

Programming Examples

INTRODUCTION TO DSO SOFTWARE TOOLS

Although the WaveMaster DSOs are unrivalled in their ability to process data internally, they are sometimes required to send information to the outside world. To this end, LeCroy provides tools which facilitate interaction with the instruments. This appendix describes two types of software.

The first type is provided in executable form, to enable users to make a quick start with remote control. This group includes examples in ActiveDSO.

The second type is provided in source code, to provide programmers with simple examples for development. Some of these use ActiveDSO as well.

EXECUTABLE PROGRAMS

These are available on the CD-ROM which is shipped with the instruments, and on LeCroy's Web site, at <http://www.lecroy.com/tm/library/software/>.

ActiveDSO Based on Microsoft's ActiveX control technology, ActiveDSO gives leverage to widely available Microsoft software tools, and makes programming within the Microsoft environment easier. ActiveDSO simplifies the computer's interface with the instruments, and programming within Visual C++, Visual Basic, or any other ActiveX compatible applications. For example, Microsoft Excel can even be used to control and retrieve data directly from the WaveMaster DSO. This tool becomes part of the target application and provides seamless access to the full power of the instruments.

Other software For users of LabView, VI's are available for most of LeCroy's DSOs.

Executable programs can be found on the CD-ROM which is shipped with the WaveMaster DSO, or on LeCroy's web-site, at <http://www.lecroy.com/tm/library/software/>.

Executable programs are described more fully in Appendix 3.

SOURCE CODE PROGRAMS

Some examples are provided in Appendix 1 of this manual. These programs can be divided into two types, those using National Instruments GPIB software and hardware, and those using ActiveDSO, which for GPIB, also connects to National Instruments software and hardware. A great benefit of ActiveDSO is that the code written by the user is completely independent of the hardware connection. The selection of GPIB, LAN, or, for earlier DSOs, RS232, is made by a single command near the start of a program.

Programming examples can be found on the CD-ROM which is shipped with the WaveMaster DSOs, or on LeCroy's web-site at <http://www.lecroy.com/tm/library/software/>.

SOURCE CODE EXAMPLE GPIB - 1

Use the Interactive GPIB Program "IBIC"

This example assumes the use of an IBM PC or compatible computer equipped with a National Instruments GPIB interface card. The GPIB driver is left in default state so that the device name "dev4" corresponds to the GPIB address 4, the oscilloscope address. All text is entered by the user. Type in bold represents prompts from the program.

```
IBIC<cr>
program announces itself
: ibfind<CR>
enter board/device name: dev4<CR>
dev4: ibwrt<CR>
enter string: "tdiv?"<CR>
[0100]( cmpl )
count: 5
dev4: ibrd<CR>
enter byte count: 10<CR>
[0100]( cmpl )
count: 10
54 44 49 56 20 35 30 45      T D I V 5 0 E
2D 39                        - 9
dev4: ibwrt<CR>
enter string: "c1:cpl?"<CR>
[0100]( cmpl )
count: 7
```

```

dev4: ibrd<CR>
enter byte count: 20<CR>
[2100]( end cml )
count: 11
43 31 3A 43 50 4C 20 44      C 1 : C P L D
35 30 0A                    5 0 z
dev4: q<CR> to quit the program.

```

SOURCE CODE EXAMPLE GPIB - 2**Use the GPIB Program for IBM PC (High-Level Function Calls)**

The following BASICA program allows full interactive control of the oscilloscope using an IBM PC as GPIB controller. As in Example 1, it is assumed that the controller is equipped with a National Instruments GPIB interface card. All commands can be used following this example simply by entering the text string of the command. For example, "C1:VDIV 50 MV", without the quotation marks. The program automatically displays the information sent back by the oscilloscope in response to queries.

In addition, a few utilities have been provided for convenience. The commands ST and RC enable waveform data to be stored on, or retrieved from, a disk if the correct drive and file names are provided. The command LC returns the oscilloscope to local mode. Responses sent back by the oscilloscope are interpreted as character strings and are thus limited to a maximum of 255 characters.

```

'  INCLUDE - This line is symbolic of the National Instruments routines which
allow your language
'  to communicate with GPIB.  Please see your NI manual for information.

CLS
PRINT "Control of the 9300 via GPIB and IBM PC" : PRINT
PRINT "Options :  EX to exit           LC local mode"
PRINT "              ST store data      RC recall data" : PRINT ""
LINE INPUT "GPIB-address of oscilloscope (1...16)? :",ADDR$
DEV$ = "DEV" + ADDR$      '      Construct DSO address.
CALL IBFIND (DEV$,SCOPE%)
IF SCOPE% < 0 THEN PRINT "IBFIND ERROR" : END      '      Cannot find DSO.
TMO% = 10      '      Timeout = 300 msec (rather than default 10 sec)
CALL IBTMO(SCOPE%,TMO%)
LOOP% = 1

WHILE LOOP%
LINE INPUT "Enter command (EX --> Exit) : ",CMD$
Select Case CMD$
Case "ex", "EX" : LOOP% = 0 : GOSUB LocalMode : END
Case "st", "ST" : GOSUB StoreData : GOTO LoopEnd

```

APPENDIX I: *Program Examples*

```
        Case "rc", "RC" : GOSUB RecallData : GOTO LoopEnd
        Case "lc", "LC" : GOSUB LocalMode : GOTO LoopEnd
        Case "" : GOTO LoopEnd
    End Select

    CALL IBWRT(SCOPE%,CMD$)
    IF IBSTA% < 0 THEN GOSUB GPIBError : END
    GOSUB GetData
    LoopEnd : WEND

LocalMode:
` Put DSO into Local Mode.
CALL IBLOC (SCOPE%) : PRINT
RETURN

GetData :
` Get data from DSO.
` If there are no data to read, simply wait until timeout occurs
CALL IBRD (SCOPE%,RD$)
I = IBCNT% 'IBCNT% is the number of characters read
FOR J = 1 TO I
    PRINT MID$ (RD$,J,1);
NEXT J
PRINT : RETURN

StoreData :
` Store waveform data in a file.
RD1$=SPACE$(3)
LINE INPUT "Specify trace (TA...TD,M1...M4,C1...C4): ",TRACE$
LINE INPUT "Enter filename : ",FILE$
CMD$="WFSU NP,0,SP,0,FP,0,SN,0; CHDR SHORT"
CALL IBWRT (SCOPE%,CMD$)
CMD$=TRACE$+":WF?"
CALL IBWRT (SCOPE%,CMD$)
CALL IBRD (SCOPE%,RD1$) ' Discard first 3 chars of response
CALL IBRDF (SCOPE%,FILE$)
IF IBSTA% < 0 THEN GODUB GPIBError : END
PRINT : RETURN

RecallData :
` Recall waveform data from file and send them to DSO.
LINE INPUT "Specify target memory (M1...M4):",MEM$
LINE INPUT "Enter filename : ",FILE$
CMD$=MEM$+": "
```

```
CALL IBWRT (SCOPE%,CMD$)
CALL IBWRTF (SCOPE%,FILE$)
  IF IBSTA% < 0 THEN GOSUB GPIB Error : END
  RETURN

GPIBError:
PRINT "GPIB ERROR -- IBERR: ";IBERR%;"IBSTA: ";HEX$(IBSTA%) : RETURN
```

NOTE:

- *It is assumed that the National Instruments GPIB driver GPIB.COM is in its default state. This means that the interface board can be referred to by its symbolic name 'GPIB0' and that devices on the GPIB with addresses 1 to 16 can be called by the symbolic name 'DEV1' to 'DEV16'.*
- *Lines 1–99 are a copy of the file DECL.BAS supplied by National Instruments. The first six lines are required for the initialization of the GPIB handler. DECL.BAS requires access to the file BIB.M during the GPIB initialization. BIB.M is one of the files supplied by National Instruments, and must exist in the directory currently in use.*
- *The first two lines of DECL.BAS each contains a string "XXXXX" that must be replaced by the number of bytes that determine the maximum workspace for BASICA (computed by subtracting the size of BIB.M from the currently available space in BASICA). For example, if the size of BIB.M is 1200 bytes and, when BASICA is loaded, it reports "60200 bytes free", "XXXXX" would be replaced by the value 59000 or less.*
- *The default timeout of 10 seconds is modified to 300 ms during the execution of this program. However, the default value of the GPIB handler remains unchanged. Whenever a remote command is entered by the user, the program sends it to the instrument with the function call IBWRT. Afterwards, it always executes an IBRD call, regardless of whether or not a response is expected. If a response is received it is immediately displayed. If there is no response, the program waits until time-out and then asks for the next command.*

SOURCE CODE EXAMPLE GPIB - 3

USE GPIB Program for IBM PC (Low-Level Function Calls)

This example has the same function as Example 2, but is written with low-level function calls. The program assumes that the controller (board) and oscilloscope (device) are at addresses 0 and 4, respectively, and the decimal addresses are:

	Listener Address	Talker Address
CONTROLLER	32(ASCII <space>)	64 (ASCII @)

APPENDIX I: *Program Examples*

DEVICE	32+4=36 (ASCII \$)	64+4=68 (ASCII D)
--------	--------------------	-------------------

```
`  INCLUDE NATIONAL INSTRUMENTS GPIB ROUTINES
CLS
PRINT "Control of the 9300 (address 4) via GPIB and IBM PC" : PRINT
PRINT "Options :      EX to exit          LC local mode"
PRINT "                  ST store data      RC recall data" : PRINT

UnListen$ = Chr$ ( 63) : UnTalk$ = Chr$ (95)          `      General UnListen and
UnTalk
BaseListen% = 32 : BaseTalk% = 64
DSOAddress% = 4
DSOListen$ = UnListen$ + UnTalk$ + Chr$ (BaseTalk%) + Chr$ (BaseListen% +
DSOAddress%)
DSOTalk$ = UnListen$ + UnTalk$ + Chr$ (BaseListener%) + Chr$ (BaseTalk% +
DSOAddress%)

BDNAME$= "GPIB0" : CALL IBFIND (BDNAME$,BRD0%)
  IF BRD0% < 0 THEN PRINT "IBFIND ERROR" : STOP
CALL IBSIC (BRD0%) :
  IF IBSTA% < 0 THEN PRINT "IBFIND ERROR" : STOP
LOOP = 1

  WHILE LOOP
  LINE INPUT "Enter command (EX --> Exit) : ",CMD$
  V% = 1: CALL IBSRE(BRD0%,V%)
    IF CMD$ = "ex" OR CMD$ = "EX" THEN LOOP = FALSE : GOTO ExitGPIB
    IF CMD$ = "st" OR CMD$ = "ST" THEN GOSUB StoreData : GOTO LoopEnd
    IF CMD$ = "rc" OR CMD$ = "RC" THEN GOSUB RecallData : GOTO LoopEnd
    IF CMD$ = "lc" OR CMD$ = "LC" THEN  GOSUB DSOLocal : GOTO LoopEnd
    IF CMD$ = "" THEN GOTO LoopEnd
  CALL IBCMD (BRD0%,DSOListen$) : CALL IBWRT(BRD0%,CMD$): GOSUB GetData
  LoopEnd : WEND

ExitGPIB : CALL IBSIC (BRD0%): V%=0 : CALL IBSRE (BRD0%,V%)
CALL IBSIC (BRD0%) : END

DSOLocal :
V% = 0 : CALL IBSRE (BRD0%,V%) : PRINT : RETURN

GetData :
CALL IBCMD (BRD0%,DSOTalk$) : CALL IBRD ( BRD0%,RD$) : I=IBCNT%
  FOR J=1 TO I
    PRINT MID$ (RD$,J,1);
```

```
        NEXT J
PRINT : RETURN

StoreData :
RD1$=SPACE$(3)
LINE INPUT "Specify trace (TA...TD,M1...M4,C1...C4) : ",TRACE$
LINE INPUT "Enter filename : ",FILE$
CALL IBCMD (BRD0%, DSOListen$)
CMD$="WFSU NP,0,SP,0,FP,0,SN,0;CHDR SHORT"
CALL IBWRT (BRD0%,CMD$)
CMD$=TRACE$+"WF?": CALL IBWRT (BRD0%,CMD$)
CALL IBCMD (BRD0%,DSOTalk$) : CALL IBRD (BRD0%,RD1$)
CALL IBRDF (BRD0%,FILE$)
    IF IBSTA% < 0 THEN GOSUB GPIBError : STOP
PRINT : RETURN

RecallData :
LINE INPUT "Specify target memory (M1...M4) : ",MEM$
LINE INPUT "Enter filename : ",FILE$
CALL IBCMD (BRD0%,DSOListen$)
CMD$=MEM$+"": CALL IBWRT (BRD0%,CMD$)
CALL IBWRTF (BRD0%,FILE$)
    IF IBSTA% < 0 THEN GOSUB GPIBError : STOP
PRINT : RETURN

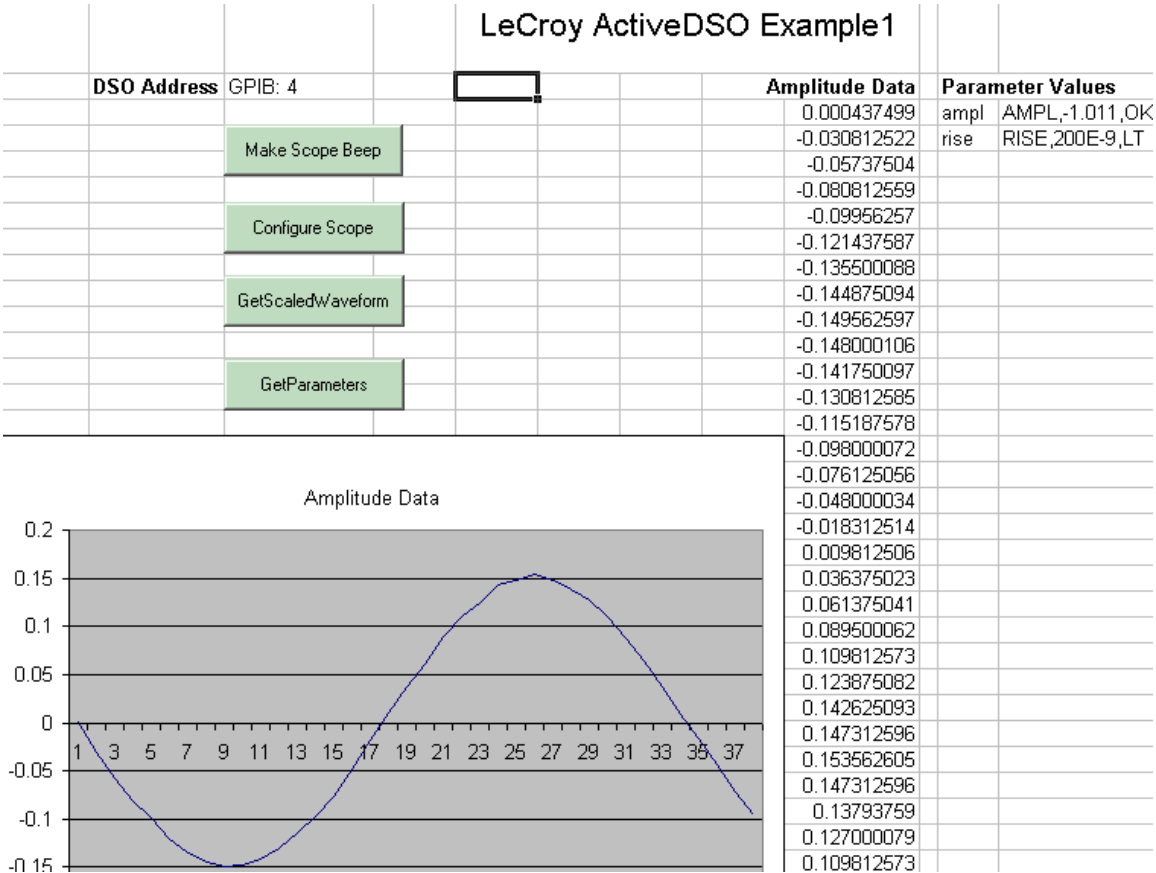
GPIBError :
PRINT "GPIB ERROR -- IBERR : ";IBERR%;"IBSTA : ";HEX$ (IBSTA%) : RETURN

END
```

NOTE: The Template also describes an array named DUAL. This is simply a way to allow you to use the INSPECT? query to examine the two data arrays together.

SOURCE CODE EXAMPLE ACTIVEDSO – 1 AND 2

The picture shows the screen of a program, ActiveDSOExcel1, that uses ActiveDSO embedded in Excel. ActiveDSOExcel2 is similar.



This example shows how to create some simple applications. It can be a basis for further explorations in ActiveDSO. This example is included in the ActiveDSO system that can be downloaded from LeCroy’s web-site at <http://www.lecroy.com/tm/library/software/>.

The fragment below is the subroutine which reads waveform data from the DSO and places the values in column I (9th column) of the spreadsheet.

```
Private Sub GetScaledWaveformButton_Click()  
Dim o As Object      '      Define variable o as an object.  
  
    ' Equate object o with the ActiveDSO object LeCroy.ActiveDSOctrl1.  
    Set o = CreateObject("LeCroy.ActiveDSOctrl1")  
  
    ' Read the device address from cell 2C, and use it to connect the PC  
    to the DSO.  
    Dim deviceAddress As String  
    deviceAddress = Worksheets("Sheet1").Cells(2, 3).Value  
    Call o.MakeConnection(deviceAddress)  
    ' Set the DSO into remote control mode.  
    Call o.SetRemoteLocal(1)  
  
    ' Define an array of the size you need for your waveform data.  
    ' Read the waveform data from the DSO into the array.  
    Dim waveArray  
    waveArray = o.GetScaledWaveform("C1", 500000, 0)  
  
    ' Place the data into column I (9th column).  
    Dim i As Long  
        For i = 0 To UBound(waveArray)  
            Worksheets("Sheet1").Cells(i + 3, 9).Value = waveArray(i)  
        Next i  
End Sub  
.  
.  
.  
  
    ' Release the control.  
    Call o.SetRemoteLocal(0)  
    Set o = Nothing  
End Sub
```



```
Dim DeviceAddress As String
' Read the device address from cell 2D, and use it to connect the PC
to the DSO.
DeviceAddress = Worksheets("Sheet1").Cells(2, 4).Value
Call o.MakeConnection(DeviceAddress)
' Set the DSO into remote control mode.
Call o.SetRemoteLocal(1)
' Set TimeOut to 3 seconds instead of the default, which is 10
seconds.
Call o.SetTimeOut(3)

Dim ErrorFound, GetOut, Waiting As Boolean
Dim Row, Column, Counter, LoopTotal, TCounter, FCounter, TestLength As
Integer
Dim HoldOff, NewTime, XTime, StartRow As Single
Dim ControlDatum, NextData, NextNextData, Query, Quit, ReStart As
String
' Column containg the remote control commands.
Column = 3: StartRow = 10
' Cell(2,6) is the cell containing the row for restart after a pause.
' It is empty when starting from top of the list of commands, which is
' Cell(StartRow,Column).
If Worksheets("Sheet1").Cells(2, 6).Value < 1 Then
Row = StartRow
Else
Row = Worksheets("Sheet1").Cells(2, 6).Value
End If

Query = "?": Quit = "quit"
' Set exit flag, error flag and waiting flag to false.
. . . . .
.

' Execute the command line LoopTotal times.
For Counter = 1 To LoopTotal
' Continue with loop only if no error was found in the loop.
If ErrorFound = False Then
' Send data to instrument.
If Len(NextData) > 1 Then Call o.WriteString(NextData, 1)
' Erase contents of cell ready for response from instrument.
Worksheets("Sheet1").Cells(Row, Column + 2).Value = ""
If LoopTotal > 1 Then
' Show progress in cell.
Worksheets("Sheet1").Cells(Row, Column + 1).Value = Counter
End If
```



```
    ' Look for "?" in command string.
      If InStr(NextData, Query) > 0 Then
        ' Collect response from instrument.
        Worksheets("Sheet1").Cells(Row, Column + 2).Value =
o.ReadString(1000)
        End If

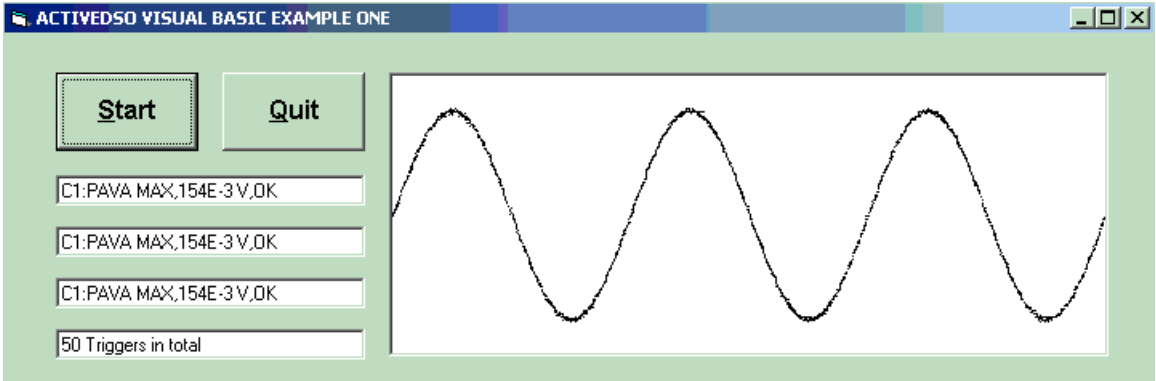
    ' Check for Error
    If o.ErrorFlag = True Then
      ' Show error message.
      Worksheets("Sheet1").Cells(Row, Column + 2).Value =
o.ErrorString
      Waiting = True: HoldOff = 0: ErrorFound = Tru
      ' Store command line for restart.
      Worksheets("Sheet1").Cells(2, 6).Value = Row + 1
      End If
    End If
  Next Counter
. . . . .
.

' Release the control.
Call o.SetRemoteLocal(0)
Set o = Nothing
End Sub
```

ActiveDSOExcel4 is a similar program using the Pass-Fail system.

EXAMPLE ACTIVEDSO – 5

The picture below shows the screen of program ActiveDSOVB1, a Visual Basic program using ActiveDSO.



This example shows how to arm the trigger, get a parameter from the WaveMaster DSO, get waveforms, and draw them on the screen of the PC.

The source code can be found on the CD-ROM, in file ActuveDSOVB1.frm.

TRANSLATION EXAMPLES

Some source code examples are available for decoding binary waveform files. These are TranWM in Microsoft Visual Basic and TraceLook in VBA/Microsoft Excel.

INTRODUCTION TO ACTIVEDSO

This ActiveX™ control enables LeCroy oscilloscopes to be controlled by, and to exchange data with, a variety of Windows applications that support the ActiveX standard. MS Office programs, Internet Explorer, Visual Basic, Visual C++, Visual Java, and Matlab (v5.3) are a few of the many applications that support ActiveX controls. ActiveDSO is available on CD-ROM, or on the internet at www.lecroy.com

With **ActiveDSO** you can develop your test program using standard GPIB commands. For easy integration of your scope data with your Windows Application (through GPIB or Ethernet 10BaseT/100BaseT), ActiveDSO, helps you:

- Generate a report by importing scope data right into Excel or Word.
- Archive measurement results on the fly in a Microsoft Access Database.
- Automate tests using Visual Basic, Java, C++, Excel (VBA).
- The ActiveDSO control hides the intricacies of programming and provides a simple and consistent interface to the controlling application. With less than 10 lines of VBA (Visual Basic for Applications) code in an Excel macro the spreadsheet can recover pre-scaled waveform data from a remote instrument.
- The ActiveDSO control can also be embedded visually in any OLE automation compatible client, and can be used manually without any need for programming. It will run on any PC running Windows 95, Windows 98, or Windows NT.

There are two fundamental ways to use the control:

- As a visible object embedded in an OLE Automation compatible Client (PowerPoint for example) showing a captured display image. See Embedded Control Example for more details.
- As an invisible object accessed via a scripting language (Visual Basic for Applications, for example) to remotely control an instrument. See Accessing from VBA for more details.

VBA (Visual Basic for Applications) is the programming language built into many of the more recent Windows applications. It is a subset of Visual Basic that makes it very simple to utilize the services of OLE Automation Servers and ActiveX Controls.

The following VBA subroutine demonstrates how easy it is to connect to a WaveMaster DSO and send remote commands to it.

```
Sub LeCroyDSOTest()  
    Dim o As Object  
        Set o = CreateObject("LeCroy.ActiveDSOctrl.1")  
    Call o.AboutBox           ' Present the control's About box  
    Call o.MakeConnection("IP: 172.28.11.26) 'Connect to device on LAN  
    Call o.WriteString("BUZZ BEEP", True) ' Make the DSO beep  
End Sub
```

APPENDIX I: *Program Examples*

Example Syntax:

Boolean controlName.WriteString

The WriteString method has the following arguments:

controlname The name of the ActiveDSO control object

textStringString Text string to send to the device

EOI Boolean TRUE = terminate with EOI

Returns:

True on success, False on failure

Remarks:

This method sends a string command to the instrument.

If EOI is set to TRUE, the device will start to interpret the command immediately. This is normally the desired behavior.

If EOI is set to FALSE, a command may be sent in several parts with the device starting to interpret the command only when it receives the final part, which should have EOI set TRUE.

USING ACTIVEDSO

ActiveDSO is highly suitable for fast program development in the Microsoft environment. This program is a control of ActiveX, the software technology developed by Microsoft as a subset of its COM model.

ActiveDSO facilitates programming with the WaveMaster DSO by providing a ready interface between the instrument and the host computer. Programs such as Visual C++, Visual Basic, or Visual Basic for Applications (VBA) can be used under remote control without concern for interfacing complications. ActiveDSO acts as the key design structure allowing effective integration of software from the different manufacturers supporting ActiveX containment.

INSTANTIATION

This ActiveX component can be instantiated more than once by using the Visual Basic function CreateObject. Once the object is created, invoking the connection method will initialize it. ActiveDSO enables control of the WaveMaster DSO from a variety of PC desktop applications. The complexities of programming with Ethernet are fully encapsulated within this control. For example, with fewer than ten lines of VBA code in an Excel Macro, the spreadsheet can recover pre-scaled waveform data from the WaveMaster DSO. An example is provided in this appendix.

ActiveDSO control can be used in two fundamental ways:

1. As a visible object embedded in an OLE automation compatible client (PowerPoint, for example) showing a captured WaveMaster DSO display image. See the Embedded Control example below for more details.

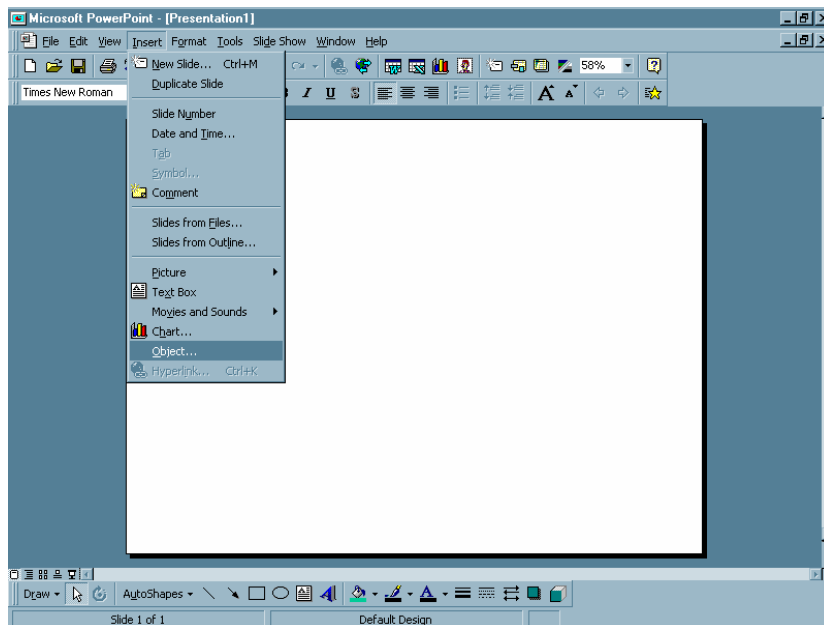
2. As an invisible object accessed through a scripting language (VBA, for example) to remotely control the WaveMaster DSO. See VBA example below for more details.

The ActiveDSO control may be embedded in any ActiveX containment-capable client, and may be used manually without need of any programming or scripting.

EXAMPLE USING POWERPOINT 97

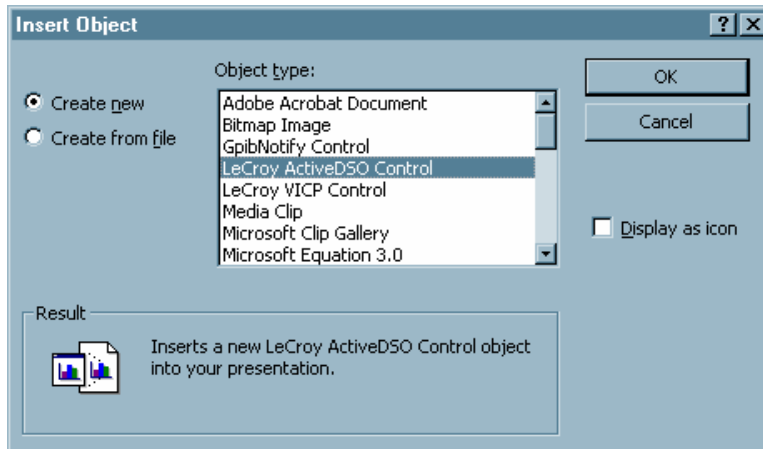
This example shows the control being embedded in a Microsoft PowerPoint slide. The waveform captured by the WaveMaster DSO can be easily imported into PowerPoint with just a few mouse clicks:

1. Ensure that the ActiveDSO files from the CD-ROM are installed on the PC.
2. Verify that the PC and WaveMaster DSO are properly connected to the Ethernet.
3. Open a new blank presentation in PowerPoint.
4. Select “Insert,” then **Object**, as shown here:

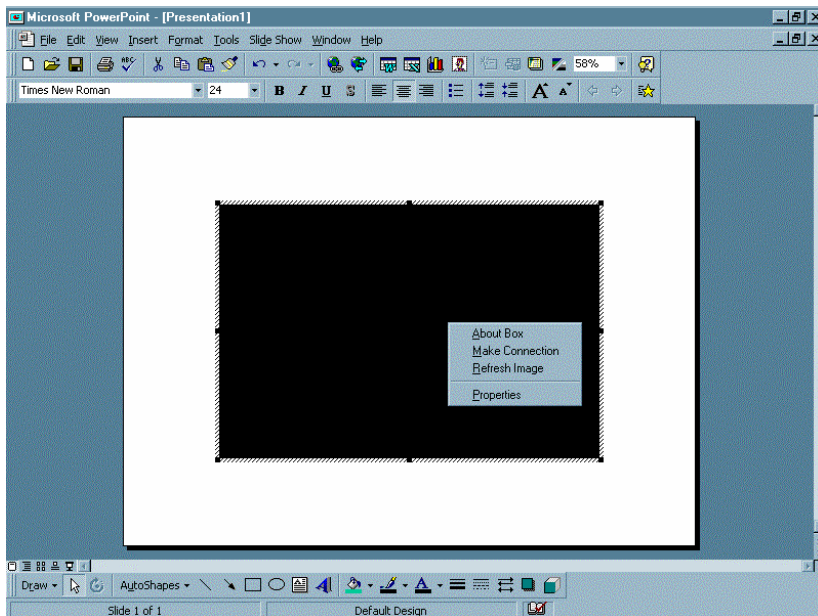


APPENDIX I: *Program Examples*

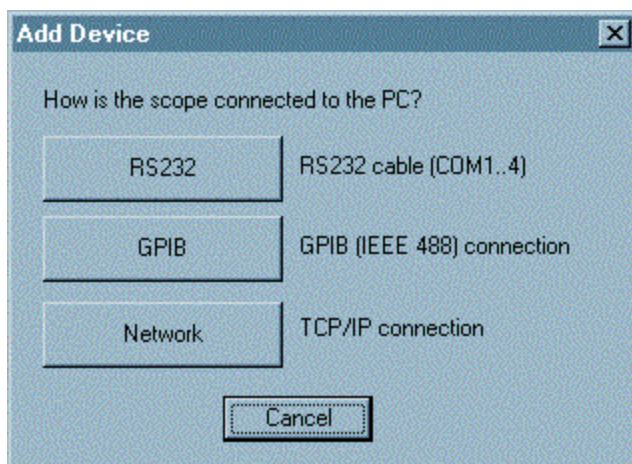
5. From the pop-up window, select LeCroy ActiveDSO object as shown here:



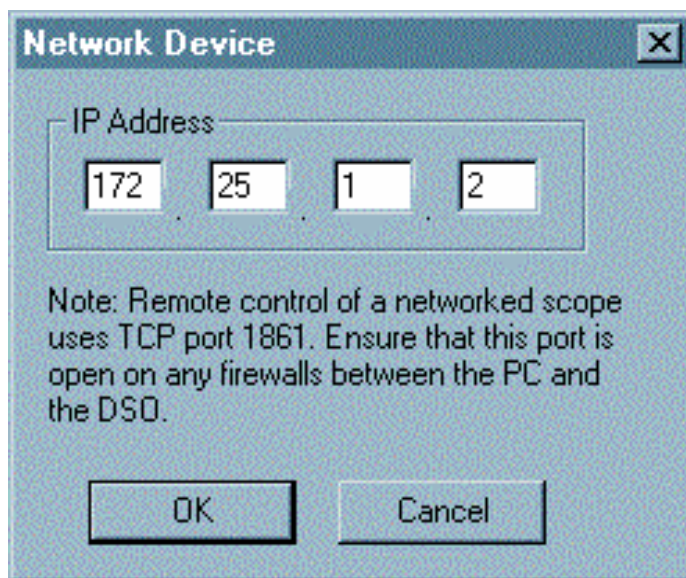
6. Right-click the object and select “Make Connection.”



7. Select “Network TCP/IP connection” as shown here (“scope” = WaveMaster):

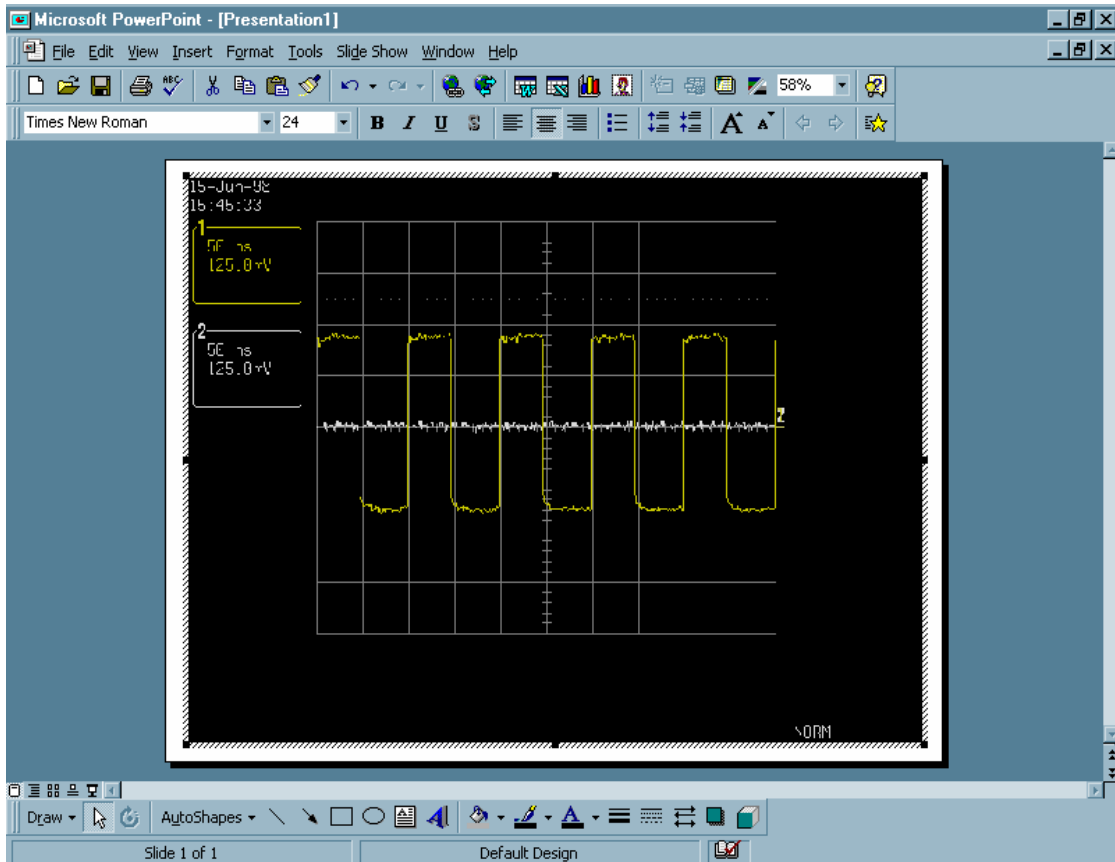


8. Enter the WaveMaster DSO's IP address and click “OK.”



APPENDIX I: *Program Examples*

9. Right-click the object again and select the **Refresh Image** menu item. A captured waveform will be displayed similar to the one shown here:



WaveMaster DSO's captured waveform imported into PowerPoint.

Once the ActiveDSO™ object has been properly set within the application, a macro script can be created utilizing an object method such as WriteString() to send DISP ON, C1:TRA ON, TRMD. Then RefreshImage() method can be used to update the screen.

EXAMPLE IN VBA

VBA is the programming language built in to many of the more recent Windows applications. It is a subset of Visual Basic that makes using OLE Automation Servers and ActiveX Controls very simple. The following VBA subroutine demonstrates how easy it is to connect to a WaveMaster DSO and send remote commands to it.

```
Sub LeCroyDSOTest()  
    Dim dso As Object  
  
    Set dso = CreateObject("LeCroy.ActiveDSO.1")  
  
    Call dso.AboutBox Present the control's About box  
    Call dso.MakeConnection("IP:172.25.1.2") Connect to the unit  
    Call dso.WriteString("DISP ON", 1) Enable the internal display routine  
    Call dso.WriteString("TRMD AUTO", 1) Set the trigger mode to AUTO  
End Sub
```

To enter the VBA editor in members of the Microsoft Office suite:

1. Select Tools ? Macro ? Visual Basic Editor menu item.
2. When the VBA window appears, select the Insert ? Module menu item.
3. Copy the above example into the editor window that appears.

To execute:

4. Position the text cursor within the subroutine.
5. Either select the Run ? Run Sub/UserForm or press function key F5.

NOTE: For more information, see the ActiveDSO on-line Help. On-line Help contains VisualC++ example, explanations of ActiveDSO Methods and Properties.



§ § §

BLANK PAGE