Agilent E5070B/E5071B ENA Series RF Network Analyzers

Executing Power Calibration

First Edition

Agilent Technologies

No. 16000-95023

August 2002

Notices

Agilent Technologies Japan, Ltd.

Component Test PGU-Kobe

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

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.

Executing Power Calibration

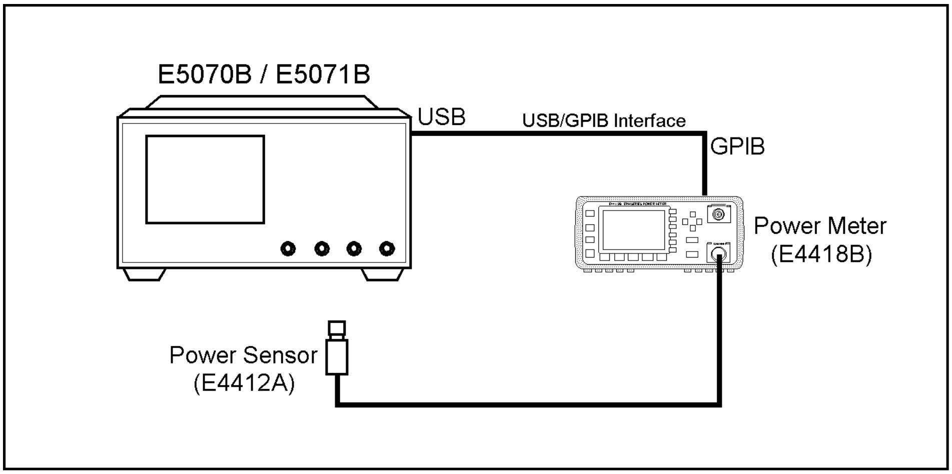
Example 1 shows a sample program (VBA program) to execute power calibration using the E4418B power meter and the E4412A power sensor. You can find the source file of this program, named pow_cal.vba, on the sample program disk. This VBA program consists of the following modules:

Object name	Module type	Description
mdlPowCal	Standard module	Executes power calibration.
Module1 Module2	Standard module	The definition files when using the VISA library.

Program overview

The program connects the E5070B/E5071B and the E4418B (GPIB address: 13) through the USB/GPIB interface as shown in Figure 1 and then executes the zero adjustment and calibration of the power sensor (E4412A) connected to the power meter (E4418B) as necessary. Then, it executes the power calibration of the E5070B/E5071B and saves the obtained power calibration data array into a file.

Figure 1 **Connection between E5070B/E5071B and power meter**



e5070bve043

Program description

When you run this VBA program, reset is performed, the GPIB address of the power meter to be controlled and the measurement conditions of the E5070B/E5071B are set, and a message “Do you perform zeroing and calibrating the power meter on channel A?” is displayed. If you want to execute the zero adjustment and the calibration of the power sensor, click the **Yes** button; to skip them, click the **No** button.

If you want to execute the zero adjustment and calibration of the power sensor, follow the displayed messages to connect the power sensor connected to the A channel of the power meter to the POWER RF port of the power meter and click the **OK** button. When the zero adjustment and calibration of the power sensor is complete, a message “Zeroing and calibrating the power sensor is complete.” is displayed.

When a message that prompts you to connect the power sensor to port 1 of the E5070B/E5071B is displayed, make the connection and then click the **OK** button. The power calibration data measurement is executed immediately and the obtained power calibration data array is saved in a file named "CORR_DATA."

The power calibration execution program (object name: mdlPowCal) is described in detail below. Line numbers are added for description purpose only, and do not appear in the actual program source code.

Lines 100 to 150 Assigns the sweep type (power sweep), the number of points (41), the power range (-20 to +12 dBm), the sweep start value (-20 dBm), the sweep stop value (-10 dBm), and the fixed frequency (1 GHz) to the Swp_type, Nop, Pow_rang, Start_p, Stop_p, and Cw_freq variables, respectively.

Lines 160 to 170 Assigns the number of power calibration data measurements for each measurement point (4) and the limit value of the power calibration data array (10 dBm) into the Num_avg and Limit variables, respectively.

Line 200 Returns the E5070B/E5071B to the preset state.

Line 230 Sets the power meter's GPIB address (13) in the E5070B/E5071B.

Lines 260 to 310 For channel 1, assigns the sweep type to the Swp_type variable, the number of points to the Nop variable, the power range to the Pow_rang variable, the sweep start value to the Start_p variable, the sweep stop value to the Stop_p variable, and the fixed frequency to the Cw_freq variable.

Lines 340 to 370 Displays a message asking you whether to execute the zero adjustment and calibration of the power sensor. If the **Yes** button is clicked, calls the procedure: Control_PowerMeter (Line 900 to Line 1340). For information on the Control_PowerMeter procedure, see the description later. On the other hand, if the **No** button is clicked, skips the zero adjustment and calibration of the power sensor.

Lines 410 to 420 Clears the error queue. Then, prompts you to connect the power sensor to port 1 of the E5070B/E5071B and waits for clicking the **OK** button after the connection.

Lines 450 to 470 For port 1 of channel 1, sets the number of power calibration data measurements for each measurement point to the Num_avg variable and then starts the measurement of the power calibration data and waits for the completion of the measurement.

Line 500 Reads out an error that has occurred in the E5070B/E5071B during the measurement of the power calibration data and sets it in the Err variable.

Lines 510 to 590 If no error has occurred, reads out the power calibration data array and sets it in the Corr_data variable. In addition, uses the Limit_Test function to check whether the read out power calibration data array exceeds the specified limit value. If the limit value is exceeded, the return value of the Limit_Test function, False, is returned. Then, displays a message asking you whether to perform the power calibration again. Click the **Yes** button to return to the start of the power calibration data measurement. Click the **No** button to terminate the program. For information on the Limit_Test function (Line 1360 to Line 1530), see the description later.

Lines 610 to 660 If an error occurs, displays an error message and a message asking you whether to execute the power calibration again. Click the **Yes** button to return to the start of the power calibration data measurement. Click the **No** button to terminate the program.

Lines 690 to 790 Writes the read out power calibration data array into a file named "CORR_DATA." Then, displays a message that notifies you that saving into a file is successfully complete.

Procedure: Control_PowerMeter (lines 900 to 1340).

Lines 990 to 1000 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 1260 to 1300).

Lines 1030 to 1040 Establishes the connection to the power meter in use (GPIB address: 13) and outputs the connection information to the E4418 variable. During this process, if an error occurs, the program goes to the error handling routine (Lines 1260 to 1300).

Lines 1070 to 1090 Returns the power meter to the preset state through VISA and clears the status byte register and the standard event status register. Then, enables the standard event status register's bit 0.

Line 1100 Prompts you to connect the power sensor to the POWER REF port of the power meter and waits for clicking the **OK** button after the connection.

Lines 1110 to 1120 Executes the zero adjustment and calibration of the power sensor through VISA. Makes the setting so that 1 is set to bit 0 of the standard event status register when all pending operations are complete.

Lines 1130 to 1190 Retrieves the value of the status byte register through VISA and sets it into the Rgst variable. Sets the AND of the read-out value and 32 (the value in which only bit 5 is 1) into the Rslt variable and displays a message that notifies you that the zero adjustment and calibration of the power sensor is complete when Rslt becomes 1 (when the zero adjustment and calibration of the power sensor is complete).

Line 1220 Breaks the communication and terminates the VISA system.

Lines 1260 to 1300 If an error occurs in a VISA function, displays the detail of the error and terminates the program.

Function: Limit_Test (Lines 1360 to 1530)

Lines 1420 to 1470 If the absolute value of the read out power calibration data array exceeds the specified limit value, turns off the power calibration function and returns the value of False. Otherwise, returns the value of True.

Example 1

Measurement of power calibration (object name: 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 E5070/71B
 200|    SCPI.SYSTem.PRESet
 210|
 220|    '''Setting GPIB address of the power meter to E5070/71B
 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
```

```
Meas_Start:
  '''Connecting the power sensor to the port 1 in the E5070/71B
  SCPI.IEEE4882.CLS
  MsgBox "Set the power sensor connected to the port 1 in the E5070/71B.", vbOKOnly, "Power meter calibration"

  '''Performing power calibration measurement
  SCPI.Source(1).POWer.PORT(1).CORRection.COLLect.AVERage.count = Num_avg
  SCPI.Source(1).POWer.PORT(1).CORRection.COLLect.ACQuire = "ASEN"
  Dmy = SCPI.IEEE4882.OPC

  '''Error handling at power meter calibration
  Err = SCPI.SYSTem.Error
  If Err(0) = 0 Then
    Corr_data = SCPI.Source(1).POWer.PORT(1).CORRection.DATA
    Verifier = Limit_Test(Nop, Limit, Corr_data)
    If Verifier = False Then
     Buff = MsgBox("Do you perform the power meter calibration measurement again?", vbYesNo, "Power meter calibration")
      If Buff = vbYes Then GoTo Meas_Start
      If Buff = vbNo Then GoTo Prog_Stop
    End If
   MsgBox "Power meter calibration measurement is complete.", vbOKOnly, "Power meter calibration"

  Else
    MsgBox "Error: " & Err(1)
   Buff = MsgBox("Do you perform the power meter calibration measurement again?", vbYesNo, "Power meter calibration")
    If Buff = vbYes Then GoTo Meas_Start
    If Buff = vbNo Then GoTo Prog_Stop
  End If

  '''Installing the corrected data to a file
  FileNo = FreeFile
  File = "CORR_DATA"

  Open File For Output As FileNo

  For I = 0 To Nop - 1
    Write #FileNo, Val(Corr_data(I))
  Next I
  Close #FileNo

  MsgBox "Installing the corrected data to the file is DONE.", vbOKOnly, "Power meter calibration"

  GoTo Prog_End

Prog_Stop:
  MsgBox "Program Interruption", vbOKOnly, "Power meter calibration"

Prog_End:

End Sub

Private Sub Control_PowerMeter()

  Dim Status As Long            'VISA function status return
  Dim Defrm As Long             'Session to default resource code
```

```
    Dim E4418 As Long             'Session to power meter
    Dim Rslt As Integer
    Dim Rgst As String * 10

    '''Initializing the VISA system
    Status = viOpenDefaultRM(Defrm)
    If (Status <> VI_SUCCESS) Then GoTo VisaErrorHandler

    '''Opening the session to the power meter
    Status = viOpen(Defrm, "GPIB0::13::INSTR", 0, 0, E4418)
    If (Status <> VI_SUCCESS) Then GoTo VisaErrorHandler

    '''Zeroing and calibratingthe power meter
    Status = viVPrintf(E4418, "SYST:PRES" & Chr$(10), 0)
    Status = viVPrintf(E4418, "*CLS" & Chr$(10), 0)
    Status = viVPrintf(E4418, "*ESE 1" & Chr$(10), 0)
    MsgBox "Set the power sensor connected to the POWER REF port in the power meter.", vbOKOnly, "Power meter calibration"
    Status = viVPrintf(E4418, "CAL1:ALL" & Chr$(10), 0)
    Status = viVPrintf(E4418, "*OPC" & Chr$(10), 0)
    Do While Rslt = 0
    Status = viVPrintf(E4418, "*STB?" & Chr$(10), 0)
    Status = viVScanf(E4418, "%t", Rgst)
    Rslt = CInt(CInt(Rgst) And 32)
    Loop

    MsgBox "Zeroing and Calibrating the power sensor is complete.", vbOKOnly, "Power meter calibration"

    '''Closing the resource manager session
    Call viClose(Defrm)

    GoTo Prog_End

  VisaErrorHandler:
    Dim VisaErr As String * 200
    Call viStatusDesc(Defrm, Status, VisaErr)
    MsgBox "Error : " & VisaErr, vbExclamation
    End

  Prog_End:

  End Sub

  Function Limit_Test(Nop As Long, Limit As Double, Corr_data As Variant) As Boolean

    Dim I As Integer

    For I = 0 To Nop - 1

      If Abs(Corr_data(I)) > Limit Then
        SCPI.Source(1).POWer.PORT(1).CORRection.STATe = False
        MsgBox "The corrected data is out of limit!", vbExclamation, "Power meter calibration"
        Limit_Test = False
        Exit Function
      End If

    Next I

    Limit_Test = True

  End Function
```