCPI.SYSTem.ISPControl.PORT

/

	Value
	(Long)
	14
	1

```
SCPI.INIT(ch).CONT = False
SCPI.SYST.ISPC.STAT = True
SCPI.SYST.ISPC.PORT = 1
```

SCPI.SYSTem.ISPC.STAT

System > **Service** > **Init Src Port [1|2|3|4]**

> > **Init Src Port [1|2|3|4]**

SCPI

:SYSTem:ISPControl:PORT <numeric>

:SYSTem:ISPControl:PORT?

{1-4}<newline><^END>

```
10 :INIT:CONT OFF
20 :SYSTem:ISPControl[:STATe] ON
30 :SYSTem:ISPControl:PORT 1
```

SCPI.SYSTem.ISPC.STAT

SCPI.SYSTem.ISPControl.STATe = *Status*

Status = SCPI.SYSTem.ISPControl.STATe

/

Status

/

(Boolean)

-
-

```
SCPI.INIT(ch).CONT = False
SCPI.SYST.ISPC.STAT = True
SCPI.SYST.ISPC.PORT = 1
```

SCPI.SYSTem.ISPC.PORT

System > **Service** > **Init Src Ctrl [ON]**

> > **Init Src Ctrl [ON]**

SCPI

:SYSTem:ISPControl [:STATe] {ON|OFF|1|0}

:SYSTem:ISPControl [:STATe] ?

{1|0}<newline><^END>

```
10 :INIT:CONT OFF
20 :SYSTem:ISPControl[:STATe] ON
30 :SYSTem:ISPControl:PORT 1
```

SCPI.SYSTem.KLOCk.KBD

SCPI.SYSTem.KLOCk.KBD = *Status*

Status = SCPI.SYSTem.KLOCk.KBD

/

Status

/

(Boolean)

-
-

```
Dim FKLock As Boolean
SCPI.SYSTem.KLOCk.KBD = True
FKLock = SCPI.SYSTem.KLOCk.KBD
```

SCPI.SYSTem.KLOCk.MOUSe

System > **Misc Setup** > **Front Panel & Keyboard Lock**
> **Misc** > **&**

SCPI

:SYSTem:KLOCk:KBD {ON|OFF|1|0}

:SYSTem:KLOCk:KBD?

{1|0}<newline><^END>

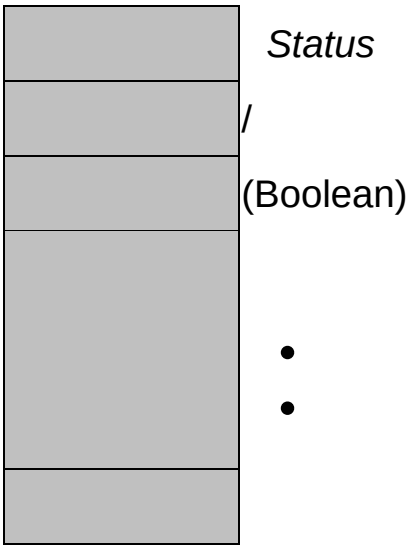
```
10 OUTPUT 717;":SYST:KLOC:KBD ON"
20 OUTPUT 717;":SYST:KLOC:KBD?"
30 ENTER 717;A
```

SCPI.SYSTem.KLOCk.MOUSe

SCPI.SYSTem.KLOCk.MOUSe = *Status*

Status = SCPI.SYSTem.KLOCk.MOUSe

/


Status
/
(Boolean)

```
Dim MTLock As Boolean
SCPI.SYSTem.KLOCk.MOUSe = True
MTLock = SCPI.SYSTem.KLOCk.MOUSe
```

SCPI.SYSTem.KLOCk.KBD

System > **Key Lock** > **Mouse Lock**

> >

SCPI

:SYSTem:KLOCk:MOUSe {ON|OFF|1|0}

:SYSTem:KLOCk:MOUSe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SYST:KLOC:MOUS ON"
20 OUTPUT 717;":SYST:KLOC:MOUS?"
30 ENTER 717;A
```

SCPI.SYSTem.POFF

SCPI.SYSTem.POFF

E5071C

SCPI.SYSTem.POFF

Standby switch

SCPI

:SYSTem:POFF

```
10 OUTPUT 717;":SYST:POFF"
```

SCPI.SYSTem.PRESet

SCPI.SYSTem.PRESet

E5071C SCPI.IEEE4882.RST1
SCPI.INITiate(Ch).CONTinuous

SCPI.SYSTem.PRESet

SCPI.IEEE4882.RST

Preset > **OK**

>

SCPI

:SYSTem:PRESet

```
10 OUTPUT 717;":SYST:PRES"
```

SCPI.SYSTem.SECurity.LEVel

SCPI.SYSTem.SECurity.LEVel = Param

Param = SCPI.SYSTem.SECurity.LEVel

/E5071C

	Param
	(String)
	• “NON” • “LOW” • “HIGH”
	“NON”
	LOWE5071CNONHIGHHIGHNONLOW HIGHNON LOWHIGH

```
Dim SecLev As String
SCPI.SYSTem.SECurity.LEVel = "LOW"
SecLev = SCPI.SYSTem.SECurity.LEVel
```

System > **Service Menu** > **Security Level** > **None|Low|High**

> > > ||

SCPI

:SYSTem:SECurity:LEVel {NON|LOW|HIGH}

:SYSTem:SECurity:LEVel?

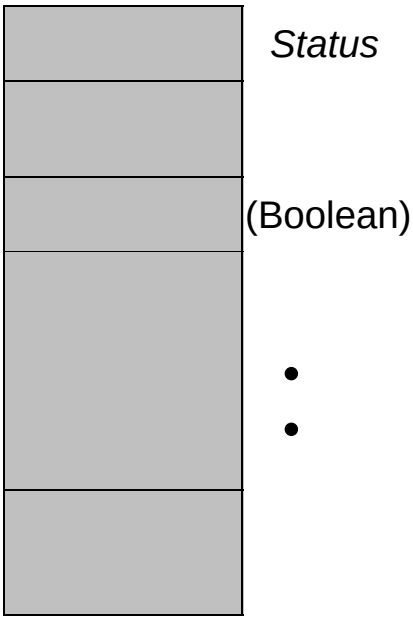
{NON|LOW|HIGH}<newline><^END>

```
10 OUTPUT 717;":SYST:SEC:LEV LOW"
20 OUTPUT 717;":SYST:SEC:LEV?"
30 ENTER 717;A$
```

SCPI.SYSTem.SERVice

Status = SCPI.SYSTem.SERVice

E5071C


Status
(Boolean)

```
Dim SvMode As Boolean
SvMode = SCPI.SYSTem.SERVice
```

LCD

SCPI

:SYSTem:SERVice?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SYST:SERV?"
30 ENTER 717;A
```

SCPI.SYSTem.TEMPerature.HIGH

SCPI.SYSTem.TEMPerature.HIGH = *Status*

Status = SCPI.SYSTem.TEMPerature.HIGH

E5071CE5070B/E5071B

	Status
	(Boolean)
	• •

```
Dim TempMode As Boolean
SCPI.SYSTem.TEMPerature.HIGH = True
TempMode = SCPI.SYSTem.TEMPerature.HIGH
```

SCPI

:SYSTem:TEMPerature:HIGH {ON|OFF|1|0}

:SYSTem:TEMPerature:HIGH?

{1|0}<newline><^END>

```
10 OUTPUT 717;":SYST:TEMP:HIGH ON"
20 OUTPUT 717;":SYST:TEMP:HIGH?"
30 ENTER 717;A
```

SCPI.SYSTem.TEMPerature.STATe

Status = SCPI.SYSTem.TEMPerature.STATe

E5071CE5070B/E5071B

	Status
	(Boolean)

```
Dim WarmUp As Boolean
WarmUp = SCPI.SYSTem.TEMPerature.STATe
```

SCPI

:SYSTem:TEMPerature[:STATe]?

{1}<newline><^END>

```
10 OUTPUT 717;":SYST:TEMP?"
30 ENTER 717;A
```

SCPI.SYSTem.TIME

SCPI.SYSTem.TIME = *Data*

Data = SCPI.SYSTem.TIME

/E5071C

	Data
	3 *0* 24 *1* *2* 0
	(Variant)
	0 023 *1* 059 *2* 059
	1

```
Dim Time As Variant
SCPI.SYSTem.TIME = Array(21,30,0)
Time = SCPI.SYSTem.TIME
```

```
Dim Time(2) As Variant
Dim Ref As Variant
Time(0) = 21
Time(1) = 30
Time(2) = 0
SCPI.SYSTem.TIME = Time
Ref = SCPI.SYSTem.TIME
```

SCPI.SYSTem.DATE

SCPI.DISPlay.CLOCk

System > **Misc Setup** > **Clock Setup** > **Set Date and Time**
> **Misc** > >

SCPI

:SYSTem:TIME <numeric 1>,<numeric 2>,<numeric 3>

:SYSTem:TIME?

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

```
10 OUTPUT 717;":SYST:TIME 17,30,0"
20 OUTPUT 717;":SYST:TIME?"
30 ENTER 717;A,B,C
```

SCPI.SYSTem.TSET.EXTernal

SCPI.SYSTem.TSET.EXTernal = *Param*

Param = SCPI.SYSTem.TSET.EXTernal

/ MODE1/MODE2S11S21

	Param
	(String)
	• “OFF” • “MODE1”MODE1 • “MODE2”MODE2
	“OFF”

```
Dim ExtTset As String
SCPI.SYSTem.TSET.EXTernal = "MODE1"
ExtTset = SCPI.SYSTem.TSET.EXTernal
```

System > **Service Menu** > **External Test Set**

> >

SCPI

:SYSTem:TSET:[EXTernal] {OFF|MODE1|MODE2}

:SYSTem:TSET:[EXTernal]?

{OFF|MODE1|MODE2}<newline><^END>

```
10 OUTPUT 717;":SYST:TSET MODE1"
20 OUTPUT 717;":SYST:TSET?"
30 ENTER 717;A
```

SCPI.SYSTem.UPReset

SCPI.SYSTem.UPReset

E5071C

D:\UserPreset.sta SCPI.SYSTem.PRESet

CPI.SYSTem.UPReset

SCPI.IEEE4882.RST

SCPI.SYSTem.PRESet

Preset > **OK**

>

SCPI

:SYSTem:UPReset

```
10 OUTPUT 717;":SYST:UPR"
```

SCPI.TRIGger.OUTPut.POLarity

SCPI.TRIGger.OUTPut.POLarity=*param*

param=SCPI.TRIGger.OUTPut.POLarity

/

	Param
	/
	(String)
	• “POSitive” • “NEGative”
	“POSitive”

```
Dim TrigPol As String
TrigPol="NEGative"
SCPI.TRIGger.OUTPut.POLarity=TrigPol
```

SCPI.TRIGger.OUTPut.POSition

SCPI.TRIGger.OUTPut.STATe

Trigger > Polarity

>

SCPI

:TRIGger:OUTPut.POLarity {POSitive|NEGative}

:TRIGger:OUTPut.POLarity?

{POSitive|NEGative}<newline><^END>

```
10 OUTPUT 717;":TRIG:OUTP:POL POS"
20 OUTPUT 717;":TRIG:OUTP:POL?"
30 ENTER 717;A$
```

SCPI.TRIGger.OUTPut.POSition

SCPI.TRIGger.OUTPut.POSition=Param

Param=SCPI.TRIGger.OUTPut.POSition

/

	Param
	/
	(String)
	• “AFTe” • “BEFore”
	“AFTer”

```
Dim TrigPos As String
TrigPos="BEFore"
SCPI.TRIGger.OUTPut.POSition=TrigPos
```

SCPI.TRIGger.OUTPut.POLarity

SCPI.TRIGger.OUTPut.STATe

Trigger > **Position**

>

SCPI

:TRIGger:OUTPut.POSition {AFTer|BEFore}

:TRIGger:OUTPut.POSition?

{AFTer|BEFore}<newline><^END>

```
10 OUTPUT 717;":TRIG:OUTP:POS AFT"
20 OUTPUT 717;":TRIG:OUTP:POS?"
30 ENTER 717;A$
```

SCPI.TRIGger.OUTPut.STATe

SCPI.TRIGger.OUTPut.STATe=Param

Param=SCPI.TRIGger.OUTPut.STATe

/

	Param
	/
	Boolean
	• •
	“OFF”

```
Dim TrigStat As boolean
TrigStat=1
SCPI.TRIGger.OUTPut.STATe=TrigStat
```

SCPI.TRIGger.OUTPut.POLarity
SCPI.TRIGger.OUTPut.POSition

Trigger > **Ext Trig Output**

>

SCPI

:TRIGger:OUTPut.STATe {ON|1|OFF|0}

:TRIGger:OUTPut.STATe?

{1|0}<newline><^END>

```
10 OUTPUT 717;":TRIG:OUTP:STAT ON"
20 OUTPUT 717;":TRIG:OUTP:STAT?"
30 ENTER 717;A$
```

SCPI.TRIGger.SEQuence.AVERage

SCPI.TRIGger.SEQuence.AVERage = *Status*

Status = SCPI.TRIGger.SEQuence.AVERage

/

	Status
	/
	(Boolean)
	• •

```
Dim Avetrig As Boolean
SCPI.TRIGger.SEQuence.AVERage = True
Avetrig = TRIGger.SEQuence.AVERage
```

SCPI.SENSe(Ch).AVERage.STATe

>

SCPI

:TRIGger[:SEQuence]:AVERage {ON|OFF|1|0}

:TRIGger[:SEQuence]:AVERage?

{1|0}<newline><^END>

```
10 OUTPUT 717;":TRIG:AVER ON"
20 OUTPUT 717;":TRIG:AVER?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.EXTernal.DELay

SCPI.TRIGger.SEQuence.EXTernal.DELay = *Value*

Value = SCPI.TRIGger.SEQuence.EXTernal.DELay

/

	Value
	(Double)
	01
	0
	s
	10 m

```
Dim Extdel As Double
SCPI.TRIGger.SEQuence.EXTernal.DELay = 0.05
Extdel = SCPI.TRIGger.SEQuence.EXTernal.DELay
```

SCPI.TRIGger.SEQuence.EXTernal.LLATency.STATe

SCPI.TRIGger.SEQuence.POINt

SCPI

:TRIGger[:SEQuence]:EXTernal:DELay <numeric>

:TRIGger[:SEQuence]:EXTernal:DELay?

{numeric}<newline><^END>

```
10 OUTPUT 717;":TRIG:EXT:DEL 0.05"
20 OUTPUT 717;":TRIG:EXT:DEL?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.EXTernal.LLATency.STATe

SCPI.TRIGger.SEQuence.EXTernal.LLATency.STATe = *Status*

Status = SCPI.TRIGger.SEQuence.EXTernal.LLATency.STATe

/

	Status
	/
	(Boolean)
	• •

```
Dim Llat As Boolean
SCPI.TRIGger.SEQuence.EXTernal.LLATency.STATe = True
Llat = SCPI.TRIGger.SEQuence.EXTernal.LLATency.STATe
```

SCPI.TRIGger.SEQuence.EXTernal.DELay
SCPI.TRIGger.SEQuence.POINt
SCPI.TRIGger.SEQuence.SOURce

Trigger > **Low Latency**

>

SCPI

:TRIGger[:SEQuence]:EXTernal:LLATency[:STATe] {ON|OFF|1|0}

:TRIGger[:SEQuence]:EXTernal:LLATency[:STATe]?

{1|0}<newline><^END>

```
10 OUTPUT 717;":TRIG:EXT:LLAT ON"
20 OUTPUT 717;":TRIG:EXT:LLAT?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.EXTernal.SLOPe

SCPI.TRIGger.SEQuence.EXTernal.SLOPe

/

	Param
	/
	(String)
	• “POSitive”/ • “NEGative”/
	“NEGative”

```
Dim InputTrigPol As String
InputTrigPol="NEGative"
SCPI.TRIGger.SEQuence.EXTernal.SLOPe=InputTrigPol
```

SCPI.TRIGger.SEQuence.EXTernal.DELay

SCPI.TRIGger.SEQuence.EXTernal.LLATency.STATe

Trigger > **Ext Trig Input**

>

SCPI

:TRIGger:SEQuence:EXTernal:SLOPe {POSitive|NEGative}

:TRIGger:SEQuence:EXTernal:SLOPe?

{POSitive|NEGative}<newline><^END>

```
10 OUTPUT 717;":TRIG:SEQ:EXT:SLOP POS"
20 OUTPUT 717;":TRIG:SEQ:EXT:SLOP?"
30 ENTER 717;A$
```

SCPI.TRIGger.SEQuence.IMMediate

SCPI.TRIGger.SEQuence.IMMediate

SCPI.TRIGger.SEQuence.SINGle

```
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.INITiate(1).CONTinuous = True
SCPI.TRIGger.SEQuence.IMMediate
```

SCPI.TRIGger.SEQuence.IMMediate

SCPI

:TRIGger[:SEQuence]:IMMediate

```
10 OUTPUT 717;":TRIG:IMM"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.POINt

SCPI.TRIGger.SEQuence.POINt = *Status*

Status = SCPI.TRIGger.SEQuence.POINt

ON/OFF

	Status
	/
	(Boolean)

```
Dim Ptrig As Boolean
SCPI.TRIGger.SEQuence.POINt = True
Ptrig = TRIGger.SEQuence.POINt
```

SCPI.TRIGger.SEQuence.SOURce

Trigger > **Trigger Event**

>

SCPI

:TRIGger[:SEQuence]:POINt {ON|OFF|1|0}

:TRIGger[:SEQuence]:POINt?

{1|0}<newline><^END>

```
10 OUTPUT 717;":TRIG:POIN ON"
20 OUTPUT 717;":TRIG:POIN?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.SINGle

SCPI.TRIGger.SEQuence.SINGle

SCPI.TRIGger.SEQuence.IMMediate
SCPI.IEEE4882.OPC

```
Dim Dmy As Long
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.INITiate(1).CONTinuous = True
SCPI.TRIGger.SEQuence.SINGle
Dmy = SCPI.IEEE4882.OPC
```

SCPI.TRIGger.SEQuence.IMMediate

SCPI.IEEE4882.OPC

SCPI

:TRIGger[:SEQuence]:SINGle

```
10 OUTPUT 717;":TRIG:SING"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.SOURce

SCPI.TRIGger.SEQuence.SOURce = *Param*

Param = SCPI.TRIGger.SEQuence.SOURce

4/

1.

2.

3. **Trigger** > **Trigger**

4. SCPI.IEEE4882.TRG

	Param
	(String)
	• “INTernal” • “EXTernal” • “MANual” • “BUS”
	“INTernal”

```
Dim TrigSour As String
SCPI.TRIGger.SEQuence.SOURce = "bus"
TrigSour = SCPI.TRIGger.SEQuence.SOURce
```

Trigger > **Trigger Source** > **Internal|External|Manual|Bus**

> > |||

SCPI

:TRIGger[:SEQuence]:SOURce {INTernal|EXTernal|MANual|BUS}

:TRIGger[:SEQuence]:SOURce?

{BUS|EXT|INT|MAN}<newline><^END>

```
10 OUTPUT 717;":TRIG:SOUR BUS"
20 OUTPUT 717;":TRIG:SOUR?"
30 ENTER 717;A$
```

E5091A

E5071C E5091A

- E5071C E5091A
- E5091A
- .
- E5091A

E5071C E5091A

-
- E5071C E5091A
- E5091A

E5091A

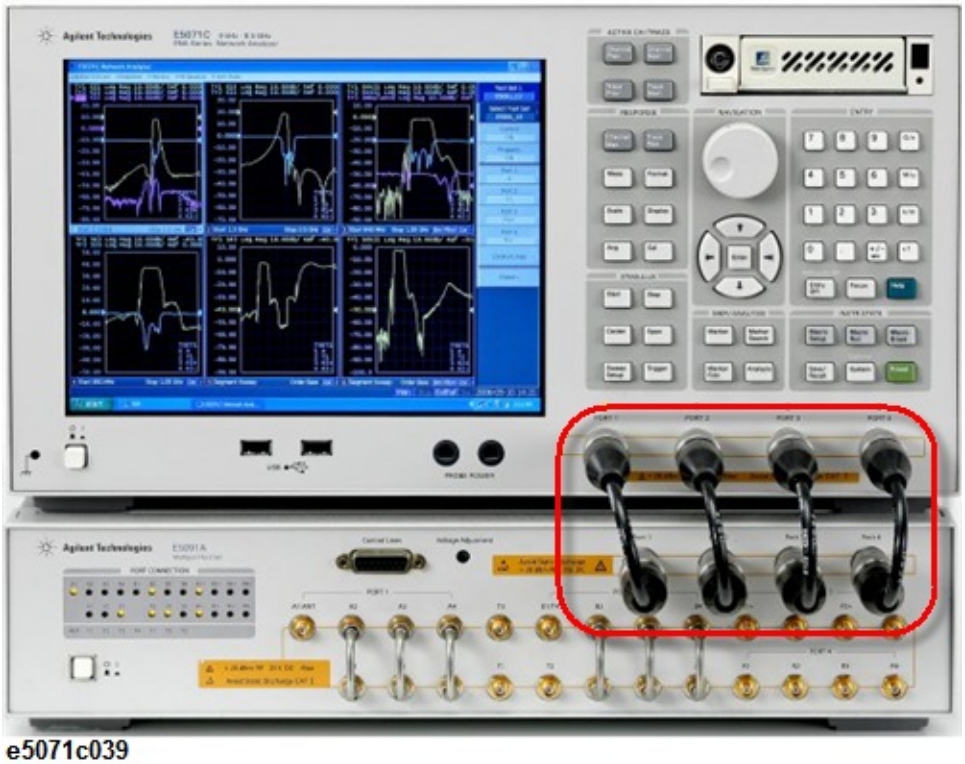
E5071C E5091A

- E5071C
- E5091A
- N N E5091A Agilent 8120-4782
- USB E5091A Agilent 8121-0770

E5071C E5091A

N

N E5071C E5091A E5071C E5091A

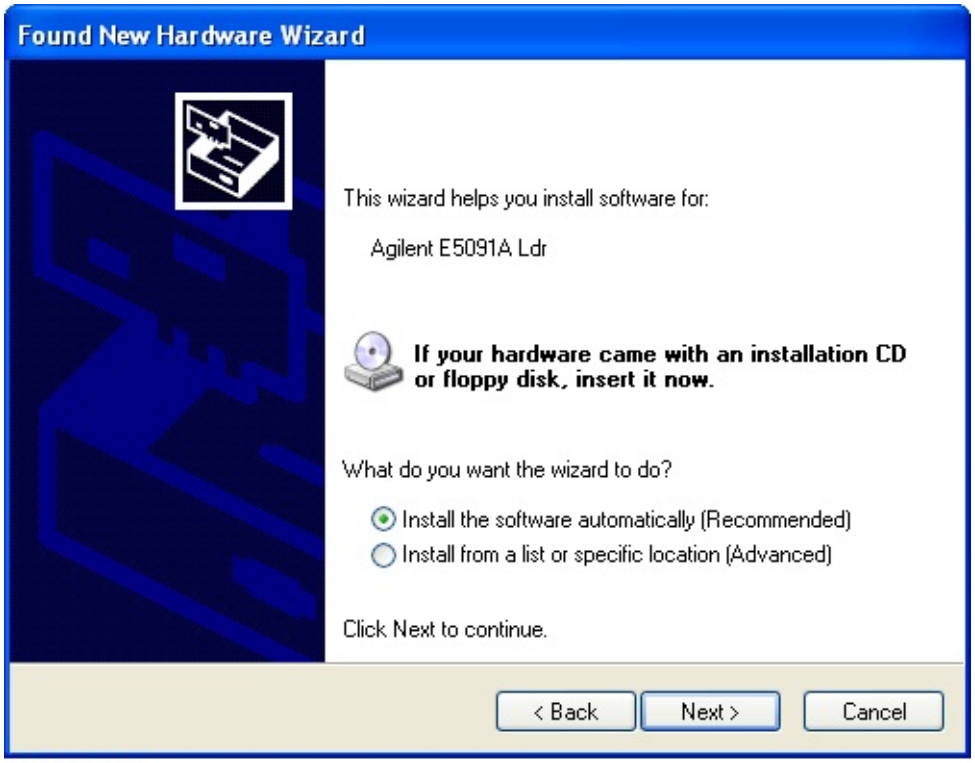


e5071c039

USB

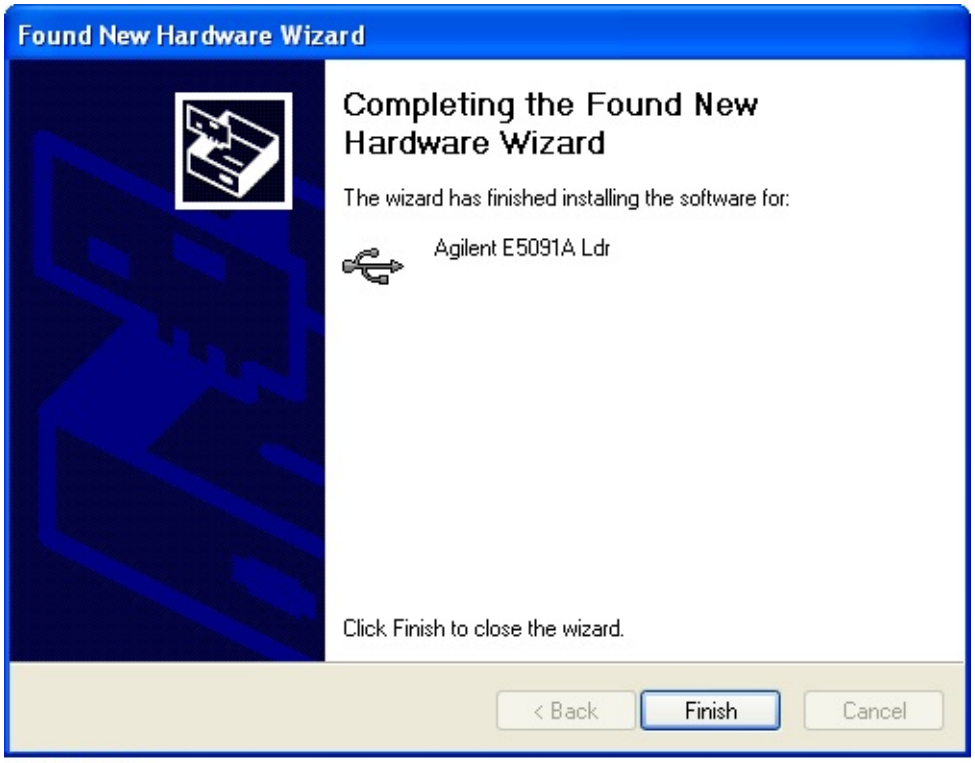
1. USB E5071C USB E5091A USB
2. E5071C E5091A
3. “ **Found New Hardware Wizard**”“ **No, not this time**” “ **Next**”

4. “Install the software automatically”“ **Next**”



e5071c174

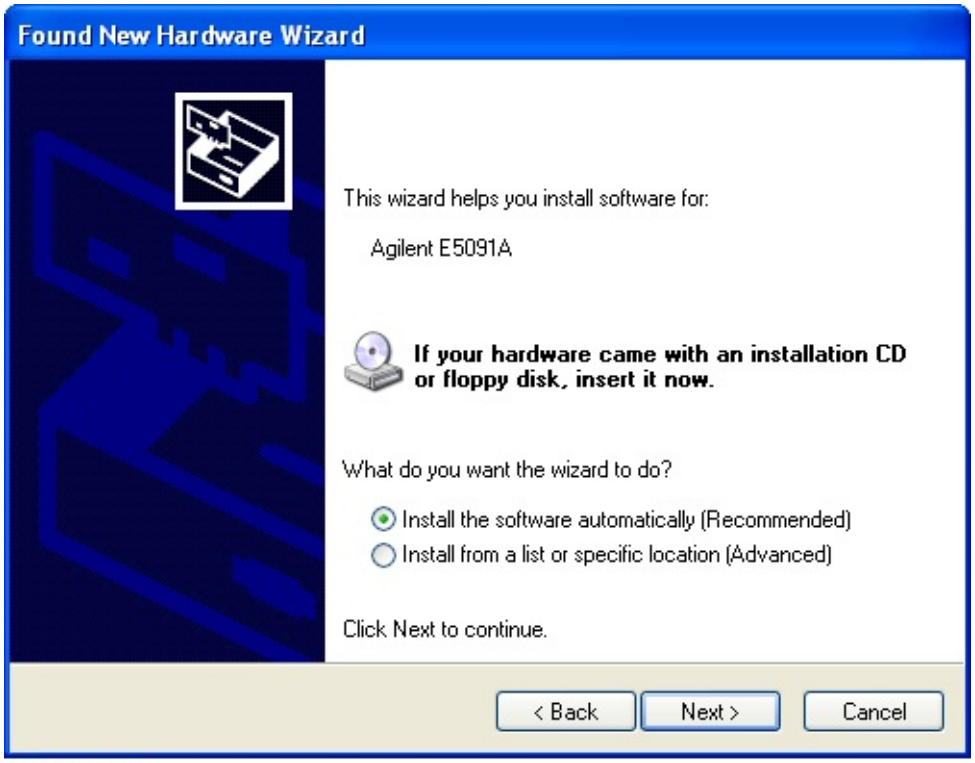
5. “ **Finish**”



e5071c176

6. “**Found New Hardware Wizard**”“ **No, not this time**” “ **Next**”

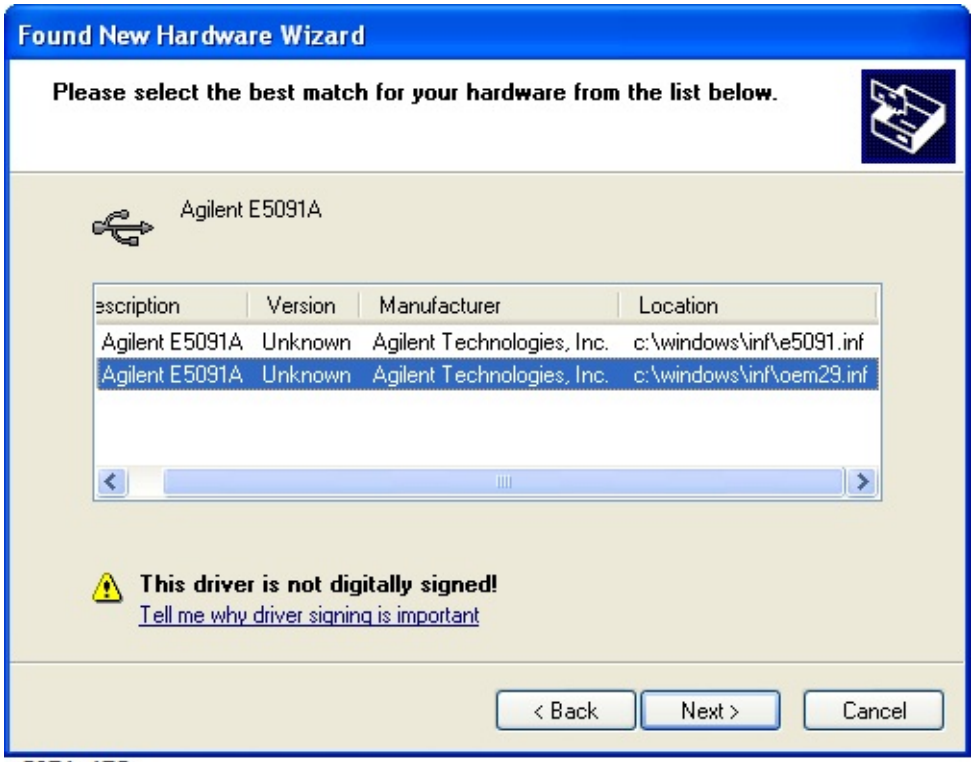
7. “Install the software automatically”“ **Next**”



e5071c170

8. “Location” **C:\windows\inf\oemxx.inf**xx

oem29.inf　Agilent E5091A“　Next”



e5071c175

9. “　**Finish**”

e5071c171

10. E5071C E5091A LED

- E5071C E5091A / USB USB
  USB

- USB E5091A USB

E5091A

E5091A

- ID ID
- E5091A DUT

E5091A

E5091A

E5091A

E5091A ID	E5091A ID
E5091A	E5091A
	E5071C E5091A
E5091A	E5091A
	DUT
E5091A	E5091A

E5091A

E5091A ID

1. ID E5091A ID
2. “ **System**” >“ **Multiport Test Set Setup**” E5091A
3. ID 1 “ **Test Set 1**” 1 ID 2 “ **Test Set 2**” 2
   ID E5091A

- E5071C ID

E5091A

E5091A E5091A-016 13 16

1. “ **System**”>“ **Multiport Test Set Setup**”>“ **Test Set 1**” 1“ **Test Set 2**” 2
2. “ **Select Test Set**”
3. E5091A

E5091_9	E5091A 009
E5091_13	13 E5091A 016
E5091_16	16 E5091A 016

	E5091_009	E5091_013	E5091_016
E5091A-009			
E5091A-016			

E5091A

1. “ **System**”“ **Multiport Test Set Setup**”>“ **Test Set 1**” 1“ **Test Set 2**” 2
2. “ **Channel Next**”“ **Channel Prev**”
3. E5091A
4. 2 3

E5091A

E5091A-009

Port1 1	E5091A 1 E5091A A T1
Port2 2	E5091A 2 E5091A T1 T2
Port3 3	E5091A 3 E5091A R1+R2+ R3+
Port4 4	E5091A 4 E5091A R1-R2- R3-

E5091A-013

Port1 1	E5091A 1 E5091A A1T1T2 T3
Port2 2	E5091A 2 E5091A T1T2T3 T4

Port3 **3**	E5091A 3 E5091A R1+R2+R3+ R4+
Port4 **4**	E5091A 4 E5091A R1-R2-R3- R4-

E5091A-016

Port1 **1**	E5091A 1 E5091A A1A2A3A4A T1T2 T3
Port2 **2**	E5091A 2 E5091A B1B2B3B4T1 T2T3 T4
Port3 **3**	E5091A 3 E5091A R1+R2+R3+ R4+
Port4 **4**	E5091A 4 E5091A R1-R2-R3- R4
Port5 **5**	E5091A 5 E5091A X1 X2
Port6 **6**	E5091A 6 E5091A Y1 Y2
Port7 **7**	E5091A 7 E5091A Z1 Z2

E5091A

E5091A

E5091A

e5071c177

1. “ **System**”“ **Multiport Test Set Setup**”>“ **Test Set 1**” 1“ **Test Set 2**” 2
2. “ **Property**”“ **ON**” E5091A

E5091A “ **ON**”/“ **OFF**”

E5071C E5091A DUT

e5071c206

1. “ **System**”“ **Multiport Test Set Setup**”>“ **Test Set 1**” 1“ **Test Set 2**” 2
2. “ **Channel Next**”“ **Channel Prev**”
3. “ **Control Lines**” DUT
4. E5091A

“**Line 1**” 1“ **Line 8**” 8	“ **High/Low**”/

5. 3 4

E5091A

E5091A E5091A

- E5091A E5071C 3 E5091A

1. “ **System**”“ **Multiport Test Set Setup**”>“ **Test Set 1**” 1“ **Test Set 2**” 2
2. “ **Control**”“ **ON**”E5091A

- E5091A “ **ON**”/“ **OFF**”

-
-

E5091A

E5091A

1. “ **Channel Next**”“ **Channel Prev**”
2. E5091A E5091A
3. E5071C E5091A E5091A

E5091A E5071C

E5091A

E5091A

	E5091A
	1. 2. 3. 4. 5.

E5091A

-
- 1 n
-

E5091A

E5091A 1 2

:SENS:MULT{1-2}:NAME

E5091A_009	009
E5091A_013	01313
E5091A_016	01616

E5091A

:SENS:MULT:CAT?

/

/ E5091A

:SENS:MULT{1-2}:STAT

E5091A E5071C 1 2

1 n

1 n

:SENS{1-36}:MULT{1-2}:PORT{1-20}

E5091A 1 n

/

1 n E5091 /

:SENS:MULT{1-2}:DISP

E5091A

:SENS:MULT{1-2}:COUN?

E5091A

:SENS{1-36}:MULT{1-2}:PORT{1-20}:CAT?

E5091A

:SENS:MULT{1-2}:INC?

HIGH/LOW/ HIGH/LOW/

:SENS{1-36}:MULT{1-2}:TSET9:OUTP

E5091A HIGH/LOW/

E5091A

-
-
-
-
-
-
-
-
-
- E5071C 8753ES

E5071C

	9 KHz 3 GHz	9 KHz 8.5 GHz	100 KHz 3 GHz	100 KHz 8.5 GHz
2	230	280	235	285
4	430	480	435	485

008	
010	
790	MWA

UNQ	
1E5	

810	
820	

A6J	ANSI Z540
1A7	ISO 17025

		Agilent
1CM		5063-9216
1CN		5063-9229
1CP		5063-9223

E5071C

E5071C

E5071C

	VBA I/O
	Windows

E5071C

http://www.agilent.com/find/ena_support/

PDFWebHelp/

PDF E5071C

E5071C http://www.agilent.com/find/assist “ **Select a Country or Area**”/

/ :DISP:CCL

IEEE488.2 GPIB

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

A

20	Additional standard needed	GPIB 1 “OPEN” “SHORT”“LOAD” SENS:CORR:COLL:SAVE SENS:CORR:OFFS:COLL:SAVE 1

B

-168	Block data not allowed	

C

240	Calibration data lost	Agilent Technology
22	Calibration method not selected	SENS:CORR:COLL:METH:xxxx SENS:CORR:OFFS:COLL:METH:xxxx SENS:CORR:COLL:SAVE SENS:CORR:OFFS:COLL:SAVE
-148	Character data not allowed	(') (')
-100	Command error	IEEE488.2 - 11.5.1.1.4

D

-222	Data out of range	-65536 +65536 -9.9e37 +9.9e37 “”“” CALC:FSIM:BAL:TOP:BBAL:PPOR SENS:CORR:COLL:ACQ:OPEN SENS:CORR:COLL:ECAL:SOLT3 SENS:CORR:COLL:CKIT:ORD:LOAD
-104	Data type error	

E

32	ECal module not in appropriate RF pathECal RF	ECal SENS:CORR:COLL:ECAL:SOLTn SENS:CORR:OFFS:COLL:ECAL:xxxx ECal (SENS:CORR:COLL :ECAL:ORI OFF) ECal
-200	Execution error	IEEE488.2 - 11.5.1.1.5
-123	Exponent too large	32,000 IEEE488.2 - 7.7.2.4.1
-178	Expression data not allowed	
-170	Expression error	-171 -179

F

31	Failed to configure ECal module ECal	ECal SENS:CORR:COLL:ECAL:SOLTn SENS:CORR:OFFS:COLL:ECAL :xxxx ECal ECal USB ECal
76	Failed to control signal generator	SENS:OFFS:LOC:STAT ON SENS:OFFS:LOC:CONT ON
102	Failed to copy file	MMEM:COPY
104	Failed to create directory	MMEM:MDIR
103	Failed to delete file	MMEM:DEL
100	Failed to read file	2 Touchstone CALC:FSIM:SEND:PMC:PORT:USER:FIL MMEM:LOAD:FDAT MMEM:STOR:LIM MMEM:LOAD:SEGM VBA MMEM:LOAD:PROG
	Failed to write file	MMEM:LOAD:FDAT MMEM:STOR:LIM MMEM:LOAD:SEGM LCD

101		MMEM:STOR:IMAG VBA MMEM:STOR:PROG
-257	File name error	
-256	File name not found	
107	File transfer failed	MMEM:TRAN
56	Fixture Simulator not allowed	(SENS:OFFS ON) (CALC:FSIM:STAT ON)
24	Frequency offset enabled	(SENS:OFFS ON) SENS:CORR:COLL:OPEN
25	Frequency offset not enabled	(SENS:OFFS OFF) SENS:CORR:OFFS:COLL:OPEN
62	Frequency out of range	(SENS:OFFS ON)

G

-105	GET not allowed GET	(GET) IEEE488.2 - 7.7

H

-114	Header suffix out of range	SCPI

I

-224	Illegal parameter value	CALC:PAR:DEF S 2 S44
-282	Illegal program name	PROG:SEL:NAME VBA
-213	Init ignored	“INIT”
35	Insufficient ECal module memoryECal	ECal ECal
-161	Invalid block data	IEEE488.2 - 7.7.6.2 “END”
28	Invalid calibration method	GPIB
-101	Invalid character	
-141	Invalid character data	
-121	Invalid character in number	“9”

-171	Invalid expression	IEEE488.2 - 7.7.7.2
-103	Invalid separator	
-151	Invalid string data	IEEE488.2 - 7.7.5.2 “END”
-131	Invalid suffix	IEEE488.2 - 7.7.3.2 E5071C

L

53	Log sweep requires 2 octave minimum span 2	1 MHz 2 MHz

M

-109	Missing parameter	SENS{1-6}:SWE:POIN “SENS1:SWE:POIN” “SENS1:SWE:POIN 201”

N

70	No GPIB system controller GPIB	GPIB USB/GPIB USB/GPIB USB/GPIB
52	No valid memory trace	DISP:WIND:TRAC:MEM ON “NORM” CALC:MATH:FUNC
-120	Numeric data error	-121 -129
-128	Numeric data not allowed	

O

200	Option not installed	
-225	Out of memory	

P

-220	Parameter error	-221 -229
-108	Parameter not allowed	“:SENS1:SWE:TYPE LIN, SEGM” “:SENS1:SWE:TYPE LIN”
41	Peak not found	CALC:MARK:FUNC:EXEC CALC:FUNC:EXEC
220	Phase lock loop unlocked	PLL Agilent Technology
221	Port 1 receiver overload 1	1 DUT DUT Agilent Technology
222	Port 2 receiver overload 2	2 DUT DUT Agilent Technology
223	Port 3 receiver overload 3	3 DUT DUT Agilent Technology
224	Port 4 receiver overload 4	4 DUT DUT Agilent Technology

73	Power meter not settled	10
241	Power on test failed	Agilent Technology
61	Power unleveled	
120	Printer error	LCD HCOP:IMM
121	Print failed	120“Printer error”
-284	Program currently running	VBA “RUN” PROG:SEL:STAT RUN
-112	Program mnemonic too long	12 IEEE488.2 - 7.6.1.4.1
-286	Program runtime error	VBA

Q

-430	Query DEADLOCKED	“” IEEE488.2 - 6.3.1.7
-400	Query error	IEEE488.2 - 11.5.1.1.7 6.3
-410	Query INTERRUPTED	“” IEEE488.1 - 6.3.2.3 (DAB) GET
-420	Query UNTERMINATED	“” IEEE488.2 - 6.3.2
-440	Query UNTERMINATED after indefinite response	IEEE488.2 - 6.5.7.5.7
-350	Queue overflow	

R

105	Recall failed	State01.sta MMEM:LOAD:STAT

S

106	Save failed	State01.sta MMEM:STOR:STAT
33	Selected parameter not valid for confidence check	S S
57	SnP request not valid for selected measurementSnP	Touchstone “ON” Touchstone
50	Specified channel hidden	DISP:WIND:ACT
23	Specified error term does not exist	/ (SENS:CORR:COEF) (SENS:CORR:COEF:METH:xxxx.)
21	Specified ports overlapped	CALC:FSIM:BAL:TOP:SSB:PPOR 1,2,3,3
51	Specified trace dose not exist	CALC:PAR:SEL CALC:PAR:COUN
26	Standard not selected	0
-150	String data error	-151 -159

-158	String data not allowed	
-138	Suffix not allowed	
-134	Suffix too long	12 IEEE488.2 - 7.7.3.4
55	Sweep mode changed to stepped sweep.	/“ON”
-102	Syntax error	
-310	System error	“”

T

40	Target value not found	CALC:MARK:FUNC:EXEC CALC:FUNC:EXEC CALC:MARK:BWID:DATA?
-124	Too many digits	0 255 IEEE488.2 - 7.7.2.4.1
-223	Too much data	254
54	Transform, Gate not allowed	/ SENS:OFFS OFF /
-211	Trigger ignored	“TRIG”

U

300	Unable to estimate adapter length	O
-113	Undefined header	“:DISP:WIND1:TABL:MEM ON”“:DISP:WIND1:TRAC1:MEM ON” CALC:FSIM:SEND:DEEM:PORTn:xxxx CALC:FSIM:SEND:PMC:PORTn:xxxx CALC:FSIM:SEND:ZCON:PORTn:Z0:R SENS:CORR:EXT:PORTn:TIME PORTn
34	User characterization not found in module	ECal ECal ECal

V

30	Valid Ecal module not found Ecal	ECal 4 SENS:CORR:COLL:ECAL:SOLT4 2 ECal
71	Valid power meter not found.	USB/GPIB E5071C GPIB E5071C GPIB
72	Valid power sensor not found.No valid sensor is connected to the power meter.	
75	Valid signal generator not found.No valid external signal source is connected.	SENS:OFFS:LOC:STAT ON SENS:OFFS:LOC:CONT ON GPIB USB/GPIB USB/GPIB

/ :DISP:CCL

GPIB/

Auto port extension not allowed	
Calibration extrapolated	ECal
Fixture simulator not allowed	
Log sweep requires 2 octave minimum span 2	1 MHz 2 MHz 4
Peak not found	
Segment table changed	
Sweep mode changed to stepped sweep	“ON”
Target value not	

found	
Transform, Gate not allowed	/ /
User Preset File not found	SYST:UPR (**D:\UserPreset.sta**) (SYST:PRES)

- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- /_____
- _____
- _____
- _____
- _____
- _____
- _____
- _____

Fixture Simulator	Fixture Simulator		OFF
Fixture Simulator	Topology	Device	SE-BAL-
Fixture Simulator	Topology	Port1 (se) 1	1
Fixture Simulator	Topology		-
Fixture Simulator	Topology		--
Fixture Simulator	Topology	Port1 (bal) 1	1-2
Fixture Simulator	Topology		-
Fixture Simulator	Topology	Port2 (bal) 2	2-3-
Fixture Simulator	Topology		3-4-
Fixture		Port2 (se)	

Simulator	Topology	2	2
Fixture Simulator	Topology		--
Fixture Simulator	Topology	Port3 (bal) 3	3-4
Fixture Simulator	Topology		--
Fixture Simulator	Topology	Property	OFF
Fixture Simulator	BalUn -		OFF
Fixture Simulator	Port Matching	Port Matching	OFF
Fixture Simulator	Port Matching	Select Port	1
Fixture Simulator	Port Matching	Select Circuit	None
Fixture Simulator	Port Matching	C	0.000e-12F

Fixture Simulator	Port Matching	G	0.000S
Fixture Simulator	Port Matching	L	0.000e-9H
Fixture Simulator	Port Matching	R	0.0000 Ω
Fixture Simulator	Port Z Conversion Z	Port Z Conversion Z	OFF
Fixture Simulator	Port Z Conversion Z	Port1 Z0 Real 1 Z0	50.000 Ω
Fixture Simulator	Port Z Conversion Z	Port1 Z0 Imag 1 Z0	0.000 Ω
Fixture Simulator	Port Z Conversion Z	Port2 Z0 Real 2 Z0	50.000 Ω
Fixture Simulator	Port Z Conversion Z	Port2 Z0 Imag 2 Z0	0.000 Ω
Fixture Simulator	Port Z Conversion Z	Port3 Z0 Real 3 Z0	50.000 Ω
Fixture Simulator	Port Z Conversion Z	Port3 Z0 Imag 3 Z0	0.000 Ω

Fixture Simulator	Port Z Conversion Z	Port4 Z0 Real 4 Z0	50.000 Ω
Fixture Simulator	Port Z Conversion Z	Port4 Z0 Imag 4 Z0	0.000 Ω
Fixture Simulator	De-Embedding	De-Embedding	OFF
Fixture Simulator	De-Embedding	Select Port	1
Fixture Simulator	De-Embedding	Select Type	None
Fixture Simulator	Diff Matching	Diff Matching	OFF
Fixture Simulator	Diff Matching	Select Bal Port	2
Fixture Simulator	Diff Matching	Select Circuit	Non
Fixture Simulator	Diff Matching	C	0.0000 F
Fixture Simulator	Diff Matching	G	0.0000 S

Fixture Simulator	Diff Matching	L	0.0000 H
Fixture Simulator	Diff Matching	R	0.0000 Ω
Fixture Simulator	Diff Z Conversion Z	Diff Z Conversion Z	OFF
Fixture Simulator	Diff Z Conversion Z	Port1 (bal) Real 1	100.00 Ω
Fixture Simulator	Diff Z Conversion Z	Port1 (bal) Imag 1	0.00 Ω
Fixture Simulator	Diff Z Conversion Z	Port2 (bal) Real 2	100.00 Ω
Fixture Simulator	Diff Z Conversion Z	Port2 (bal) Imag 2	0.00 Ω
Fixture Simulator	Diff Z Conversion Z	Port3 (bal) Real 3	100.00 Ω
Fixture Simulator	Diff Z Conversion Z	Port3 (bal) Imag 3	0.00 Ω

Fixture Simulator	Cmn Z Conversion Z	Cmn Z Conversion Z	OFF
Fixture Simulator	Cmn Z Conversion Z	Port1 (bal) Real 1	25.000 Ω
Fixture Simulator	Cmn Z Conversion Z	Port1 (bal) Imag 1	0.000 Ω
Fixture Simulator	Cmn Z Conversion Z	Port2 (bal) Real 2	25.000 Ω
Fixture Simulator	Cmn Z Conversion Z	Port2 (bal) Imag 2	0.000 Ω
Fixture Simulator	Cmn Z Conversion Z	Port3 (bal) Real 3	25.000 Ω
Fixture Simulator	Cmn Z Conversion Z	Port3 (bal) Imag 3	0.000 Ω
Fixture Simulator	De-Embedding S4P S4P	De-Embedding S4P S4P	OFF
Fixture Simulator	De-Embedding S5P S5P	Topology	A
Gating	Gating		OFF
Gating	Start		-10.000 ns

Gating	Stop		10.000 ns
Gating	Center		0.0000 s
Gating	Span		20.000 ns
Gating	Type		Bandpass
Gating	Shape		Normal
Transform	Transform		OFF
Transform	Start		-10.000 ns
Transform	Stop		10.000 ns
Transform	Center		0.0000 s
Transform	Span		20.000 ns
Transform	Type		Bandpass
Transform	Window		Normal
Transform	Window	Impulse Width	230.69 ps
Transform	Window	Kaiser Beta β	6.0000
Conversion	Conversion		OFF

Conversion	Function		Z:ReflectionZ
Limit Test	Limit Test		OFF
Limit Test	Limit Line		OFF
Limit Test	Limit Line Offsets	Stimulus Offset	0.0000 Hz
Limit Test	Limit Line Offsets	Amplitude Offset	0.0000 Hz
Limit Test	Fail Sign		ON
Ripple Limit	Ripple Limit Test		OFF
Ripple Limit	Ripple Limit		OFF
Ripple Limit	Ripple Value		OFF
Ripple Limit	Ripple Band		1
Ripple Limit	Fail Sign		ON
Bandwidth Limit	BW Test		OFF
Bandwidth Limit	BW Display		OFF

Bandwidth Limit	BW Marker	OFF
Bandwidth Limit	N dB PointsN dB	3.0000 dB
Bandwidth Limit	Min Bandwidth	10,000 kHz
Bandwidth Limit	Max Bandwidth	300.00 kHz
Bandwidth Limit	Fail Sign	ON

Avg Factor	16
Averaging	OFF
Avg Trigger	OFF
Smo Aperture	1.5000%
Smoothing	OFF
IF BandwidthIF	70 kHz

Correction			OFF
Calibrate	Response (Open)	Select Port	1
Calibrate	Response (Short)	Select Port	1
Calibrate	Response (Thru)	Select Ports	2-1 (S21)
Calibrate	Enhanced Response	Select Ports	2-1(S21 S1
Calibrate	1-Port Cal1	Select Port	1
Calibrate	2-Port Cal2	Select Ports	1-2
Calibrate	3-Port Cal3	Select Ports	1-2-3
Calibrate	2-Port TRL Cal2 TRL	Select Ports	1-2
Calibrate	3-Port TRL Cal3 TRL	Select Ports	1-2-3
	Adapter		

Calibrate	Removal	Select Ports	1
Calibrate	Adapter Removal	Cal Kit	85033E
ECal	Enhanced Response		2-1(S21 S1
ECal	Isolation		OFF
ECal	Characterization		Factory
ECal	Orientation		Auto
Property			OFF
Cal Kit			85033E
Modify Cal Kit	Define STDs		Define STD
Modify Cal Kit	Specify CLSs	Sub Class	Sub Class 1 1
Modify Cal Kit	TRL ReflectTRL		Short
Modify Cal Kit	TRL OptionTRL	Impedance	System
Modify Cal Kit	TRL OptionTRL	Reference Plane	THRU
Modify Cal Kit	Label Kit		85033E

Port Extensions	Extensions			OFF
Port Extensions	Auto Port Extension	Select Ports		1,2,3,4
Port Extensions	Auto Port Extension	Method		Current Spa
Port Extensions	Auto Port Extension	Method	User Span Start	9.0000 kHz 230/280/43
Port Extensions				100.00 kHz 235/285/43
Port Extensions	Auto Port Extension	Method	User Span Stop	3.0000 GHz 230/235/43
Port Extensions				8.5000 GHz 280/285/48
Port Extensions	Auto Port Extension	Adjust Mismatch		OFF
Port Extensions	Auto Port Extension	Include Loss		OFF
Port Extensions	Extension Port1 1			0.0000 s

Port Extensions	Extension Port2 2		0.0000 s
Port Extensions	Extension Port3 3		0.0000 s
Port Extensions	Extension Port4 4		0.0000 s
Port Extensions	Loss	Select Port	1
Port Extensions	Loss	Loss1 1	OFF
Port Extensions	Loss	Loss1 1	0.0000 dBm
Port Extensions	Loss	Freq1 1	1.0000 GHz
Port Extensions	Loss	Loss2 2	OFF
Port Extensions	Loss	Loss2 2	0.0000 dBm
Port Extensions	Loss	Freq2 2	1.0000 GHz

Port Extensions	Loss	Loss at DC	0.0000 dBm
Velocity Factor			1.0000
Set ZO ZO			50,000 Ω
Power Calibration	Select Port		1
Power Calibration	Correction		OFF
Power Calibration	Use Sensor		A
Power Calibration	Num of Readings		1
Power Calibration	Tolerance		5 dB
Power Calibration	Loss Compen	Compensation	OFF
Power Calibration	Sensor A Settings A	Ref Cal Factor	100.00%
Power Calibration	Sensor B Settings B	Ref Cal Factor	100.00%

Receiver Calibration	Select Port			1
Receiver Calibration	Correction			OFF
Receiver Calibration	Source Port			1
Mixer/Converter Calibration /	Scalar Cal (Manual)	Select Ports		2-1 (fwd)2-1
Mixer/Converter Calibration /	Scalar Cal (Manual)	Power Meter	Use Sensor	A
Mixer/Converter Calibration /	Scalar Cal (ECal)	Select Ports		2-1 (fwd)2-1
Mixer/Converter Calibration /	Scalar Cal (ECal)	Power Meter	Use Sensor	A
Cal Trig Source				Internal

	4.25005 GHz 230/235/430/435 8.5000 GHz 280/285/480/485

Allocate Channels	
Number of Traces	1
Allocate Traces	
Display	Data
Data Math	OFF
Title Label	OFF
Graticule Label	ON
Invert Color	OFF
Frequency	ON
Update	ON

	Log Mag

Echo Window	OFF

Discrete		OFF
Couple		ON
Marker Table		OFF
Statistics		OFF
Annotation Options	Marker Info X PosX	1 %
Annotation Options	Marker Info Y PosY	1 %
Annotation Options	Align	ON
Annotation Options	Active Only	ON

Peak		Search Peak
Peak	Peak Excursion	3.0000 dB
Peak	Peak Polarity	Positive
Target		Search Target
Target	Target Value	0.0000 dB“Target”
Target	Target Transition	Both“Target”
Tracking		OFF
Search Range	Search Range	OFF
Search Range	Start	0.0000 Hz
Search Range	Stop	0.0000 Hz
Search Range	Couple	ON
Bandwidth		OFF
Bandwidth Value		-3.0000 dB

Marker1 1	1
Ref Marker Mode	OFF

		S11
		Sss11 “Fixture Simulator”“ON” “Topology”“SE-Bal”- “Analysis”“BalUn” - “ON”
		Sdd11 “Fixture Simulator”“ON” “Topology”“Bal-Bal”- “Analysis”“BalUn” - “ON”
		Sss11 “Fixture Simulator”“ON” “Topology”“SE-SE-Bal”-- “Analysis”“BalUn” - “ON”
Absolute		A(1)
Aux Input 1Aux 1	Range	10 V
Aux Input 1Aux 1	Sweep Port	1
Aux Input 2Aux	Range	10 V

2		
Aux Input 2Aux 2	Sweep Port	1

Save Type		State & Cal
Channel/Trace/		Disp Only
Save SnP SnP	SnP FormatSnP	Auto

Divisions	10
Scale/Div/	10.000 dB/div
Reference Position	5 Div
Reference Value	0.0000 dB
Electrical Delay	0.0000 s
Phase Offset	0.0000 °

	2.9997 GHz 230/235/430/435 8.4997 GHz 280/285/480/485

	100.00 kHz

	3.0000 GHz 230/235/430/435 8.5000 GHz 280/285/480/485

Power	Power		0 dBm
Power	Port Couple		ON
Power	Port Power	Port 1 Power 1	0 dBm
Power	Port Power	Port 2 Power 2	0 dBm
Power	Port Power	Port 3 Power 3	0 dBm
Power	Port Power	Port 4 Power 4	0 dBm
Power	Slop [xx dB/GHz] [xx dB/GHz]		
Power	Slop [ON/OFF] [/]		OFF
Power	CW FreqCW		100.00 kHz
Power	RF OutRF		ON
Sweep Time			AUTO
Sweep Delay			0.0000 s
Sweep Mode			Stepped
Points			201

Sweep Type)			Lin Freq
Edit Segment Table	Freq Mode		Start/Stop/
Edit Segment Table	List IFBW IFBW		OFF
Edit Segment Table	List Power		OFF
Edit Segment Table	List Delay		OFF
Edit Segment Table	List Sweep Mode		OFF
Edit Segment Table	List Time		OFF
Segment Display			Order Base
Frequency Offset	Frequency Offset		OFF
Frequency Offset	Port 1 1	Multiplier	1.0000
Frequency Offset	Port 1 1	Divisor	1.0000
Frequency Offset	Port 1 1	Offset	0.0000 Hz
Frequency Offset	Port 1 1 Port 1 1	Start	9.00 KHz

Frequency Offset	Port 1 1 Port 1 1	Stop	8.5000 GHz
Frequency Offset	Port 2 2	Multiplier	1.0000
Frequency Offset	Port 2 2	Divisor	1.0000
Frequency Offset	Port 2 2	Offset	0.0000 Hz
Frequency Offset	Port 2 2 Port 2 2	Start	9.00 kHz
Frequency Offset	Port 2 2 Port 2 2	Stop	8.5000 GHz
Frequency Offset	Port 3 3	Offset	0.0000 Hz
Frequency Offset	Port 3 3 Port 3 3	Start	9300 kHz
Frequency Offset	Port 3 3 Port 3 3	Stop	8.5000 GHz
Frequency Offset	Port 4 4	Offset	0.0000 Hz

Frequency Offset	Port 4 4 Port 4 4	Start	9.00 kHz
Frequency Offset	Port 4 4 Port 4 4	Stop	8.5000 GHz
	External Source	LO FrequencyLO	OFF
	External Source	Multiplier	0.0000
Frequency Offset	External Source	Divisor	1.0000
Frequency Offset	External Source	Offset	0.0000 Hz
Frequency Offset	External Source	Start	0.0000 Hz
Frequency Offset	External Source	Stop	0.0000 Hz
Frequency Offset	External Source	Control	OFF
Frequency Offset	External Source	Power	-10 dBm
Frequency Offset	External Source	Slope [xxdB/GHz] [xxdB/GHz]	0 dB/GHz

Frequency Offset	External Source	Slope [ON/OFF] [/]	OFF
Frequency Offset	X-AxisX		Normal “Frequency Offset” “OFF”
Frequency Offset	X-AxisX		Response “Frequency Offset” “ON”

Invert Image		
Multiport Test Set Setup	Test Set 1 1	Select Test Set
Multiport Test Set Setup	Test Set 1 1	Control
Multiport Test Set Setup	Test Set 1 1	Property
Multiport Test Set Setup	Test Set 1 1	Port1 1
Multiport Test Set Setup	Test Set 1 1	Port2 2
Multiport Test Set Setup	Test Set 1 1	Port3 3
Multiport Test Set Setup	Test Set 1 1	Port4 4

Multiport Test Set Setup	Test Set 1 1	Port5 5	
Multiport Test Set Setup	Test Set 1 1	Port6 6	
Multiport Test Set Setup	Test Set 1 1	Port7 7	
Multiport Test Set Setup	Test Set 1 1	Control Lines	Line 1 1
Multiport Test Set Setup	Test Set 1 1	Control Lines	...
Multiport Test Set Setup	Test Set 1 1	Control Lines	Line 8 8
Multiport Test Set Setup	Test Set 2 2	Select Test Set	
Multiport Test Set Setup	Test Set 2 2	Control	

Multiport Test Set Setup	Test Set 2 2	Property
Multiport Test Set Setup	Test Set 2 2	Port1 1
Multiport Test Set Setup	Test Set 2 2	Port2 2
Multiport Test Set Setup	Test Set 2 2	port3 3
Multiport Test Set Setup	Test Set 2 2	Port4 4
Multiport Test Set Setup	Test Set 2 2	Port5 5
Multiport Test Set Setup	Test Set 2 2	Port6 6
Multiport Test Set Setup	Test Set 2 2	Port7 7

Multiport Test Set Setup	Test Set 2 2	Control Lines	Line 1 1
Multiport Test Set Setup	Test Set 2 2	Control Lines	...
Multiport Test Set Setup	Test Set 2 2	Control Lines	Line 8 8
Misc Setup	Beeper	Beep Complete	
Misc Setup	Beeper	Beep Warning	
Misc Setup	GPIB SetupGPIB	Talker/Listener Address/	
Misc Setup	GPIB SetupGPIB	Power Meter Address	
Misc Setup	GPIB SetupGPIB	Signal Generator Address	Address
Misc Setup	GPIB SetupGPIB	Signal Generator Address	Switching Time

Misc Setup	GPIB SetupGPIB	Signal Generator Address	Custom Commands	Pres
Misc Setup	GPIB SetupGPIB	Signal Generator Address	Custom Commands	Turr ON
Misc Setup	GPIB SetupGPIB	Signal Generator Address	Custom Commands	Set Freq
Misc Setup	GPIB SetupGPIB	Signal Generator Address	Custom Commands	Set Leve
Misc Setup	Network Setup	Telnet ServerTelnet		
Misc Setup	Network Setup	SICL-LAN ServerSICL-LAN		
Misc Setup	Network Setup	SICN-LAN AddressSICN-LAN		
Misc Setup	Network Setup	Web ServerWeb		
Misc Setup	Network Setup	MAC AddressMAC		
Misc Setup	Color Setup	Show Clock		

Misc Setup	Key Lock	Front Panel & Keyboard Lock	
Misc Setup	Key Lock	Touch Screen & Mouse Lock	
Misc Setup	Color Setup	Normal	Data Trace 1 1
Misc Setup	Color Setup	Normal	Data Trace 2 2
Misc Setup	Color Setup	Normal	Data Trace 3 3
Misc Setup	Color Setup	Normal	Data Trace 4 4
Misc Setup	Color Setup	Normal	Data Trace 5 5

Misc Setup	Color Setup	Normal	Data Trace 6 6
Misc Setup	Color Setup	Normal	Data Trace 7 7
Misc Setup	Color Setup	Normal	Data Trace 8 8
Misc Setup	Color Setup	Normal	Data Trace 9 9
Misc Setup	Color Setup	Normal	Mem Trace 1 1
Misc Setup	Color Setup	Normal	Mem Trace 2 2
Misc Setup	Color Setup	Normal	Mem Trace 3 3

Misc Setup	Color Setup	Normal	Mem Trace 4 4
Misc Setup	Color Setup	Normal	Mem Trace 5 5
Misc Setup	Color Setup	Normal	Mem Trace 6 6
Misc Setup	Color Setup	Normal	Mem Trace 7 7
Misc Setup	Color Setup	Normal	Mem Trace 8 8
Misc Setup	Color Setup	Normal	Mem Trace 9 9

Misc Setup	Color Setup	Normal	Graticule Main
Misc Setup	Color Setup	Normal	Graticule Sub
Misc Setup	Color Setup	Normal	Limit Fail
Misc Setup	Color Setup	Normal	Limit Line
Misc Setup	Color Setup	Normal	Background
Misc Setup	Channel/Trace Setup/		
Misc Setup	Preset Setup	State	

Misc Setup	Preset Setup	Confirm		
Backlight				
Service Menu	System Correction			
Service Menu	External Test Set			
Service Menu	Init Src Ctrl			
Service Menu	Init Src Port			
Service Menu	Security Level			

Continuous	Continuous (Ch1) 1 Hold (Other Ch)
Trigger Source	Internal
Trigger Event	On Sweep
Ext Trig Input	Negative Edge
Ext Trig Delay	0.0000 s
Low Latency	OFF
Ext Trig Output	OFF
Polarity	Positive Pulse
Position	After Point

Agilent E5071C

Agilent Technology

-
-
-

(|)	• •
SVC	
/ 241	

“Port N receiver overload” N N 221 224 “Port N receiver overload” N N	• •
	• DUT • DUT • • • •
	•
	• **“System”> “Key Lock”> “Front Panel & Keyboard Lock”**
	• “ System”>“ **Key Lock**”>“ **Touch Screen & Mouse Lock**” •

	• “ System”>“ Key Lock”>“ Touch Screen & Mouse Lock”
	•
	•
	•
	•
	•
	•
/	• .

120 121	• • • • •
/ GPIB	• GPIB • GPIB • GPIB GPIB • GPIB

CD-R Agilent Technologies

PC

LAN PC E5071C D: D: PC PC E5071C

GPIB :MMEM:TRAN E5071C D: PC

D:

-
-

Agilent Technologies

E5071C

Agilent Technologies

- LCD
- /
- LCD

LCD

-
-
  LCD

LCD

/

E5071C N RF

-
-
-
-

LCD

LCD/

/

-
-
-

E5071C

1.

2. Agilent 0960-2562

3.

e5071c190

1.

2.

3.

4. E5071C

E5071C 8 LED “U”

1. E5071C
2. “ **Macro Setup**”
3. “ **Load Project...**”...
4. **RestoreSysCorFile.vba** **D:\Agilent\Service** “ **Open**”
5. “ **Macro Run**”
6. RestoreSysCorFile
7. “ **OK**”

- E5071C

Agilent

(HDD)	5 20,000 LAN D	3
CPU	50,000	5
	50,000 E5071C	5
	50,000	5
LCD	50,000 LCD LCD	5
		5

E5071C Windows

http://www.agilent.com/find/ena_support

-
-
- ECal

http://www.agilent.com/find/na_accessories/

85033E		3.5 mm	DC 9 GHz
85052D		3.5 mm	DC 26.5 GHz
85032F		N 50 Ω	DC 9 GHz
85036B		N 75 Ω	DC 3 GHz
85036E		N 75 Ω	DC 3 GHz
85031B		APC-7	DC 6 GHz
85050C		APC-7	DC 18 GHz
85050D		APC-7	DC 18 GHz
85052C		3.5 mm	DC 26.5 GHz

ECal

85091C	2	APC-7	300 kHz 9 GHz
85092C	2	N 50 Ω	300 kHz 9 GHz
85093C	2	3.5 mm	300 kHz 9 GHz
85096C	2	N 75 Ω	300 kHz 3 GHz
85098C	2	7-16	300 kHz 7.5 GHz
85099C	2	F	300 kHz 3 GHz
N4431B	4	3.5 mmN 50 Ω	300 kHz 13.5 GHz

/

-
-
-

Agilent Technologies
http://www.agilent.com/find/na_accessories/

N6314A 50 Ω N RF 300 kHz 9 GHz

RF 610 mm N

N6315A 50 Ω N RF 300 kHz 9 GHz

RF 610 mm N

11853A 50 Ω N

11853A

Agilent	
1250-1472	N -
1250-1475	N -
11511A	N
11512A	N

11878A N 3.5 mm

11878A

Agilent	
1250-1744	3.5 mm N 50 Ω
1250-1743	3.5 mm N 50 Ω
1250-1745	3.5 mm N 50 Ω
1250-1750	3.5 mm N 50 Ω

11854A 50 Ω BNC

11854A

Agilent	
1250-0929	BNC

1250-1473	BNC N
1250-1474	BNC N
1250-1476	BNC N
1250-1477	BNC N

11852B

004	N 50 Ω 75 Ω

- USB/GPIB
- GPIB
- Agilent IO

82357A USB/GPIB

E5071C 82357A GPIB USB PC E5071C
http://www.agilent.com/find/gpib

GPIB

GPIB

10833A	1.0 m (3.3 ft)
10833B	2.0 m (6.6 ft)
10833C	3.0 m (9.9 ft)
10833D	0.5 m (1.6 ft)

Agilent IO

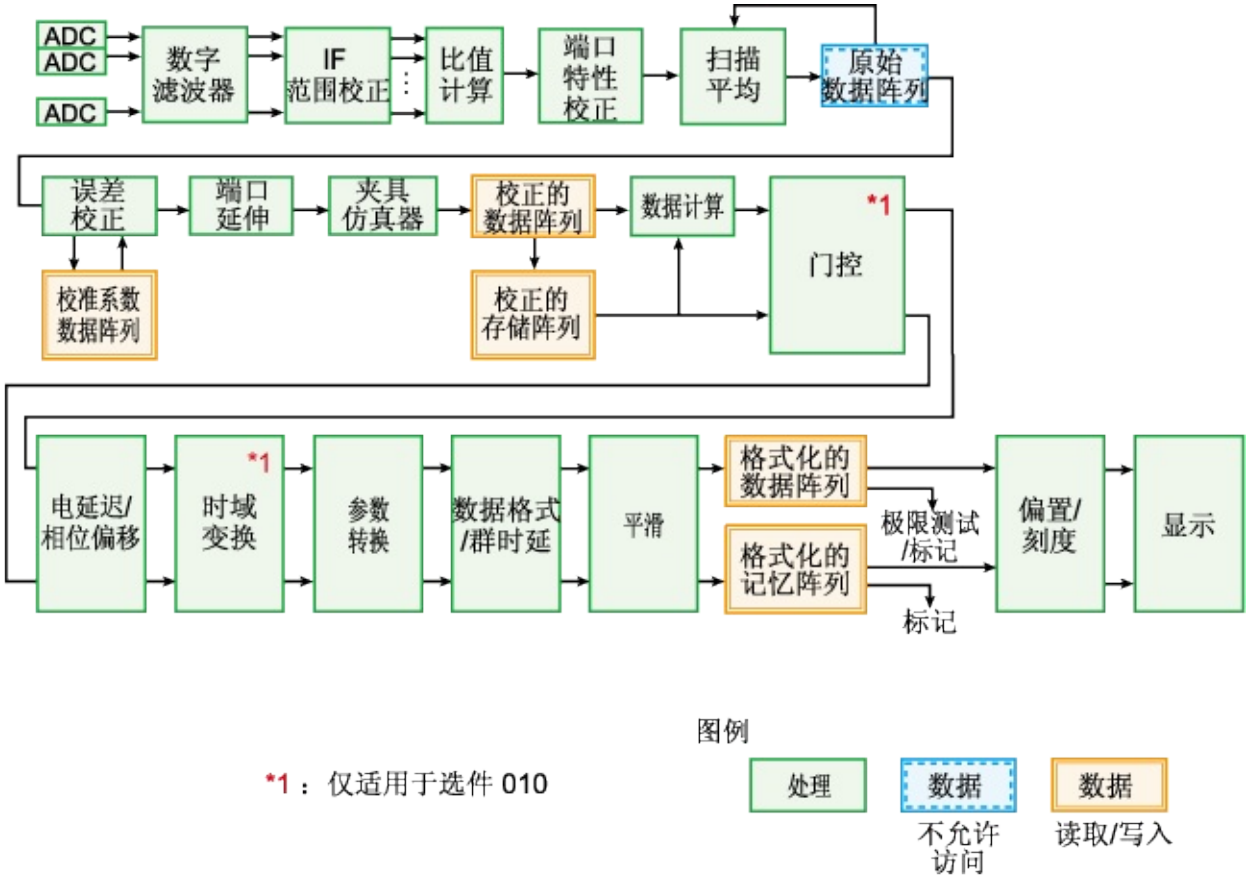
Agilent IO

IO http://www.agilent.com/find/iolib IO

- Agilent E5071C IO
  http://www.agilent.com/find/ena_support/

-
-

E5071C



图例

*1 ： 仅适用于选件 010

处理

数据
不允许
访问

数据
读取/写入

e5071c465

ADC

ADC IF R $_1R_2$.... R_n T $_1T_2$.... T_n ADC

(DFT) IF IF IF DFT 10 Hz ~ 100 kHz
IF

IF

008

008

NN=2 4 4 × N S /

/

12 /

-

“ **Display**”> **“Data” -> “Mem”**

/

/

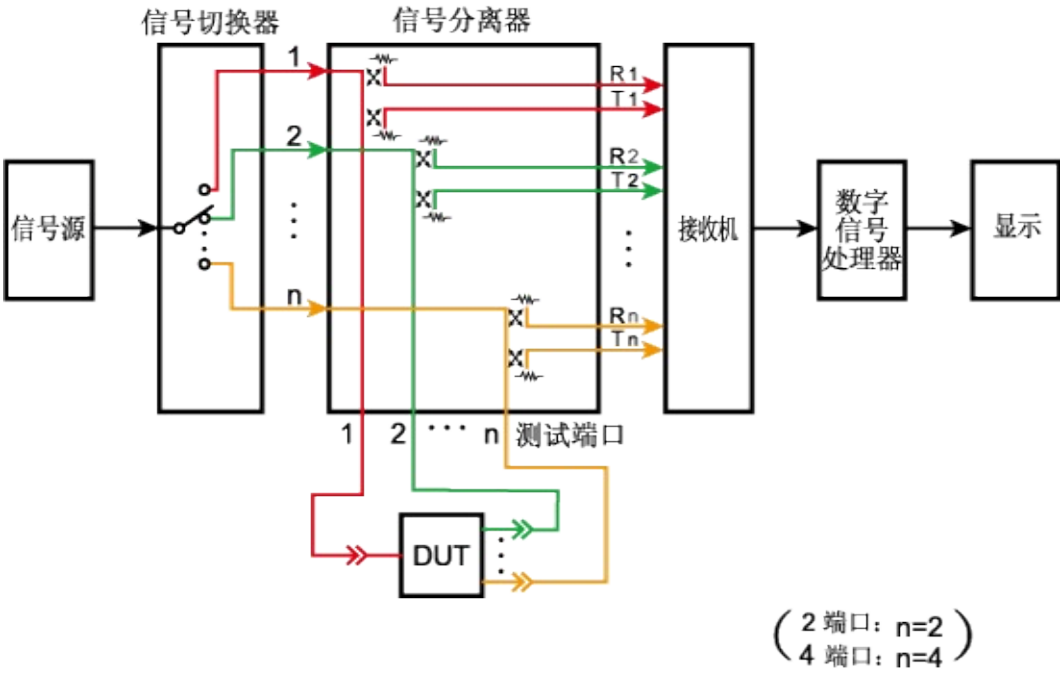
/

/

/

DUTE5071C

E5071C



e5071c241

RF

RF ALC -15 +10 dBm

RF

DUT (R) (T) DUT (T)

IF ADC

E5071C 8753ES

Agilent E5071C Agilent 8753ES Agilent “”
http://www.agilent.com/find/ena/

-
-

Agilent 8753ES Agilent E5071C

8753ES 1 2 3 4 3 4 1 2 1 3
2 4 1 2 1 2 E5071C
_____ 36 16

E5071C S 8753ES ABRA/RB/RA/B

8753ES E5071C 230280235 285 430480
435 4858753ES 75 Ω 075 E5071C

8753ES E5071C

	8753ES	E5071C
	• 30 kHz 3 GHz • 30 kHz 6 GHz 006	• 9 kHz 3 GHz 230430 • 9 kHz 8.5 GHz 280 480 • 100 kHz 3 GHz 235 435 • 100 kHz 8.5 GHz 285 485

8753ES E5071C

	8753ES	E5071C
	• - 85 dBm 10 dBm • - 85 dBm 8 dBm 014 075	• - 55 dBm 10 dBm
	• - 15 dBm 10 dBm	

	- 15 dBm 8 dBm 014 075 • - 25 dBm 0 dBm • - 35 dBm - 10 dBm • - 45 dBm - 20 dBm • - 55 dBm - 30 dBm • - 65 dBm - 40 dBm • - 75 dBm - 50 dBm • - 85 dBm - 60 dBm	

8753ES E5071C 8753ES 30
E5071C 201 8753ES GPIB (SCPI) E5071C

8753ES IF 6 kHz E5071C IF 500 kHz 8753ES

8753ES E5071C

8753ES 008 E5071C

8753ES E5071C

	8753ES	E5071C
7 mm	• 85031B	• 85031B • 85050C/D
3.5 mm	• 85033C/D/E	• 85033D/E • 85052D
N	• 50 Ω85032B/E/F • 75 Ω85036B/E	• 50 Ω85032B/F • 75 Ω85036B/E
2.4 mm	• 85056/D	N/A
TRL 3.5 mm	• 85052C	• 85052C

8753ES 15 2 12 TRL*/LRM* /

E5071C 20

ECal

8753ES E5071C ECal 8753ES 1 S11 S22
 2 E5071C 1 S11 S22 2 3 4

4 4

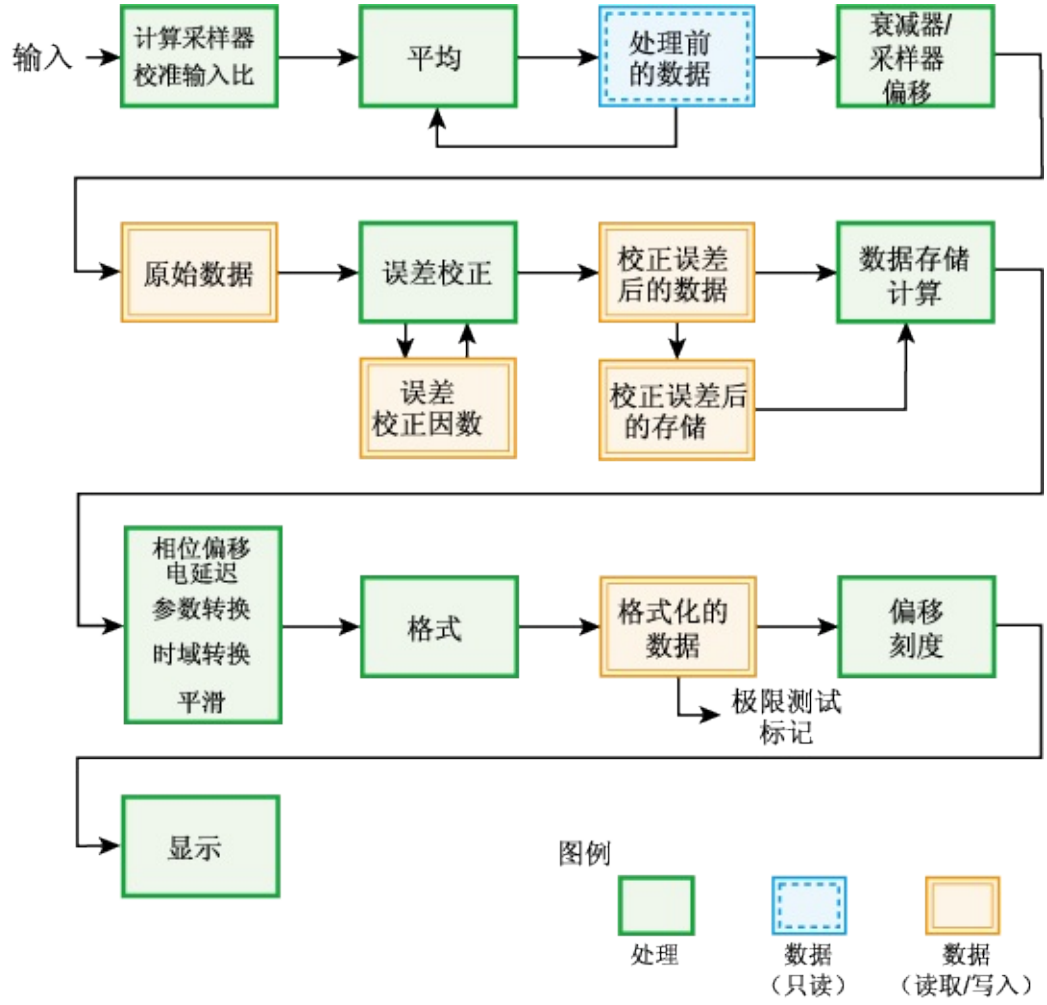
8753ES E5071C

8753ES 1 3 2 4
1 2

E5071C
E5071C E5071C

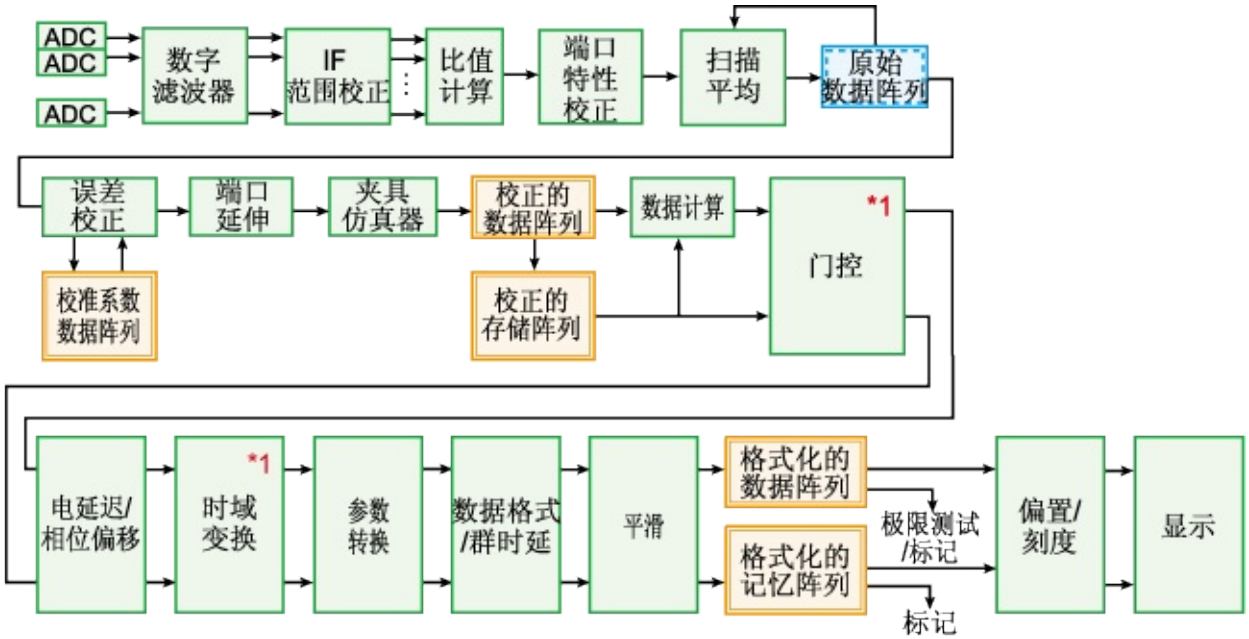
8753ES 8753ES E5071C E5071C 8753ES / E5071C /

8753ES



e5071c363

E5071C



图例

*1 ：仅适用于选件 010

e5071c465

/

8753ES E5071C

/

	8753ES	**E5071C**
	• • IEEE 32 • IEEE 64 • ASCII • PC-DOS 32	• IEEE 64 ASCII
/	• • • • • • •	• • • • • •
	• Pre-raw Take4 • • • •	

8753ES

E5071C E5071C

8753ES E5071C 8753ES 8753ES
E5071C

8753ES “ **Data / Memory**” / “ **Data - Memory**”
- E5071C“ **Data * Memory**” *
Memory” +
8753ES “”“”“”“”“”E5071C “”

8753ES E5071C 8753ES E5071C

8753ES 18 E5071C 100

8753ES // E5071C ///

8753ES E5071C E5071C -

/

8753ES GPIB E5071C USB
8753ES E5071C VBA 8753ES JPEG
E5071C Windows (BMP) (PNG)

8753ES E5071C VBA

/

8753ES 1/4 E5071C /

8753ES GPIB E5071C USB

GPIB

8753ES PC E5071C

LAN

8753ES LAN E5071C LAN

8753ES Take4 002E5071C 008

	8753ES	E5071C

	8753ES	E5071C

	8753ES	E5071C
	• • 7 mm85031B • 3.5 mm85033C/D/E • N (50 Ω)85032B/E/F • N (75 Ω)85036B/E • 2.4 mm85056A/D • TRL 3.5 mm85052C	• 7 mm85031B85050C/D • 3.5 mm85033D/E85052D • N (50 Ω)85032B/F • N (75 Ω)85036B/E • TRL 3.5 mm85052C 11
	• • • • S11 1 • S22 1 • 2 • TRL*/LRM* • •	• • • • 1 • 2 • TRL/LRM • • • • • • /

			• 3 430/435/480/485 • 4 430/435/480/485 • TRL/LRM • • • • • • • TRL • • /
		GPIB	2 4

	1 2 A B	
	/	
	/	
	• S11A • S11B • S11C • S22A • S22B • S22C • • •	• • • •

		• • • • TRL • TRL • TRL /	• TRL • TRL • TRL /
		• • • • •	
		• C0C1C2C3	
		•	L0L1L2L3

		• •	•
ECal		• • S11 1 • S22 1 • 2	• • • 1 • 2 • 3 430/435/480/485 • 4 430/435/480/485

8753ES		E5071C
	• • 18	• • 100
GPIB		
	• • / • / •	• / • •

		8753ES	E5071C
		•	•
		• / • -	• / • * • - • +
		• • • •	• • • • •
/			/
		•	• 36 • 16
1/ 2 2			
		10	4 30 2
/			

/ LCD	
/	

	8753ES	E5071C
		VBA
	“AUTO” 6	

8753ES	E5071C
?	
Delta	
/	/
/	/
/	/
• • • •	• • • •
•	• •
• • • Q •	• • • • Q •

			/
		/	/
CW			

		8753ES	E5071C
		GPIB	GPIB Telnet
		4	36
		3 4 1 2 1 2 1 2 /	
			16
		• S • A • B • R • A/R • B/R • A/B •	• S S • •
S		• • • 1/S	
		• • • • •	• • • • •

		• • SWR • •	• • • • SWR • •
		• 2	• 2 230/235/280/285 • 4 430/435/480/485
		• 30 kHz 3 GHz • 30 kHz 6 GHz 006	• 9 kHz 3 GHz 230/430 • 100 kHz 3 GHz 235/435 • 9 kHz 8.5 GHz 280/480 • 100 kHz 8.5 GHz 285/485
		• -85 +10 dBm • -85 +8 dBm 014/075	• -55 10 dBm
		• 50 Ω • 75 Ω 075	• 50 Ω
	/	/	

	/			
			/	
			10 dBm -15 dBm +10 dBm -85 dBm -60 dBm	
			3, 11, 21, 26, 51, 101, 201, 401, 801, 1601	2 1601 / 1 ch/4 Tr 20001
			• • • • • CW	• • • •
			1 3 2 4	16
			30	201
		GPIB(SCPI)		
			1 1601 1601	2 200001 20001
		/		
			24	

	IF	10 Hz30 Hz100 Hz300 Hz 1 KHz3 kHz3.7 kHz6 kHz	10 Hz15 Hz20 Hz30 Hz 40 Hz50 Hz70 Hz 100 Hz 150 Hz200 Hz300 Hz 400 Hz500 Hz700 Hz1 kHz 1.5 kHz2 kHz3 kHz 4 kHz 5 kHz7 kHz10 kHz 15 kHz 20 kHz30 kHz40 kHz50 kHz 70 kHz100 kHz150 kHz 200 kHz300 kHz400 kHz 500 kHz
		1 999	1 999
		0.05% 20%	
		0 10	-10 +10
		-360 +360	
/		1 3 2 4 1 2	
		•	•
		•	•
		•	•
		•	

	• • •	• • • •

		8753ES	E5071C
Take4			
			008
		002	008
GPIB		GPIB	
LAN			• 10 Base-T • 100 Base-TX
			• TCP/IP
			• Telnet • SICL-LAN
GPIB		/	
	GPIB	GPIB LO	GPIB
		GPIO	

8753ES	E5071C
1/4	
	/
• • • GPIB	• • USB

/	8753ES	E5071C
	• • IEEE 32 • IEEE 64 • ASCII • PC-DOS 32	• IEEE 32 • IEEE 64 • ASCII • PC-DOS 32
/	• • • • • •	• • • • • •
	• Pre-raw Take4 • •	

/	8753ES	E5071C
	• • • GPIB	• • USB
	LCD	• • LCD
/	• LCD • •	• • • • VBA
	• • ASCII	•
	• JPEG	• Windows • PNG
	• •	• • • •

8753ES	E5071C
• • • • B	• • • • •
B	• • • • • • • • • • • •

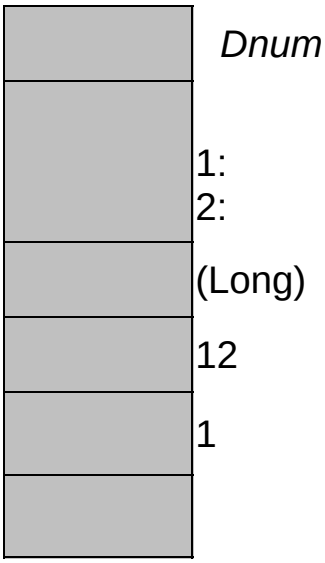
8753ES	E5071C
SCPI	
/	/
• • • •	

1-2	1 2
1-3	1 3
1-4	1 4
2-3	2 3
2-4	2 4
3-4	3 4

(*) “ **Done**”

Port 1 1	1
Port 2 2	2
Port 3 3	3
Port 4 4	4
2-1 (fwd)2-1	1 2
1-2 (rev)1-2	1 2
1, 2 (both)12	1 2
3-1 (fwd)3-1	1 3
1-3 (rev)1-3	1 3
1, 3 (both)13	1 3
4-1 (fwd)4-1	1 4
1-4 (rev)1-4	1 4
1, 4 (both)14	1 4
3-2 (fwd)3-2	2 3

2-3 (rev)2-3	2 3
2, 3 (both)23	2 3
4-2 (fwd)4-2	2 4
2-4 (rev)2-4	2 4
2, 4 (both)24	2 4
4-3 (fwd)4-3	3 4
3-4 (rev)3-4	3 4
3, 4 (both)34	3 4


Dnum
1:
2:
(Long)
12
1

	Ch
	(Long)
	136
	1

	Pt
	(Long)
	14
	1

- .
- _____
- .
- /__
- .
- .
- .

-
-
-