

Agilent E5070B/E5071B ENA Series RF Network Analyzers

Control an External Measurement Instrument

Second Edition



Agilent Technologies

No. 16000-95011

August 2002

Notices

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

This document contains proprietary information that is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced, or translated to another language without the prior written consent of Agilent Technologies.

Agilent Technologies Japan, Ltd.

Component Test PGU-Kobe

1-3-2, Murotani, Nishi-ku, Kobe, Hyogo, 651-2241 Japan

MS-DOS®, Windows®, Windows 98, Windows NT®, Visual C++®, Visual Basic®, VBA, Excel and PowerPoint® are U.S. registered trademarks of Microsoft Corporation.

Portions ©Copyright 1996, Microsoft Corporation. All rights reserved.

© Copyright Agilent Technologies Japan, Ltd. 2002

Sample Program

The customer shall have the personal, non-transferable rights to use, copy, or modify SAMPLE PROGRAMS in this manual for the customer's internal operations. The customer shall use the SAMPLE PROGRAMS solely and exclusively for their own purposes and shall not license, lease, market, or distribute the SAMPLE PROGRAMS or modification of any part thereof.

Agilent Technologies shall not be liable for the quality, performance, or behavior of the SAMPLE PROGRAMS. Agilent Technologies especially disclaims any responsibility for the operation of the SAMPLE PROGRAMS to be uninterrupted or error-free. The SAMPLE PROGRAMS are provided AS IS.

AGILENT TECHNOLOGIES DISCLAIMS ANY IMPLIED WARRANTY OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Agilent Technologies shall not be liable for any infringement of any patent, trademark, copyright, or other proprietary right by the SAMPLE PROGRAMS or their use. Agilent Technologies does not warrant that the SAMPLE PROGRAMS are free from infringements of such rights of third parties. However, Agilent Technologies will not knowingly infringe or deliver software that infringes the patent, trademark, copyright, or other proprietary right of a third party.

Control an External Measurement Instrument

This chapter explains how to control peripherals connected to the E5070B/E5071B with GPIB by using the software (VISA library) installed in the E5070B/E5071B.

Overview

The E5070B/E5071B macro function (E5070B/E5071B VBA) can be used not only to automate measurements but also to control external measurement instruments connected via GPIB cables by acting as a self-contained system controller.

The E5070B/E5071B macro function (E5070B/E5071B VBA) performs communications via the COM interface when controlling the E5070B/E5071B itself, but it communicates via VISA (Virtual Instrument Software Architecture) when controlling external measurement instruments.

To control peripherals connected to the E5070B/E5071B via GPIB, the following two preparations are required.

Preparations

1. Placing E5070B/E5071B in System Control Mode

When the E5070B/E5071B and its peripherals are controlled with the E5070B/E5071B macro function, this is done via the GPIB bus inside the E5070B/E5071B. Therefore, you need to set the GPIB system's control mode to the system controller mode. Then you need to set the GPIB address as the system controller. Follow these steps using the front panel to place the E5070B/E5071B in the system control mode.

Step 1. Place the E5070B/E5071B in the system controller mode.

- **[System] - GPIB Setup - GPIB Configuration (Sys Controller)**

Step 2. Set the GPIB address of the system controller. “xx” represents the address number.

- **[System] - GPIB Setup - System Controller Address (xx)**

Step 3. Reboot the E5070B/E5071B.

2. Importing Definition Files

To use the VISA library in the E5070B/E5071B macro (E5070B/E5071B VBA), you need to import two definition files into your project with the Visual Basic editor to define the VISA functions and perform other tasks. The definition files are stored on the sample programs disk under the following filenames (for information on importing modules, refer to “Saving a Module (Exporting)”).

- visa32.bas
 - vpptype.bas
-

Programming with VISA

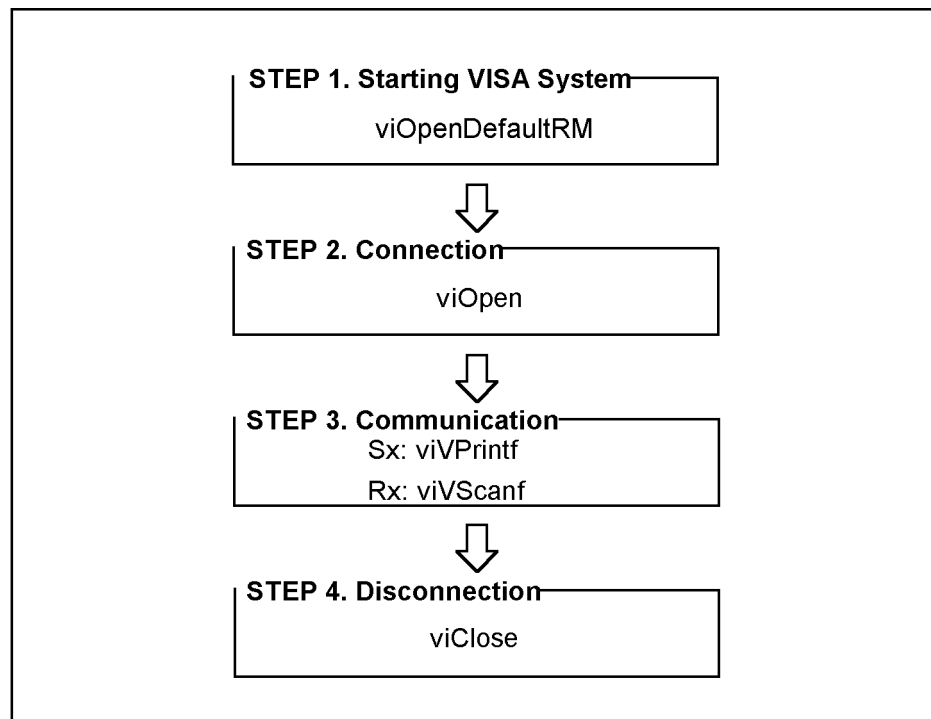
Figure 1 shows the flow of controlling the instrument with VISA. When developing a VISA program in the Visual Basic language, a special programming notice (in the readme text file listed below) must be reviewed.

For details on the use of the VISA library and the programming notice for using the VISA library with the E5070B/E5071B macro (E5070B/E5071B VBA), refer to the following files contained on the CD-ROM (Agilent part number: E5070-905xx).

- visa.hlp (on-line help for the VISA library)
- vbreadme.txt (notes on using the VISA library with VB)

Figure 1

Flow of instrument control with VISA



e4991ape033

STEP 1. Starting Up VISA System

The VISA system startup session is processed in Line 90 in Example 1-1. VISA's viOpenDefaultRM function initializes and starts up the VISA system. The viOpenDefaultRM function must be executed before other VISA functions are called, and the parameter of this function is startup information (Defrm in Example 1-1).

Syntax

viOpenDefaultRM(*param*)

Parameter

| | |
|-------------|------------------------------|
| | <i>(param)</i> |
| Description | Startup information (output) |
| Data type | Long integer type |

STEP 2. Connection

The connection session is handled in Line 130 in Example 1-1. VISA's viOpen function makes connection with the specified instrument. The viOpen function returns a value so that the VISA functions can apply it to the specified instrument. The parameters of this function are startup information (Defrm in Example 1-1), the address information of the specified instrument ("GPIB0::17::INSTR" in Example 1-1), access mode (0 in Example 1-1), timeout (0 in Example 1-1), and connection information (Equip in Example 1-1).

Syntax

viOpen(*param1,param2,param3,param4,param5*)

Parameters

| | |
|-------------|-----------------------------|
| | <i>(param1)</i> |
| Description | Startup information (input) |
| Data type | Long integer type |

| | |
|-------------|--|
| | <i>(param2)</i> |
| Description | Address information of the specified instrument (input) |
| Data type | Character string type |
| Syntax | GPIB[<i>board</i>] ^{*1} :: <i>primary address</i> ^{*2} ::INSTR |

*1. GPIB0 for the E5070B/E5071B.

*2. The GPIB address of the instrument controlled by the E5070B/E5071B.

| | |
|-------------|-----------------------|
| | <i>(param3)</i> |
| Description | Access mode (Enter 0) |

| | |
|-------------|-------------------|
| | <i>(param4)</i> |
| Description | Timeout (Enter 0) |

| | |
|-------------|---------------------------------|
| | <i>(param5)</i> |
| Description | Connection information (output) |
| Data type | Long integer type |

STEP 3. Communication

The communication session is conducted in Line 170 in Example 1-1. VISA's viVPrintf function sends a program message (GPIB command) to the specified instrument. The parameters of this function are connection information (Equip in Example 1-1), the program message ("*IDN?" in Example 1-1), and the variable to be formatted (0 in Example 1-1).

NOTE

To input/output GPIB commands, the viVPrintf function and the viVScanf function are mainly used, but other VISA functions are also available. For more information, refer to visa.hlp (online help for the VISA library).

Syntax

viVPrintf(*param1,param2,param3*)

Parameters

| | |
|-------------|--------------------------------|
| | <i>(param1)</i> |
| Description | Connection information (input) |
| Data type | Long integer type |

| | |
|-------------|---------------------------------------|
| | <i>(param2)</i> |
| Description | Program message (input) ^{*1} |
| Data type | Character string type |

^{*1}. When sending a program message of the GPIB command, a message terminator is required at the end of the message (Chr\$(10) in Example 1-1).

| | |
|-------------|--|
| | <i>(param3)</i> |
| Description | A variable to be formatted ^{*1} |
| Data type | Specified data type |

^{*1}. If not applicable, enter 0.

The receiving session is controlled in Line 210 in Example 1-1. VISA’s viVScanf function receives the result from the specified instrument and stores it in the output variable. The parameters of this function are connection information (Equip in Example 1-1), the format parameter for the output variable (%t in Example 1-1), and the output variable (Prod in Example 1-1).

Syntax

```
viVScanf(param1,param2,param3)
```

Parameters

| | |
|-------------|--------------------------------|
| | <i>(param1)</i> |
| Description | Connection information (input) |
| Data type | Long integer type |

| | |
|-------------|--|
| | <i>(param2)</i> |
| Description | Format parameter for the output variable |
| Data type | Character string type |

| | |
|-------------|--------------------------|
| | <i>(param3)</i> |
| Description | Output variable (output) |
| Data type | Character string type |

STEP 4. Disconnection

The disconnection session is handled in Line 280 in Example 1-1. VISA’s viClose function disconnects communication and terminates the VISA system. The parameter of this function is startup information (Defrm in Example 1-1).

Syntax

```
viClose(param)
```

Parameter

| | |
|-------------|-----------------------------|
| | <i>(param)</i> |
| Description | Startup information (input) |
| Data type | Long integer type |

Example Program to Read Out the Product Information of Peripheral (Instrument)

Here is a sample program to control instruments connected through GPIB using the E5070B/E5071B as the system controller. The sample program disk contains a sample program, named “ctrl_ext.vba“, that reads out the product information of external instrument connected via GPIB. This VBA program consists of the following modules.

| Object name | Module type | Content |
|--------------------|-----------------|---|
| mdlVisa | Standard module | Reads out the product information of external instrument. |
| Module1 Module2 | Standard module | Two definition files to use VISA library |

NOTE

When you control peripherals from E5070B/E5071B VBA, use the GPIB commands provided for the instrument to communicate over VISA. On the other hand, when you control the E5070B/E5071B itself from E5070B/E5071B VBA, use the COM objects provided for the E5070B/E5071B to communicate.

| | |
|------------------|---|
| Lines 90 to 100 | Initializes and starts up the VISA system and outputs the startup information to the Defrm variable. During this process, if an error occurs, the program goes to the error handling routine (Lines 320 to 360). |
| Lines 130 to 140 | Establishes the connection to the external instrument (GPIB address: 17) connected via GPIB and outputs the connection information to the Equip variable. During this process, if an error occurs, the program goes to the error handling routine (Lines 320 to 360). |
| Lines 170 to 180 | Queries the product information of the external instrument connected via GPIB through VISA. During this process, if an error occurs, the program goes to the error handling routine (Lines 320 to 360). |
| Lines 210 to 250 | Retrieves the product information through VISA and outputs it into the Prod variable. Displays the read-out result in the message box. During this process, if an error occurs, the program goes to the error handling routine (Lines 320 to 360). |
| Line 280 | Breaks the communication and terminates the VISA system. |
| Lines 320 to 360 | If an error occurs in a VISA function, displays the detail of the error and terminates the program. |

Example 1-1**Sample program to read out the product information**

```
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
```
