Communicating with Logic Analyzers through the LAN

Most current Agilent logic analyzers have a network interface that can be used to control them or download files. These analyzers can be grouped into two families based on the method used to communicate with them: The 1670 and 16500 series share the same type of programming, through a command parser. The 16700 series does not have a SCPI (Standard Commands for Programmable Instruments) parser. Instead, it has a Remote Programming Interface (RPI). Both logic analyzer families have built-in NFS capability as well as an FTP server for transferring files to and from the analyzer over the LAN.

I. The 1660/1670 and 16500 series

While the LAN interface is standard with the 1670 and 16500C series, it was optional with the 1660 and 16500A/B series. In this section we will attempt to show how to connect to the instrument for file transfer or control. Examples given will only be for the purpose of testing connectivity. For more detail on programming these instruments, please refer to their respective programming manuals.

A. File transfer

There are two methods of transferring files

1. NFS (Network File System)

The 1670/16500C analyzers have a built-in NFS server. Their file system can be mounted from a UNIX system (or a Windows system that has an NFS client) making them appear to be part of the local file system. There is no need to make any changes on the logic analyzer as the appropriate directories are already exported.

The syntax for mounting the NFS filesystem from a UNIX x-term window is:

```
mount <hostname or IP address>:/<data or control> /<local directory>
```

for example:

```
mount 156.140.153.48:/data  /remotelogic
```

2. FTP (File Transfer Protocol)

These analyzers also have an FTP server. Log-in to the FTP server for file transfer with the user-name "data." No password is required.

Example:

At the DOS prompt type "ftp <IP address or hostname> and press ENTER. You will be prompted for a user name. Enter the user name "data" and you will be granted access without a password. At this point you can use standard ftp commands such as "put" and "get" to transfer files.

Here is a list of commands available from the FTP client:

```
!           delete      literal     prompt      send
?           debug       ls          put         status
append      dir         mdelete     pwd         trace
ascii       disconnect  mdir        quit        type
bell        get         mget        quote       user
binary      glob        mkdir       recv        verbose
bye         hash        mls         remotehelp
cd          help        mput        rename
close       lcd         open        rmdir
```

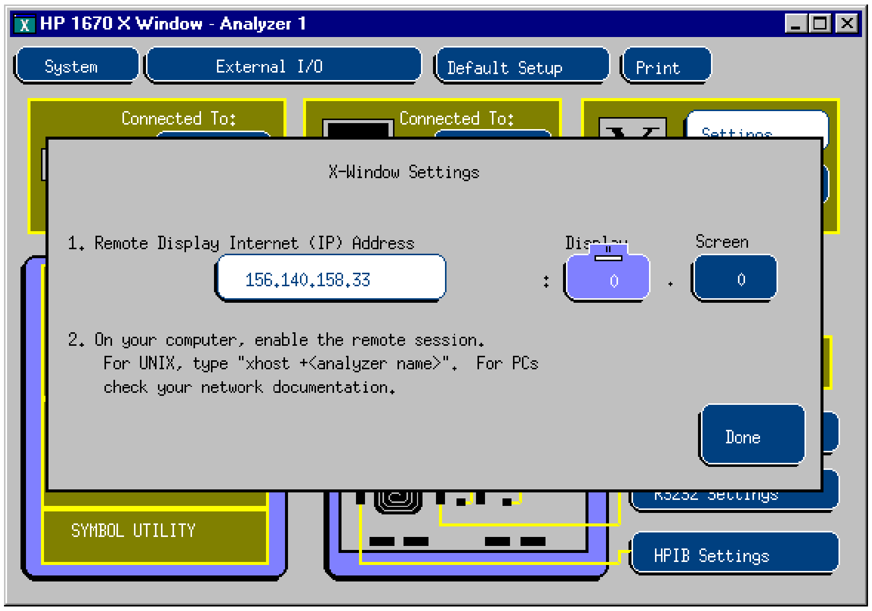
B. Control

There are five ways to control the 1670 and 16500C over the LAN.

1. X-Windows - a remote "front panel"

The logic analyzer is capable of "serving" its video data for interactive display on a remote station. An X-Windows client is required to use this feature.

a. Start an X-Windows process on your workstation
b. Enter the IP address of your workstation in the X-Windows setup screen of the logic analyzer (upper right-hand corner of system configuration screen).

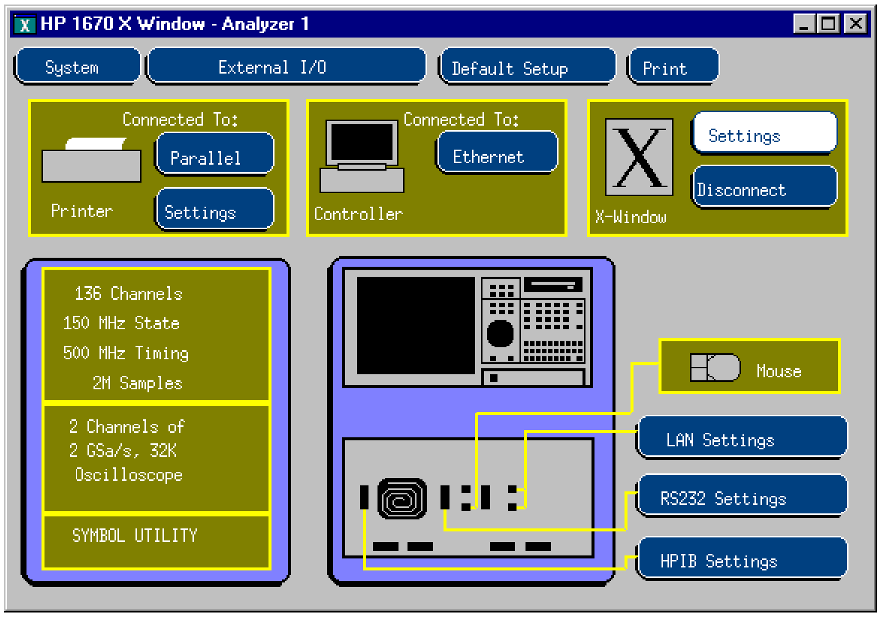


X-Window Settings

1. Remote Display Internet (IP) Address 156.140.158.33 : Display 0 . Screen 0

2. On your computer, enable the remote session.
For UNIX, type "xhost +<analyzer name>". For PCs
check your network documentation.

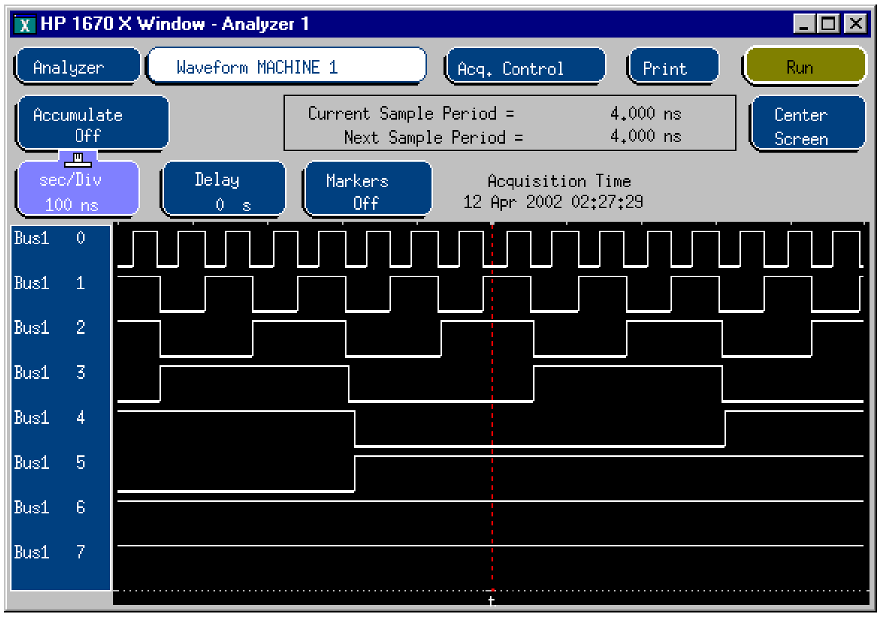
c. Press “Connect” on the logic analyzer



136 Channels
150 MHz State
500 MHz Timing
2M Samples

2 Channels of
2 GSa/s, 32K
Oscilloscope

A window that looks like the logic analyzer display will appear on your workstation and you will be able to control the analyzer from the workstation by using the mouse (touch-screen is not available for remote control).



Examples given in the remainder of this section will be primarily aimed at downloading data from the analyzers. In doing so, commands will be sent to the instrument illustrating the process of programming the analyzer. For further details, please refer to the respective programming manual. Data can be downloaded in ASCII or binary format.

ASCII

To get results in ASCII format you can display a listing of desired data on the logic analyzer screen and then use the ":SYST:PRINT? ALL" query to move it to the output buffer. This is the simplest method, but also the slowest.

Binary

Unlike ASCII downloads, it is not necessary to display the listing with a binary download. The data will be downloaded from the selected module. Use the ":SEL n" command to specify desired module (where n is the position of the module in the mainframe--for slot A, n=1).

In order to get the data in the proper binary format for downloading, it must be "unpacked" first. This can be done with the ":DBL UNP" command.

To send data to the output buffer in binary format, use the ":SYST:DATA?" query.

2. FTP

A text file named "program" containing commands will be executed if placed in the logic analyzer's /system directory. After the commands are executed, the results will overwrite the original file.

ASCII download

a. After capturing the desired listing on the logic analyzer display, create a text file and place it in a directory that is easily accessible, for example c:\control\myprog.txt. This file should contain the following line:

```
:SYST:PRINT? ALL
```

b. From the DOS prompt type

```
"ftp <IP Address or Hostname>"   and press ENTER
```

c. Login as "control"

d. From the ftp prompt type

```
"put  c:\control\myprog.txt  /system/program" and ENTER
```

e. At this point you must wait sufficient time for the results to be copied to the "/system/program" file. This can take a great deal of time, especially if you have acquired a lot of data.

f. From the ftp prompt type "binary" and press ENTER

   Even though the data is in ASCII format, transfers seem to work better in binary mode.

g. From the ftp prompt type

   "get /system/program c:\control\mydata.txt" and press ENTER. The "mydata.txt" file now contains the data in ASCII format.

Binary download

a. After acquiring the data, create two text files. The first one should contain the commands:

```
:SEL n
:DBL UNP
```

The second file should contain the query:

```
:SYST:DATA?
```

b. Following the methods described in steps b-d, above, "put" these files in the /system directory waiting a few minutes between the first file and the second to allow the unpacking process to start before requesting the data.

c. Complete the download following steps e-g, above.

3. NFS

For either Binary or ASCII downloads, you can place the "program" files (described in the previous section) in the /system directory and retrieve the results using NFS. You must connect as “control.”

a. NFS-mount the logic analyzer's file system as "control"

Example: `mount 156.140.153.48:/control  /remotelogic`

b. This file system is now seen as part of the remote workstation's file system. Copy the "program" file (described above) to the /system directory (use the appropriate file/s depending on whether you are doing a Binary or ASCII file transfer.

c. Wait a sufficient amount of time for the data to be written to the program file

d. Copy the overwritten program file to another location, under another name (for example mydata.dat).

4. Telnet

A Telnet client can be used to connect directly to the command parser at port 5025. The client's ability to log the session can be used to save the data returned by the logic analyzer.

a. After listing the desired data on the logic analyzer's screen, telnet to port 5025

```
C:\>telnet 156.140.153.38 5025
```

b. Turn on local echo

   Terminal/Preferences/Local Echo

c. Turn on logging

   Terminal/Start Logging - Give path and filename

d. Send query

   ```
   :SYST:PRINT? ALL
   ```

If doing a binary download, use the following commands, instead:

```
:SEL n
:DBL UNP
```

Wait a while before sending the query:

```
:SYST:DATA?
```

Log file will contain the desired data

If you want to make a fresh acquisition, send the following commands before you unpack the data (binary) or send the :SYST:PRINT? query (ASCII):

```
:SEL n
:START
```

5. TCP/IP sockets

Programming languages such as C, VB, VEE, LabWindows and LabVIEW can be used to implement socket communications with the analyzers. In the following examples, the analyzer was set up ahead of time so that the :START command (if issued) would initiate a new acquisition. In some cases, it was assumed that the data had already been acquired.

a. C example using Agilent SICL to download ASCII data

```
#include "sicl.h"
#include<stdio.h>

      void main()
      {
            FILE *datafile;
            datafile = fopen("c:\\datafile.txt","w"); //open file

            char data[115000]={" "};            //allocate memory
                                                //for 4096 states
            INST la;
            la=iopen("lan,5025[192.168.1.5]");  //open session
                                                //to port 5025
            iprintf(la, ":SELECT 1\n");         //select LA
                                                //board
            iprintf(la, ":START\n");            //start new
                                                //acquisition
            iprintf(la, ":SYST:PRINT? ALL\n");  //query result
            iread(la,data,114986,0,NULL);       //read response
            printf("%s\n",data);                //display result
            fprintf(datafile,"%s\n",data);      //save data to file
            iclose(la);                         //close session
            fclose(datafile);                   //close file
      }
```

b. C example using Agilent VISA to download ASCII data

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include "visa.h"
void main()
{
      char junktest[]={"Absol"};    //or "Relat"
      char str0[10],str1[8],str2[8],str3[6];
      char junkstr[]={"xxxxxxxxxxxxxxx"};
      const int states=4096;  //number of states in acquisition
      int statenum[states],i=0;
      double acqtime;

      FILE *savedata;
      savedata = fopen("c:\\savedata.txt","w"); //open file

      ViSession sesn, la;
      viOpenDefaultRM (&sesn);                  //open VISA session

      viOpen(sesn,"TCPIP0::192.168.1.5::5025::SOCKET",
                        VI_NULL, VI_NULL,&la);  //open INST session

      viPrintf(la,":SYST:PRINT? ALL\n");  //query instrument

      while(!strstr(junkstr,junktest))
      viScanf(la,"%s",&junkstr);          //discard unwanted data
      viScanf(la,"%s\t%s\t%s\t%s\r\n",
            &str0,&str1,&str2,&str3);     //read first state
      do{
            statenum[i] = atoi(str0);     //convert ascii to integer
            acqtime = atof(str2);         //convert ascii to double

            if(strstr(str3,"ns"))
                  acqtime = acqtime * 0.001;    //convert ns to us
            if(strstr(str3,"ms"))
                  acqtime = acqtime * 1000;     //convert ms to us

            printf("%s\t%s\t%s\t%s\n",str0,
                              str1,str2,str3);  //display data

            fprintf(savedata,"%d\t%s\t%2.3lf\tus\n",
                  statenum[i],str1,acqtime);    //save data to file

            viScanf(la,"%s\t%s\t%s\t%s\r\n",
                  &str0,&str1,&str2,&str3);     //read data from inst
            i++;                          //increment loop counter
      }while(i<states);                         //end loop

      viClose(la);                              //close INST session
      viClose(sesn);                            //close VISA session
      fclose(savedata);                         //close FILE
}
```

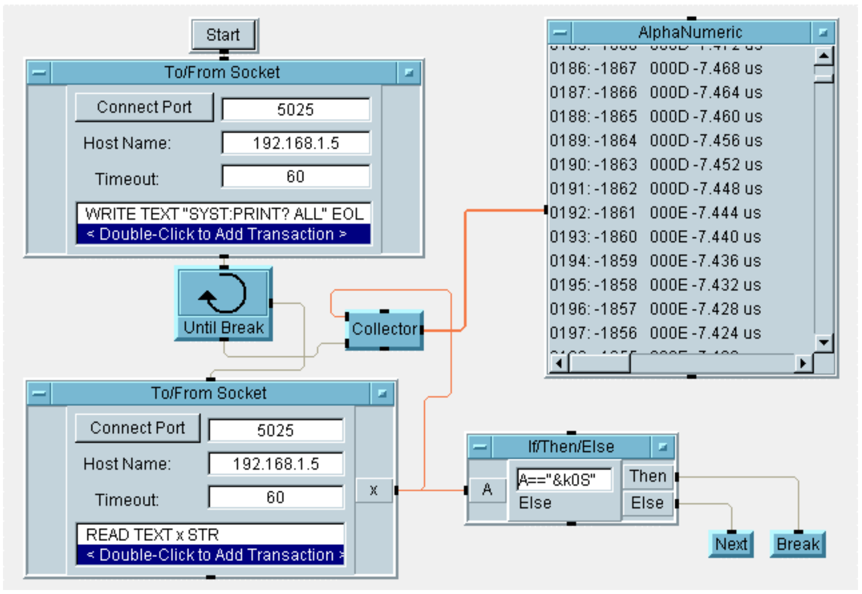
c. VB example using Agilent SICL

```
Dim la As Integer
Dim bufr As String * 40
Dim i As Integer
Private Sub cmdQuit_Click()
      iclose (la)                                   'close instrument session
      End                                           'exit program
End Sub

Private Sub Form_Load()

      la = iopen("lan,5025[156.140.153.48]")        'open instrument session
      Call iprintf(la, ":SYST:PRINT? ALL" & Chr$(10)) 'query instrument
      Call itermchr(la, 10)                         'set termination character

      Do
            Call iread(la, bufr, 40, 0, 0&)         'get data from instrument
      Loop Until (InStr(1, bufr, "Absolute") > 0)   'discard unwanted data

      Call iread(la, bufr, 40, 0, 0&)               'discard blank line

      For i = 1 to 4096
            bufr = "                              "  'clear buffer
            Call iread(la, bufr, 40, 0, 0&)          'read another line
            lstRdgs.AddItem (bufr)                   'add line to display
      next i

End Sub
```

d. VB example using Agilent VISA

```
Dim VISAsesn As Long
Dim la As Long
Dim buffr As String * 40
Dim retcnt As Long

Private Sub cmdQuit_Click()
      End
End Sub

Private Sub Form_Load()

      Call viOpenDefaultRM(VISAsesn)
      Call viOpen(VISAsesn, "TCPIP0::192.168.1.4::5025::SOCKET",
                              VI_NULL,VI_NULL, la)

      Call viSetAttribute(la, VI_ATTR_TERMCHAR, 10)

      Call viWrite(la, "SYST:PRINT? ALL" & Chr$(10), 16, retcnt)

      Do
            Call viRead(la, buffr, 30, retcnt)        'get data from
                                                      'instrument
      Loop Until (InStr(1, buffr, "sol") > 0)   'discard unwanted data

      Call viRead(la, buffr, 30, retcnt)              'discard blank
                                                      'line

      Do
            buffr = "                            "    'clear buffer
            Call viRead(la, buffr, 28, retcnt)  'read another line
            lstRdgs.AddItem (buffr)             'add line to display
      Loop Until InStr(1, buffr, "ime")

      Call viClose(la)
      Call viClose(VISAsesn)

End Sub
```

e. VEE ASCII example



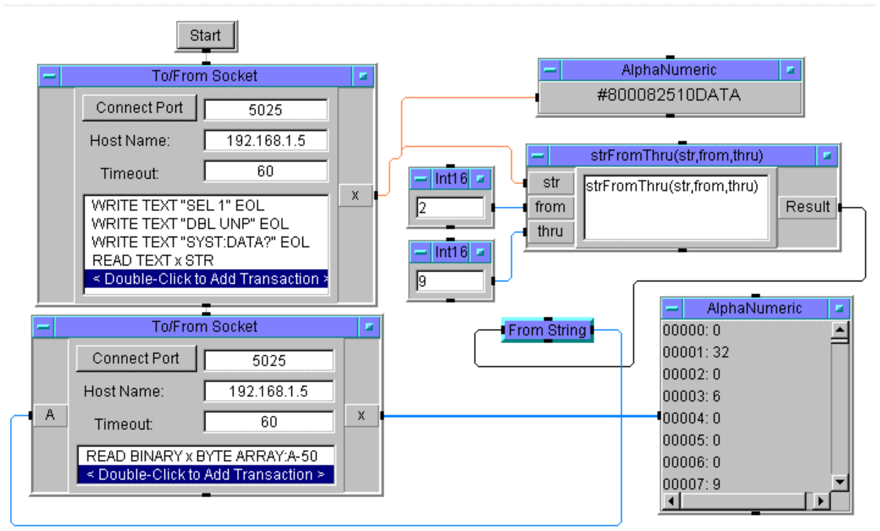
```
To/From Socket
Connect Port   5025
Host Name:     192.168.1.5
Timeout:       60
WRITE TEXT "SYST:PRINT? ALL" EOL
< Double-Click to Add Transaction >
```

```
To/From Socket
Connect Port   5025
Host Name:     192.168.1.5
Timeout:       60
READ TEXT x STR
< Double-Click to Add Transaction >
```

```
If/Then/Else
A=="&k0S"
```

```
AlphaNumeric
0186: -1867  000D -7.468 us
0187: -1866  000D -7.464 us
0188: -1865  000D -7.460 us
0189: -1864  000D -7.456 us
0190: -1863  000D -7.452 us
0191: -1862  000D -7.448 us
0192: -1861  000E -7.444 us
0193: -1860  000E -7.440 us
0194: -1859  000E -7.436 us
0195: -1858  000E -7.432 us
0196: -1857  000E -7.428 us
0197: -1856  000E -7.424 us
```

f. VEE binary example



```
To/From Socket
Connect Port   5025
Host Name:     192.168.1.5
Timeout:       60
WRITE TEXT "SEL 1" EOL
WRITE TEXT "DBL UNP" EOL
WRITE TEXT "SYST:DATA?" EOL
READ TEXT x STR
< Double-Click to Add Transaction >
```

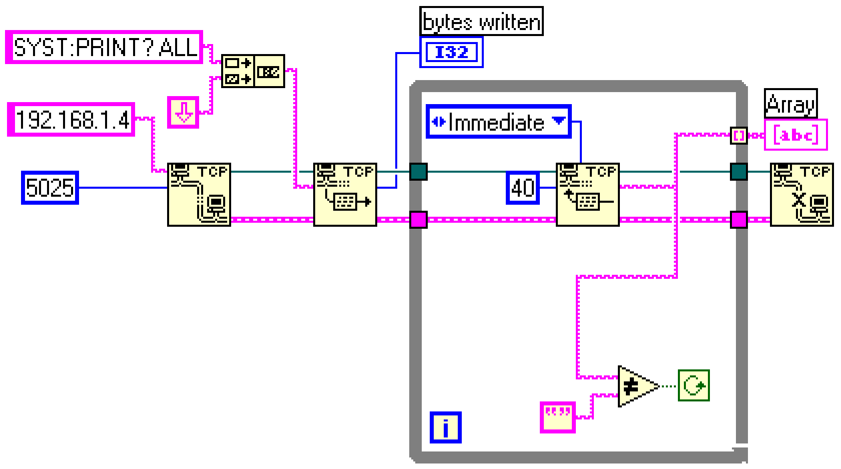
```
To/From Socket
Connect Port   5025
Host Name:     192.168.1.5
Timeout:       60
READ BINARY x BYTE ARRAY:A-50
< Double-Click to Add Transaction >
```

```
AlphaNumeric
#800082510DATA
```

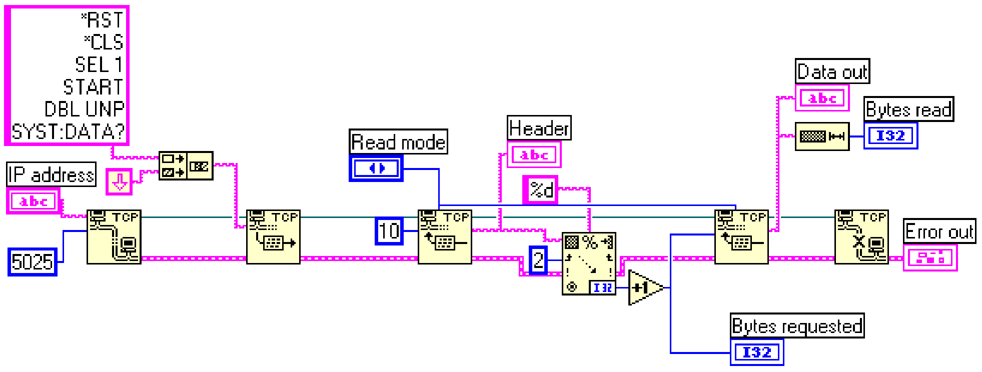
```
Int16  2
Int16  9
strFromThru(str,from,thru)
```

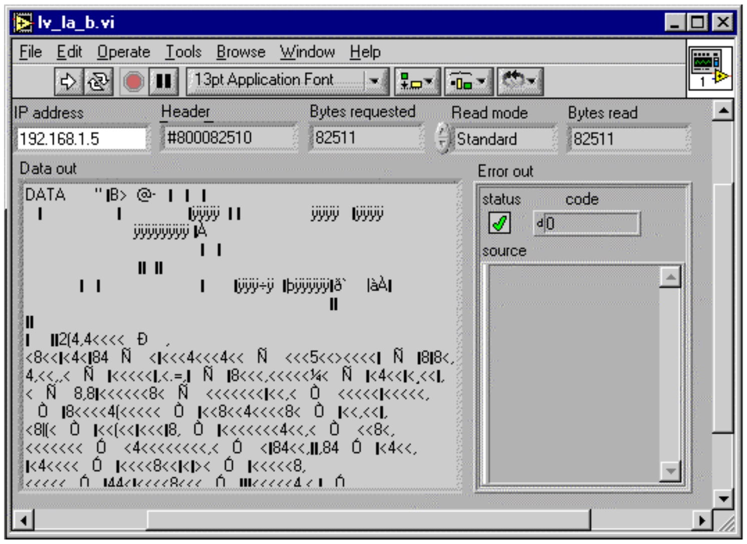
```
AlphaNumeric
00000: 0
00001: 32
00002: 0
00003: 6
00004: 0
00005: 0
00006: 0
00007: 9
```

g. LabVIEW ASCII example



h. LabVIEW binary example





IP address: 192.168.1.5
Header: #800082510
Bytes requested: 82511
Read mode: Standard
Bytes read: 82511

Data out

```
DATA    "|B> @-  |  |  |
 |          |         |ÿÿÿÿ ||          ÿÿÿÿ  |ÿÿÿÿ
                ÿÿÿÿÿÿÿÿÿ |À
                           |  |
                  || ||
        |  |            |    |ÿÿÿ÷ÿ |þÿÿÿÿÿ|ð`   |àÀ|
                                            ||
||
|   ||2(4,4<<<<   Ð   ,
<8<<|<4<|84  Ñ   <|<<<4<<<4<<  Ñ   <<<5<<><<<<|  Ñ  |8|8<,
4,<<,,<  Ñ  |<<<<<|,<.=,|  Ñ  |8<<<,<<<<<¼<  Ñ  |<4<<|<,<<|,
<  Ñ  8,8|<<<<<<8<  Ñ   <<<<<<<|<<,<  Ô   <<<<<|<<<<<,
  Ô  |8<<<<4(<<<<<  Ô  |<<8<<4<<<<8<  Ô  |<<,<<|,
<8||<  Ô  |<<(<<|<<<|8,  Ô  |<<<<<<<4<<,<  Ô   <<8<,
<<<<<<<  Ô   <4<<<<<<<<,<  Ô   <|84<<,||,84  Ô  |<4<<,
|<4<<<<  Ô  |<<<<8<<|<|><  Ô  |<<<<<8,
<<<<<  Ô  |44<|<<<<8<<<  Ô  |||<<<<<4<|  Ô
```

Error out
status: ✓   code: 0
source

12

i. LabWindows example

```
#include <stdio.h>
#include <tcpsupp.h>
#include <utility.h>
#include <ansi_c.h>

int CVICALLBACK callbk(unsigned handle,
                                        int event,
                                        int error,
                                        void *callbackData);

void main ()
{
        int status,i,sentinel;
        char hostname[30] = {"156.140.153.48"};
        int portNum = 5025;
        static unsigned TCPsession;
        char tempBuf[5] = {" "};
        char lastBuf[5];

   /* Attempt to connect to TCP server... */

        status =  ConnectToTCPServer (&TCPsession,
                                                portNum,
                                                hostname,
                                                callbk,
                                                NULL,
                                                5000);
        Delay(1);

        /* Talk to instrument...*/

   status = ClientTCPWrite (TCPsession,
                                        "SYST:PRINT? ALL\n",
                                        16,2000);

do
{

        status = ClientTCPRead (TCPsession,
                                        tempBuf,
                                        1,
                                        4000);
        if(strstr(lastBuf, "S") && strstr(tempBuf, "\n"))
                sentinel = -1;
```

```
            lastBuf[0] = tempBuf[0];

            printf("%s", tempBuf);

}while(sentinel != -1);

   /* Disconnect from the TCP server */

   status = DisconnectFromTCPServer (TCPsession);


   /* Close console */

   printf("\nPress ENTER to End");
   getchar();


}

/* Callback function definition */

int CVICALLBACK callbk (unsigned handle,
                        int event,
                        int erCode,
                        void *callbackData)
{
  //Develop callback code here
  return 1;
};
```

II. 16505, 16600 and 16700 series

Programming the 16505, 16600 and 16700 series requires a completely different approach. While previous logic analyzers had a command parser that accepted SCPI commands to program practically every operation of the logic analyzer, the new paradigm introduced with these analyzers follows a "load, run, save" algorithm.

In this approach, the programmer creates a configuration file, manually, from the front panel of the logic analyzer. This file is loaded programmatically at the appropriate time and the defined acquisition is started. When the data has been acquired, it can be saved to a file in one of three formats: internal, fast-binary and ASCII.

The internal format file can be saved for the purpose of re-loading, but it cannot be parsed nor analyzed outside of the logic analyzer.

The file format for the fast-binary file is described in the online help. This file can be parsed, displayed and analyzed. Tools are available for handling this format, including Intuilink, which allows the user to import the data directly into MS Word or Excel.

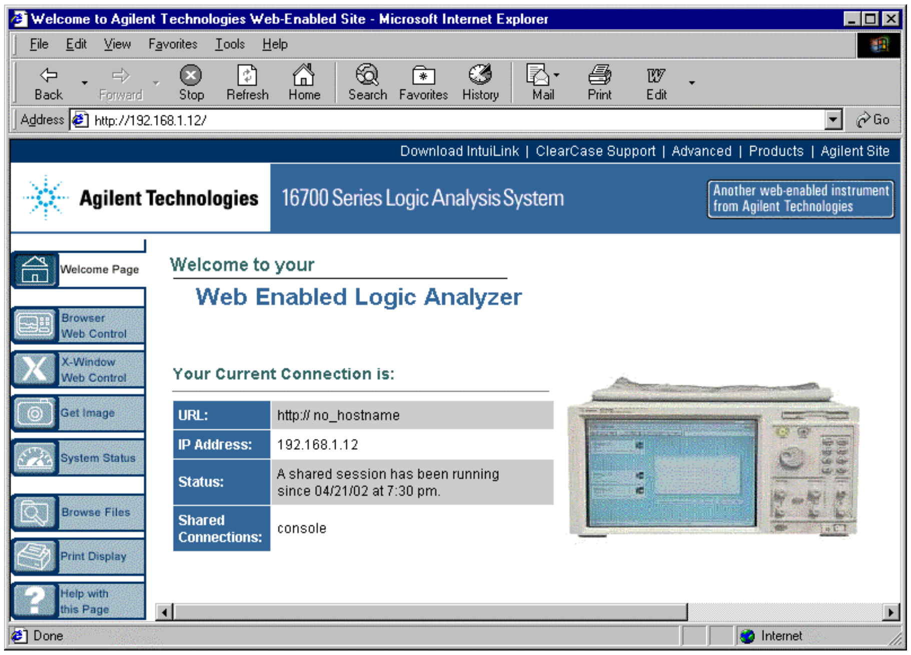
The ASCII format file can be imported into MS Word or Excel without further parsing.

With this approach to programming, file transfer is of key importance. In addition there are a few ways to control the logic analyzer from a remote location. Other ways of downloading data include: downloading directly from the analyzer module and downloading from the lister.

A. File transfer

1. HTTP

The 16700 series has a built-in web server, which makes available all files in the /logic directory and its subdirectories. Enter the hostname or IP address in your browser as a URL. You will be served a web page similar to this:

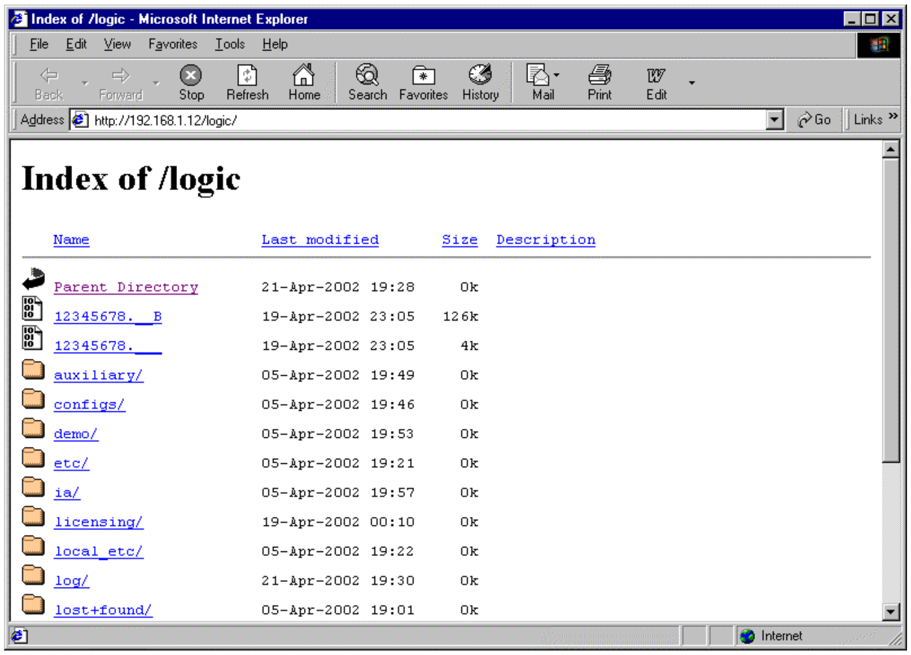

Welcome to Agilent Technologies Web-Enabled Site - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Address http://192.168.1.12/ Go
Download IntuiLink | ClearCase Support | Advanced | Products | Agilent Site
Agilent Technologies
16700 Series Logic Analysis System
Another web-enabled instrument from Agilent Technologies
Welcome Page
Browser Web Control
X-Window Web Control
Get Image
System Status
Browse Files
Print Display
Help with this Page
Done
Internet

Welcome to your
Web Enabled Logic Analyzer

Your Current Connection is:

URL:	http:// no_hostname
IP Address:	192.168.1.12
Status:	A shared session has been running since 04/21/02 at 7:30 pm.
Shared Connections:	console

After reaching the logic analyzer's main web page, click on "Browse Files." The analyzer will then serve a page like the one pictured below.


Index of /logic - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Address http://192.168.1.12/logic/
Go Links
Internet

Index of /logic

```
    Name                  Last modified       Size  Description
Parent Directory          21-Apr-2002 19:28     0k
12345678.__B              19-Apr-2002 23:05   126k
12345678.___              19-Apr-2002 23:05     4k
auxiliary/                05-Apr-2002 19:49     0k
configs/                  05-Apr-2002 19:46     0k
demo/                     05-Apr-2002 19:53     0k
etc/                      05-Apr-2002 19:21     0k
ia/                       05-Apr-2002 19:57     0k
licensing/                19-Apr-2002 00:10     0k
local_etc/                05-Apr-2002 19:22     0k
log/                      21-Apr-2002 19:30     0k
lost+found/               05-Apr-2002 19:01     0k
```

2. FTP

The 16700 series also has an FTP server which allows access to the files found in the /logic directory and its subdirectories. This FTP server accepts guest logins from the user "ftp" with the password "ftp."

3. NFS

The 16700 series has both an NFS server and an NFS client. Consequently, the /logic directory (which has been exported) can be mounted from a workstation that has an NFS client. In like manner, the 16700's file system can mount an NFS file system on a remote server. The directory to be mounted must first be exported with appropriate permissions. To export a directory you can create the /etc/exports file and include a line that names the directory to be exported and the permissions to grant.

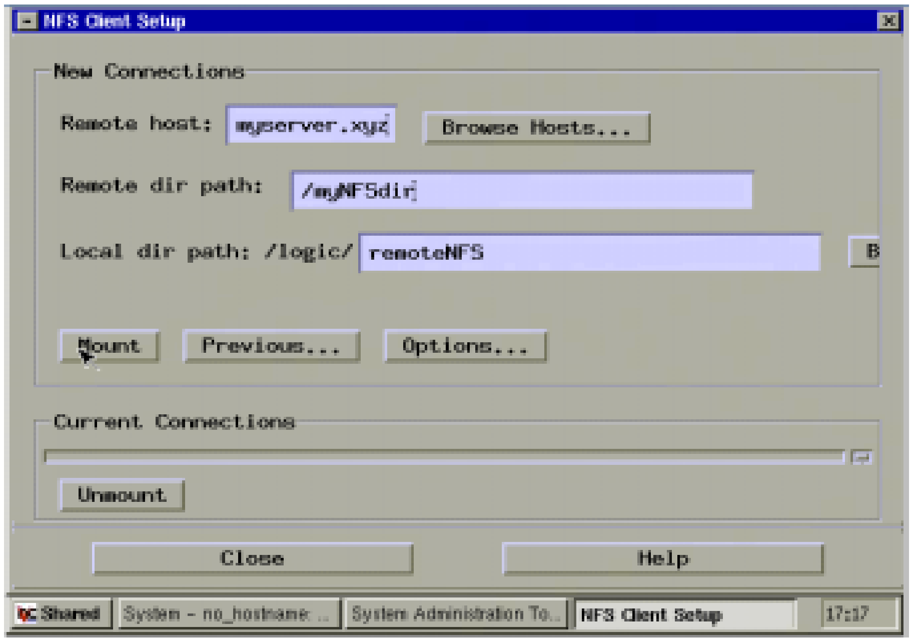
Example exports file entry: /myNFSdir analyzer1(rw),analyzer2(ro)

This entry gives analyzer1 read and write permission to files in /myNFSdir directory. It also gives analyzer2 read-only permission to files in /myNFSdir.

The /etc/exports file on the logic analyzer has the single-line simple entry "/logic" to indicate that all permissions are granted to the world for files in the /logic directory and its subdirectories.

a. Mount remote system from logic analyzer

To mount a remote NFS file system, go to the logic analyzer's System window, click on the System Administration icon (the one with the screwdriver and wrench), then click on Mount NFS filesystem… Fill out the required fields and click on Mount.


NFS Client Setup

New Connections

Remote host: myserver.xyz

Remote dir path: /myNFSdir

Local dir path: /logic/ remoteNFS

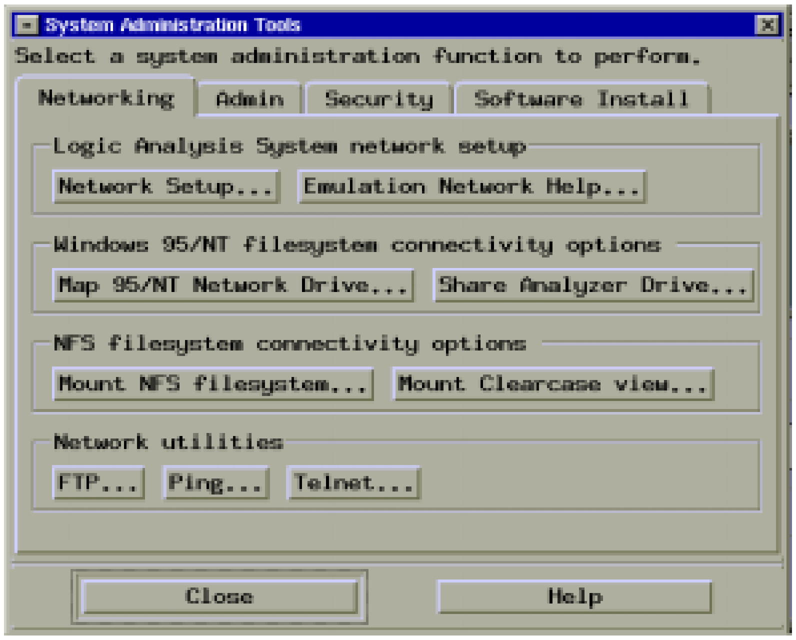
Current Connections

b. Mount logic analyzer from remote system

In order to mount the exported /logic directory, you must have an NFS client. In addition to HP-UX, Solaris and Linux, for which this feature is readily available, NFS clients are available also for the PC running Windows.

4. Clearcase

As indicated by the logic analyzer's online help system, "Using both the ability of Rational ClearCase to export views, and the logic analysis system to NFS mount external file systems, you are now able to access versioned source code to perform source correlation, use symbols, and view data in trace listings." Go to the logic analyzer's System window, click on the System Administration icon (the one with the screwdriver and wrench), then click on Mount ClearCase view… For further instructions on mounting a ClearCase view, please refer to the logic analyzer's online help.


System Administration Tools
Networking
Admin
Security
Software Install
Logic Analysis System network setup
Network Setup...
Emulation Network Help...
Windows 95/NT filesystem connectivity options
Map 95/NT Network Drive...
Share Analyzer Drive...
NFS filesystem connectivity options
Mount NFS filesystem...
Mount Clearcase view...
Network utilities
FTP...
Ping...
Telnet...
Close
Help

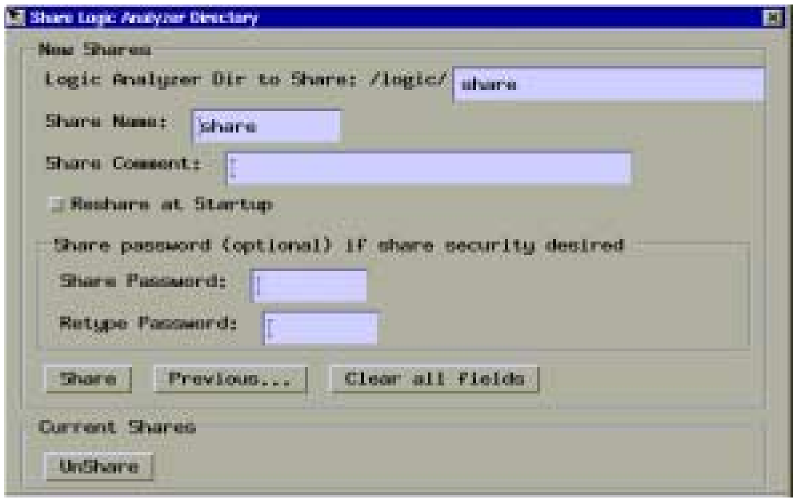
Select a system administration function to perform.

5. Map Windows drive

The 16700 series also has the capability of sharing a folder so that a Windows system can map to it. Likewise, it can map to a Windows remote shared folder. When this is done, the remote folder appears as part of the local file system.

a. Share analyzer folder

In the System Administration window, click on Share Analyzer Drive… Fill in required fields and click on Share.



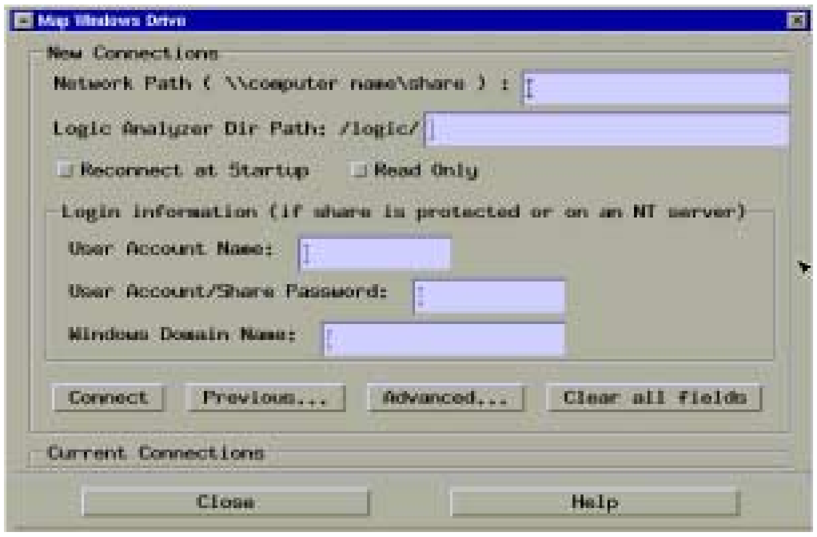
Logic Analyzer Dir to Share: /logic/ share

Share Name: share

Share Comment:

b. Share PC folder

Once you have shared the PC folder with the appropriate permissions, go to the System Administration window of the logic analyzer and click on Map 95/NT Network Drive… Fill in the required fields and click on Connect.



B. Control

The display of the logic analyzer is exported so that it can be viewed in X-Windows, a browser such as IE or Netscape and a VNC client. This view is interactive, allowing the use of a mouse to control the analyzer remotely. RPI and Intuilink are means of remote control which can be used in the automated test environment.

1. X-Windows

If you have an X-server running on your PC or workstation, clicking on "X-Window Web Control" while connected to the logic analyzer's web page, and then clicking on "Join Session," will "serve" the interactive display made available through VNC.

2. Web Browser

Clicking on "Browser Web Control" while connected to the logic analyzer's web page will import an interactive display into the browser. This method is somewhat slower than the X-Window Web Control mentioned above.

3. VNC

VNC (Virtual Network Computing) is a remote display system, which allows one to interact with a "desktop" from anywhere on the internet. You can download this software from the AT & T web site, use it, and redistribute it under the terms of the GNU Public License. When you start the VNC viewer you will have to enter the IP address or hostname of the logic analyzer followed by the display number (for example: 192.168.1.12:1). VNC is faster than the browser and on a par with X-Windows.

X-Windows, the browser and VNC allow a person to interact with the logic analyzer's front panel, remotely. However, if you need to automate control or download data, other tools must be used:

4. RPI

The Remote Programming Interface is a process that listens to port 6500 on the logic analyzer. It is not a SCPI command parser as implemented in the 16500 series, but is capable of receiving ASCII commands to load a previously saved configuration file, make minor modifications, start an acquisition and download the acquired data. It can only interact with objects that are already in the "Workspace." For this reason, it is necessary to load a configuration file instead of writing a program "from scratch."

The following examples are designed for small acquisitions (4096 states). They use the lister as the source of downloadable data. For larger acquisitions, it is advisable to use the file-out tool and save the data to a fast-binary file. This can be transferred through FTP rather quickly.

Example RPI programs:

a. C example

```
#include<stdio.h>
#include<windows.h>
#include"visa.h"


void main()
{

/*************************************************/
/**************** Declare variables ****************/
/*************************************************/

        ViSession sesn,la;
        FILE *mydata;

        // The following strings must be large enough for
        // amount of data to be downloaded, or the
        // program may crash

        char modules[500]={" "};
        char listing[400000]={" "};
```

```
/**************************************************/
/****** Open File---Open VISA and Instrument sessions *****/
/**************************************************/

        mydata = fopen("c:\\mydata.txt","w");
        viOpenDefaultRM (&sesn);
        viOpen(sesn, "TCPIP0::192.168.1.12::6500::SOCKET",
                                                VI_NULL,
                                                VI_NULL,
                                                &la);

/***************************************************/
/************ Get list of modules in mainframe ************/
/***************************************************/

        viPrintf(la,"modules\n");
        viScanf(la,"%t", &modules);

        printf("%s\n",modules);

/***************************************************/
/**************** Load configuration file ****************/
/***************************************************/

        viPrintf(la,"config -l /logic/myconfig.___\n");

/***************************************************/
/**************** Start acquisition ****************/
/***************************************************/

        viPrintf(la,"start\n");
        viPrintf(la,"wait -complete\n");
        Sleep(10000);

/***************************************************/
/************* Download data and save to file *************/
/***************************************************/

        viPrintf(la,"listing -n Listing<1> -d -l all -r all\n");
        Sleep(40000);
        viScanf(la,"%t", &listing);
        fprintf(mydata,"%s\n", listing);
```

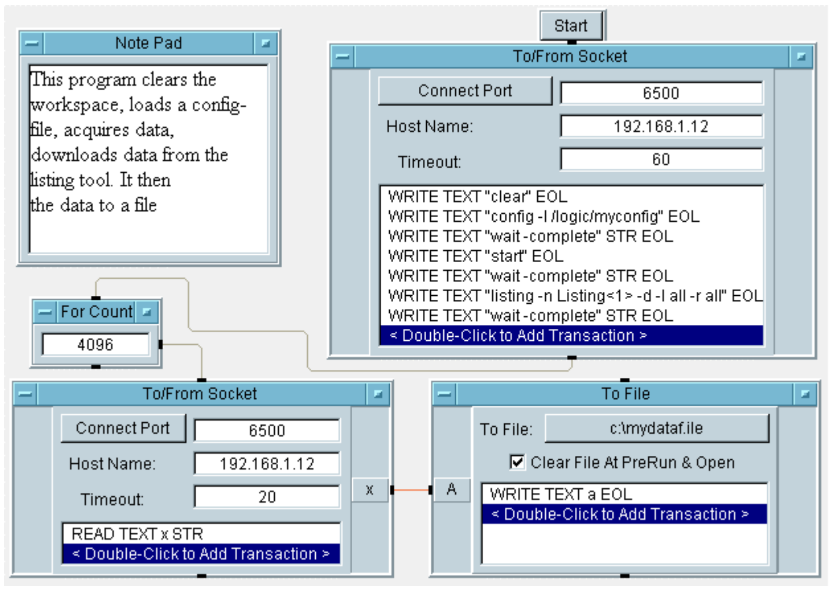
```
/***************************************************/
/****** Close File--Close Instrument and VISA sessions ******/
/***************************************************/

        fclose(mydata);
        printf("Press any key to continue\n");
        viClose(la);
        viClose(sesn);
}
```

b. VB example

```
Dim VISAsesn As Long
Dim la As Long
Dim moduleBuf As String * 1000
Dim rdgsBuf As String
Dim tmpChar As String * 1
Dim retcnt As Long
Dim i As Integer


Private Sub cmdClear_Click()
   Call viWrite(la, "clear" & Chr$(10), 6, retcnt)
End Sub

Private Sub cmdDownload_Click()

   Call viWrite(la, "listing -n Listing<1> -d -l all -r all" & Chr$(10), 39, retcnt)

   For i = 1 To 4096
      rdgsBuf = " "
      Do
         Call viRead(la, tmpChar, 1, retcnt)
         rdgsBuf = rdgsBuf & tmpChar
      Loop Until (tmpChar = Chr$(13))
      LstRdgs.AddItem (rdgsBuf)
   Next i

End Sub

Private Sub cmdLoadConfig_Click()
    Call viWrite(la, "config -l /logic/myconfig.___" & Chr$(10), 30, retcnt)
End Sub
```

```
Private Sub cmdQuit_Click()
   Call viClose(la)
   Call viClose(VISAsesn)
   End
End Sub

Private Sub cmdStart_Click()
    Call viWrite(la, "start" & Chr$(10), 6, retcnt)
    Call viWrite(la, "wait -complete" & Chr$(10), 15, retcnt)
End Sub

Private Sub Form_Load()

   Call viOpenDefaultRM(VISAsesn)
   Call viOpen(VISAsesn, "TCPIP0::192.168.1.12::6500::SOCKET", VI_NULL,
                                                                        VI_NULL, la)
   Call viSetAttribute(la, VI_ATTR_TERMCHAR, 10)
   Call viWrite(la, "modules" & Chr$(10), 8, retcnt)
   Call viRead(la, moduleBuf, 1000, retcnt)
   txtModules.Text = moduleBuf

End Sub
```

c. VEE example



Note Pad:
This program clears the workspace, loads a config-file, acquires data, downloads data from the listing tool. It then the data to a file

To/From Socket: Connect Port 6500; Host Name: 192.168.1.12; Timeout: 60

```
WRITE TEXT "clear" EOL
WRITE TEXT "config -l /logic/myconfig" EOL
WRITE TEXT "wait -complete" STR EOL
WRITE TEXT "start" EOL
WRITE TEXT "wait -complete" STR EOL
WRITE TEXT "listing -n Listing<1> -d -l all -r all" EOL
WRITE TEXT "wait -complete" STR EOL
< Double-Click to Add Transaction >
```

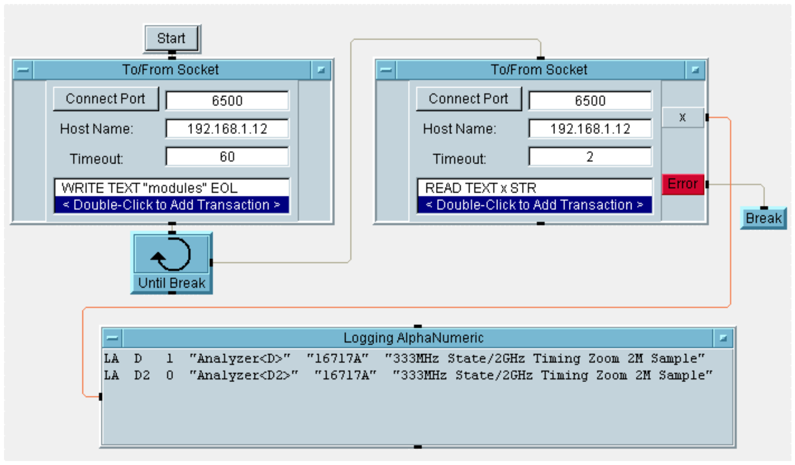
For Count: 4096

To/From Socket: Connect Port 6500; Host Name: 192.168.1.12; Timeout: 20

```
READ TEXT x STR
< Double-Click to Add Transaction >
```

To File: c:\mydataf.ile; ☑ Clear File At PreRun & Open

```
WRITE TEXT a EOL
< Double-Click to Add Transaction >
```

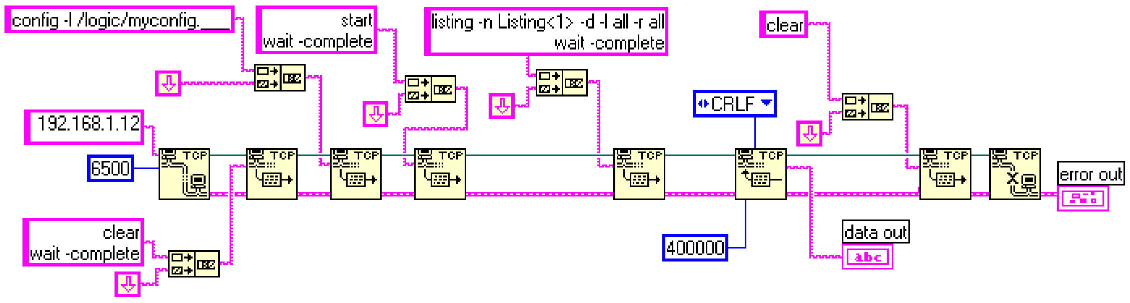


Connect Port 6500
Host Name: 192.168.1.12
Timeout: 60
WRITE TEXT "modules" EOL
< Double-Click to Add Transaction >

Connect Port 6500
Host Name: 192.168.1.12
Timeout: 2
READ TEXT x STR
< Double-Click to Add Transaction >

```
LA  D   1  "Analyzer<D>"  "16717A"  "333MHz State/2GHz Timing Zoom 2M Sample"
LA  D2  0  "Analyzer<D2>"  "16717A"  "333MHz State/2GHz Timing Zoom 2M Sample"
```

d. LabVIEW example



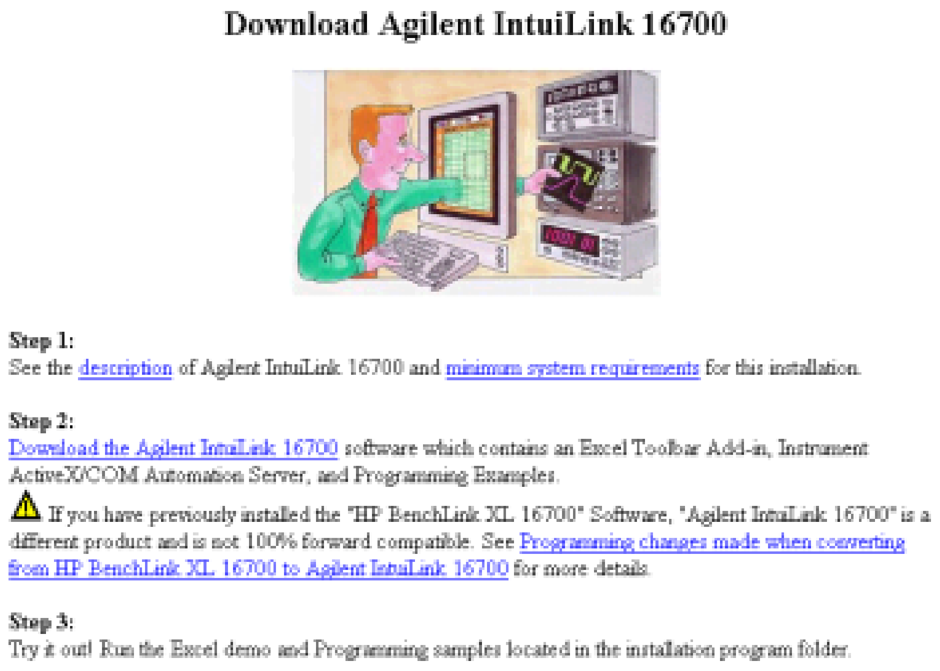
e. LabWindows example

```
#include <tcpsupp.h>
#include<stdio.h>
#include<utility.h>

int CVICALLBACK callbk (unsigned handle,
                                        int event,
                                        int errCode,
                                        void *callbackData);

void main()
{
int status;
static unsigned int TCPsession;
char tempBuf[256];
ConnectToTCPServer (&TCPsession,
                                        6500,"192.168.1.12",
                                        callbk, NULL, 5000);
Delay(2);

//*******************************************

ClientTCPWrite(TCPsession,
                                "modules\n",
                                8,2000);
Delay(4);

ClientTCPRead(TCPsession,tempBuf,256,2000);
printf("%s",tempBuf);

//*******************************************

DisconnectFromTCPServer(TCPsession);

printf("Press ENTER to continue");
getchar();
}

int CVICALLBACK callbk (unsigned handle,
                                int event,
                                int errCode,
                                void *callbackData)
{
//Callback function must be developed
return 1;
}
```

5. Intuilink

Intuilink, an ActiveX/COM-based driver object, can simplify the task of controlling the logic analyzer remotely, from a Windows environment. Intuilink only works in Windows. Its objects create the RPI commands necessary to interact with the logic analyzer, but this is totally transparent to the user.

You can obtain Intuilink by pointing your browser to the logic analyzer's built-in web page. The first selection in the top bar is "Download Intuilink." This will take you to the following page:



Download Agilent IntuiLink 16700

Step 1:
See the description of Agilent IntuiLink 16700 and minimum system requirements for this installation.

Step 2:
Download the Agilent IntuiLink 16700 software which contains an Excel Toolbar Add-in, Instrument ActiveX/COM Automation Server, and Programming Examples.

⚠ If you have previously installed the "HP BenchLink XL 16700" Software, "Agilent IntuiLink 16700" is a different product and is not 100% forward compatible. See Programming changes made when converting from HP BenchLink XL 16700 to Agilent IntuiLink 16700 for more details.

Step 3:
Try it out! Run the Excel demo and Programming samples located in the installation program folder.

On this page, you will be able to read a more detailed description of Intuilink. You will also be able to download the software. For the sake of maximum compatibility, it is advisable to download Intuilink from the analyzer to be controlled, and thus insure that the version of Intuilink is the one intended for the operating system installed on that analyzer. After installing this software, you will find example programs for VEE, VB, C++, Excel and LabVIEW.

Conclusion:

We have seen how to connect to logic analyzers to enable downloading data from the analyzers. In addition, importing the analyzer's display to a browser or X-Windows server will allow us to remotely control the analyzer interactively as if we were at its front panel. It is now left to the reader to carry this one step further. With the information found in the programming manuals, the programmer can automate the remote control process by writing programs to send the appropriate commands in a timely fashion.