SRInstComms™

ActiveX Control for *SIGNAL RECOVERY* Instruments

Instruction Manual

197319-A-MNL-A

Trademarks

Microsoft is a registered trademark; Windows and NT are trademarks of Microsoft Corporation.
IBM is a registered trademark of International Business Machines Corporation.
NI is a registered trademark of National Instruments Corporation.

AMETEK® and the AMETEK® and logos are registered trademarks, and SRInstComms a trademark of AMETEK, Inc. Other product and company names mentioned are trademarks or trade names of their respective companies.

Company Names

SIGNAL RECOVERY is part of Advanced Measurement Technology, Inc, a division of AMETEK, Inc. It includes the businesses formerly trading as EG&G Princeton Applied Research, EG&G Instruments (Signal Recovery), EG&G Signal Recovery and PerkinElmer Instruments (Signal Recovery)

Table of Contents

Chapter One, Introduction

Chapter Two, Installation & Initial Checks

Chapter Three, Operation

Chapter Four, Demonstration Files

Introduction

Chapter 1

1.1 Overview

The SRInstComms ActiveX Control and Software Development Kit (SDK) is designed to allow easy control of any compatible ***SIGNAL RECOVERY*** product from applications software written in languages that provide ActiveX control containers, such as LabVIEW, Visual Basic, Visual C++, Visual Basic for Applications (VBA) as found in Excel, Word, Access, Outlook, and VBScript as used in Internet Explorer and other Microsoft products.

The control eliminates the need for users to write the low-level code needed to send commands to and receive responses from the instrument, allowing them to concentrate on developing the higher level program to run their experiments. Typical applications include:

- Creating Excel spreadsheets that record instrument outputs as a function of time with direct graphical display in a chart
- Creating simple Visual Basic programs that allow single measurements to be taken and displayed in the correct units, e.g. measuring AC currents using a current transformer and presenting the result in amperes
- Test and measurement systems implement in LabVIEW where a ***SIGNAL RECOVERY*** product needs to be controlled at the same time as instruments from different suppliers.
- Measurement system using scripted web pages (HTML files) operated via Internet Explorer.

Full on-screen help included with the control, as well as examples of its use in LabVIEW, Visual Basic, VBScript, Visual C++ and Excel.

1.2 Changes from the PESRInstComms Control

The SRInstComms control supercedes the PESRInstComms control, which is now obsolete. The new control has the following changes when compared to the old one:

- The logo, dialog boxes and icons are updated to the new company name ***SIGNAL RECOVERY***.
- The array parameters returned by the GetActiveInstruments and WriteReadSRInstrument function have changed to the Variant data type. The reason for making this change is that it allows the control to be used with VBScript, as implemented in Internet Explorer. Hence it becomes possible to use the control in active HTML files that can be operated from a browser window.
- The new control draws itself correctly so that the proper control icon is shown.

- The old control had a memory leak that meant if it was called repeatedly in a loop it took up more and more of the computer's resources. This has been fixed.
- During a scan for instruments the progress dialog box is now updated correctly.
- The routine used for handling binary dumps of data curves has been improved so that it makes fewer demands on system resources.

Installation & Initial Checks

Chapter 2

2.1 Introduction

This chapter describes how to install the SRInstComms™ ActiveX Control and supporting software on a personal computer.

2.1.01 Media

The SRInstComms™ ActiveX Control software is supplied on two 1.44 Mbyte 3.5" floppy disks complete with installation program.

2.1.02 Product Registration

The first time the SRInstComms control is used following installation it will automatically attempt to connect to our website and display a form for you to enter the relevant details to register your purchase. If the computer has browser software (Internet Explorer or Netscape Navigator) but cannot establish a link to the site, perhaps because there is no network connection, then you will receive a "404 Page Not Found" or similar error and you should simply close the browser window to continue, and perform manual registration (see below) later. Similarly, if browser software is not found then a message is displayed inviting you to perform the registration manually.

Manual registration is possible simply by visiting **www.signalrecovery.com/register** and clicking on the relevant link.

The control continues to attempt automatic registration until the **CreateSRCommsRegKeys** function has been run.

Registration will ensure that you are kept informed of software enhancements and developments as they become available. In addition, should you need to contact us for support we will check your details against those we have on file, and, if your copy of the software is not registered then we will not be able to help you.

2.1.03 Requirements

The SRInstComms™ ActiveX Control software requires:-

- A ***SIGNAL RECOVERY*** model 5105, 5106, 5109, 5110, 5209, 5210, 5302, 7220, 7220BFP, 7225, 7225BFP, 7260, 7265 or 7280 lock-in amplifier, a model 5113 preamplifier or a model 7310 noise rejecting voltmeter. Note that it is not designed for use with the models 5102, 5104, 5205, 5206, 5207, 5208 or 5301/5301A lock-in amplifiers. The control allows up to ten of these compatible instruments to be operated simultaneously.

- An IBM PC or 100% compatible personal computer with a 166 MHz Pentium or better processor, hard disk, 3.5" floppy disk drive and Microsoft Windows 95/98/ME/2000/XP or NT version 4.0 or later. 2 Mbyte free disk space is required for the full installation.

Either

- A National Instruments GPIB (IEEE-488) Interface Card type PCI-GPIB or PCMCIA-GPIB fitted into the computer and suitable GPIB cables to connect this to the instruments.

and/or

- A free RS232 port (COM1, COM2, COM3 or COM4) on the computer and an RS232 interconnecting cable compatible with both the computer port and the connected instrument. The connectors required on this cable depend on the type of connector used on the computer's RS232 port and on the instrument, as follows:-

Computers fitted with 9 pin RS232 connectors:-
a) Models 5105 and 5106 Lock-in Amplifiers
9-pin subminiature D type female connector to 9-pin subminiature D type female connector modem eliminator cable, ***SIGNAL RECOVERY*** Part Number C01001. This cable is included with each unit and is the same as that used when operating them with the software package supplied with these instruments.

b) Models 5109, 5110, 5209, 5210 and 5302
9-pin subminiature D type female connector to 25-pin subminiature D type male connector modem eliminator cable, ***SIGNAL RECOVERY*** Part Number C01002.

c) Models 7220, 7220BFP, 7225, 7225BFP, 7260, 7265, 7280 and 7310
9-pin subminiature D type female connector to 9-pin subminiature D type male connector modem eliminator cable, ***SIGNAL RECOVERY*** Part Number C01003.

d) Model 5113
9-pin subminiature D type female connector to 25-pin subminiature D type male connector. ***Note that this cable is not a modem eliminator type.***

Computers fitted with 25-pin RS232 connectors
Although it is possible to construct a dedicated cable to suit computers fitted with 25-pin RS232 connectors, it is suggested that one of the four cables listed above is used with a 25-pin female to 9-pin male adapter, ***SIGNAL RECOVERY*** Part Number K02001, fitted at the computer end.

With the exception of the cable for the model 5113, the listed items are available from ***SIGNAL RECOVERY*** but users with reasonable practical skills may construct their own RS232 cables if they wish. Connection diagrams for these are given in Appendix A of this manual.

2.1.03 Interface Selection

The models 5105, 5106 and 5113 are only fitted with an RS232 interface port, and hence can only be operated via this interface. All the other supported instruments are fitted with both GPIB and RS232 interfaces, in which case it is necessary to decide which one to use. Generally speaking, operation via the GPIB is better and is recommended if large numbers of data points are to be acquired, but obviously involves the purchase of a suitable GPIB interface card. RS232 operation offers a cheaper route to instrument control, but at the penalty of slightly degraded performance.

2.1.04 RS232 Daisy Chain

The control does not support the “daisy chain” connection method whereby several compatible instruments can be operated via a single RS232 computer port. Consequently no connection should be made to the connectors marked **AUX RS232** or **Port 2** on the instrument(s).

Next, follow the relevant procedures in section 2.2 or 2.3 below to connect the instrument(s) to the computer.

2.2 GPIB Installation

1) Turn off the computer and instrument(s) and disconnect them from their power source.

2) In the case of the PCI-GPIB, remove the computer's cover to gain access to the expansion slots. The PCMCIA-GPIB simply plugs into a spare PCMCIA slot in the computer, which for this card is normally a notebook or laptop.

3) Install the National Instruments GPIB interface card in accordance with the hardware installation instructions supplied by National Instruments.

4) Replace the computer's cover.

5) Connect the GPIB cable to the interface card and to the GPIB connector (marked **GPIB**) on the rear panel of the first instrument. Carefully tighten the fixing screws on the connectors. If using more than one instrument, connect a second GPIB cable from the GPIB connector on the first instrument to the GPIB connector on the second, and so on.

6) Reconnect both the computer and instrument(s) to their power source and turn them both on.

7) Install the National Instruments driver software in accordance with the software installation instructions supplied by National Instruments.

   Note: The SRInstComms™ control will not work unless this is done.

8) Note the software serial number and then insert the supplied SRInstComms™ Setup Disk 1 into the computer's disk drive (`a:` or `b:`) and from the Start Menu select the Run item.

9) Type `a:setup` and press the `RETURN` or `ENTER` key. Of course, if the computer's disk drive is drive `b:` then simply substitute the letter "`b`" for "`a`" in this line.

10) The installation program will now run, copying files to the computer's hard disk. Simply follow the on-screen instructions, changing disks if prompted to do so

11) On completion, the Windows desktop screen returns.

2.3 RS232 Installation

1) Turn off both the computer and instrument(s) and disconnect them from their power source.

2) Identify a spare RS232 port (COM1, COM2, COM3 or COM4) connector on the computer and connect the matching connector on the RS232 cable to it. If the cable being used has a 9-pin connector and the computer a 25-pin connector then an adapter (Part Number K02001) will be needed as well. Carefully tighten the fixing screws holding the connector in place.

3) Connect the other end of the RS232 cable to the rear-panel connector on the instrument, marked **RS232**, or in the case of the models 5105 and 5106, **Port 1**. Do not connect to the port marked **AUX RS232**, found on the models 7220, 7220BFP, 7225, 7225BFP, 7260, 7265, 7280 or 7310, or **Port 2** on the models 5105 and 5106.

4) If using several instruments, repeat sections 2) and 3) above for each instrument.

5) Reconnect both the computer and instrument(s) to their power source and turn them both on.

6) Note the software serial number and then insert the supplied SRInstComms™ Setup Disk into the computer's disk drive (`a:` or `b:`) and from the Start Menu select the Run item.

7) Type `a:setup` and press the `RETURN` or `ENTER` key. Of course, if the computer's disk drive is drive `b:` then simply substitute the letter "`b`" for "`a`" in this line.

8) The installation program will now run, copying files to the computer's hard disk. Simply follow the on-screen instructions, changing disks if required.

9) On completion, the Windows desktop screen returns.

2.4 Instrument Communications Settings

The control is designed to automatically identify compatible instruments that are connected and powered on. In order to do this, some of the instrument's communication settings need to be set to acceptable values, which for convenience have been chosen to be the "factory default" values, i.e. the settings used in new instruments supplied from the factory.

Before using the control it is recommended that the settings in the connected instrument(s) are reset to these values. With the 7220, 7220BFP, 7225, 7225BFP, 7260, 7265, 7280 and 7310 this can be done by using the Default Setting/Auto Default front panel control. With the models 5113, 5109, 5110, 5209, 5210 and 5302 use the information in their instruction manuals to set the controls as follows:-

- RS232: set to 9600 baud, 7 data bits, 1 stop bit, even parity, prompt on and echo on.

- GPIB: set to test echo off, <CR> (carriage return) terminator. Set the GPIB address as required, typically to 12 if using only one instrument.

The control will directly operate with the fixed communications settings used in the models 5105 and 5106.

If using several GPIB instruments set each of them to a different address. Note that the Default Setting/Auto Default function sets the address to 12.

2.5 Initial Checks

Once the control is installed and the instrument(s) connected, the easiest way to test that it is operating correctly is by using the compiled Visual Basic example, SRInstComms.exe, that by default is installed in the C:\Program Files\Signal Recovery\SRInstComms\VB6 and VBScript-HTML Demonstration\Visual Basic 6 Demonstration folder.

Use Explorer, browse to this file which has the ***SIGNAL RECOVERY***  icon, and double-click on the icon to start the program. Since the control is being used for the fist time, it will attempt to connect you to the relevant software registration page on the **www.signalrecovery.com** website by starting your default web browser. Enter the details requested and then close the broswer window to continue loading the control.

Click the OK button in the Information dialog to start the search for any connected instruments; when this completes the main dialog will be displayed. For example, if a single model 7225 was connected with its GPIB address set to "12" then this dialog would appear as in figure 2-1 below.

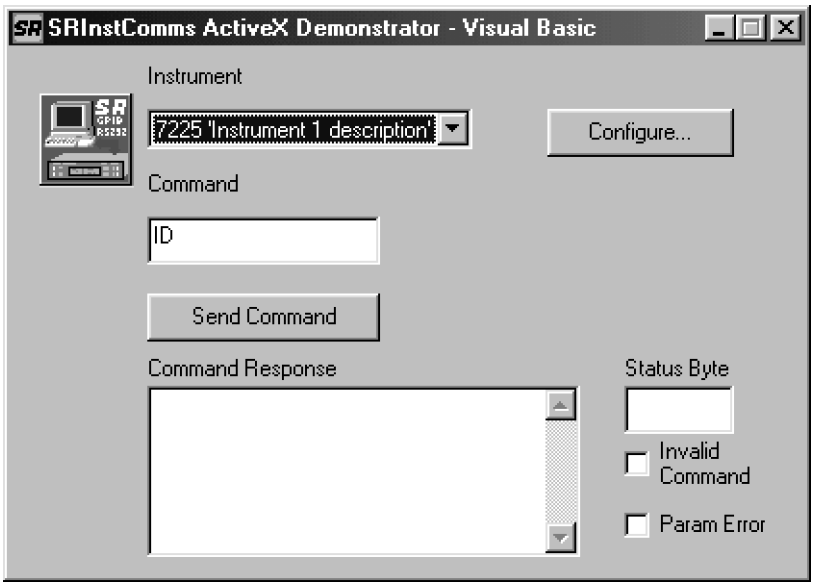

Figure 2-1, SRInstComms.exe Main Dialog

Click **Send Command**, which will send the command ID to the instrument. The response in the **Command Response** text box should then be the instrument's ID, in this case the digits "7225".

The **Instrument** list box shows the instrument being controlled. If only one instrument is connected then it has no effect, but if several units are connected then it is used to select which one of them is addressed when the **Send Command** button is clicked.

Click **Configure** to access the **Instrument Connections** dialog box, shown in figure 2-2 below.

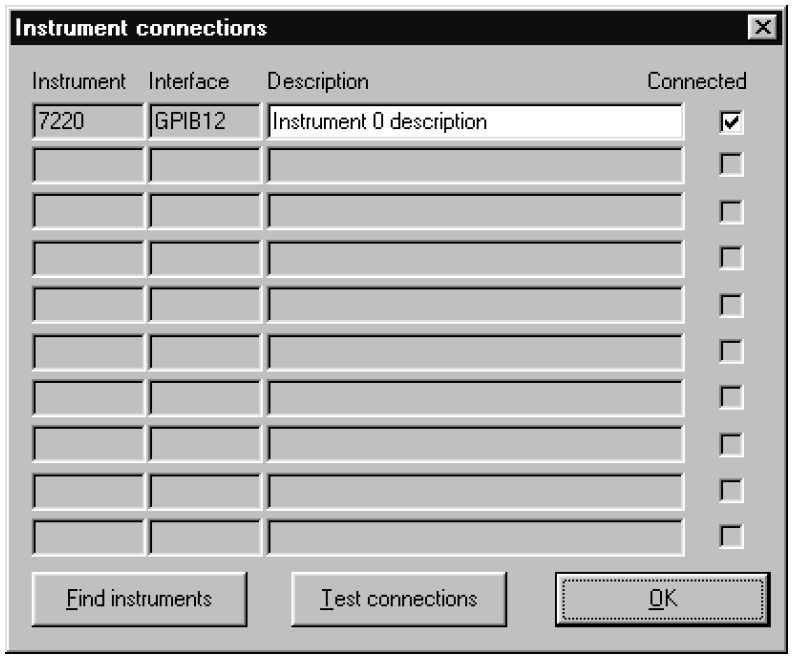

Figure 2-2, Instrument Connections Dialog Box

This dialog box lists all the connected instruments. If desired, the default instrument descriptions can be changed so that they are meaningful in your experiment. For example, descriptions such as "photocurrent", "sample voltage" or "temperature" could be used.

The **Connected** column on the right hand side shows whether the listed instrument is actually present and connected. The next time this dialog box is activated, the stored connection profile is reloaded and tested, and if an instrument is not found then the relevant connected box will be unchecked. If the instrument should be present, then make any necessary connections or turn it on and click **Test Connections** to re-test the connections.

The **Find Instruments** button clears all the details about the connected instruments from the system and initiates a new search for compatible units.

Click **OK** to return to the main dialog box.

Assuming that the SRInstComms.exe program works as described then you can be certain that the underlying SRInstComms.ocx ActiveX control has been installed correctly and is functioning properly.

Operation

Chapter 3

3.1 Introduction

This chapter describes how the SRInstComms ActiveX control operates and gives details of the methods it offers to control containers.

3.2 Technical Description

3.2.01 Overview

The SRInstComms ActiveX Control is designed to allow up to ten compatible ***SIGNAL RECOVERY*** products to be operated via their computer interfaces from a PC computer. It currently supports the models 5113, 5105, 5106, 5109, 5110, 5209, 5210, 5302, 7220, 7260, 7225, 7265, 7280 and 7310.

The control features:-

- Automatic testing for the presence of any ***SIGNAL RECOVERY*** products on the GPIB interface or RS232 ports COM1 to COM4.
- When using the RS232 interface, the Windows file handle used to communicate with the port can be held open while a session is in progress, preventing other programs from accessing the same port.
- Ability to communicate with up to ten different compatible instruments simultaneously.
- Automatic support for special commands offered by some of the instruments, such as binary curve dumps and the "star" command.

The control requires a **National Instruments** GPIB interface card, typically a PCI-GPIB or PCMCIA-GPIB, in order use the GPIB interface. It will **NOT** work with other manufacturer's cards.

3.2.02 Background

NOTE: The information in this section is provided for the benefit of advanced programmers who wish to understand the interaction of the control with the Windows registry. It is not necessary to understand this information in order to use the control.

The control uses the Windows registry to store a structure containing details of all the connected instruments, allowing it to easily re-establish communications with any instrument at any time. This structure contains details of ten instruments, numbered Instrument 0 to Instrument 9, and is saved under the HKEY_CURRENT_USER/Software/SIGNAL RECOVERY/SRComms key.

To view the registry, use the Windows utility Regedit. On the **Start** menu select **Run**

and type `regedit`, which will open an Explorer type window. To close Regedit simply click on the "x" in the top right hand corner or select Exit on the Registry menu.

WARNING: Users should be aware that the Windows registry contains information vital to the operation of the computer, and changes made using Regedit may cause it to stop working. Do not, therefore, make any changes to the registry unless you are absolutely certain of what you are doing.

A typical registry structure is shown in figure 3-1 below.

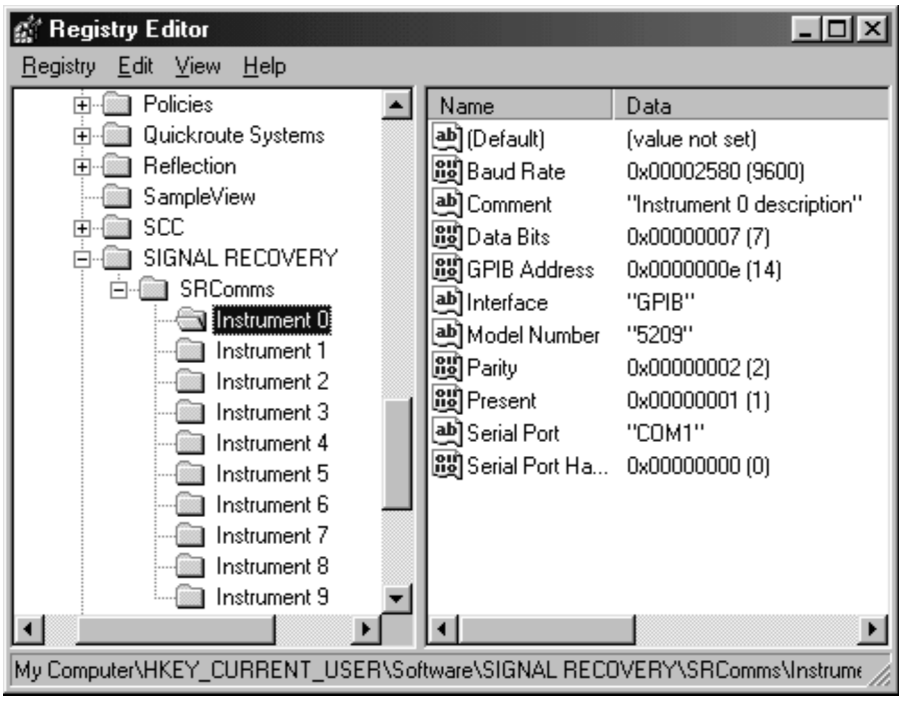

Figure 3-1, Typical Registry Structure

Within this structure, the subkeys under each instrument number have the following significance:

- (Default): key not used
- Baud Rate, valid only if an RS232 interface is being used for this instrument
- Comment, used to describe the instrument
- Data Bits, the number of data bits used for serial communications
- GPIB Address, valid only if the GPIB interface is being used for this instrument
- Interface, for which the only options are GPIB, COM1, COM2, COM3 and COM4

- Model Number, as reported by ID command. If this is a null then no instrument is connected
- Parity, the serial port parity
- Present, set to “1” (TRUE) if an instrument of the same model number as stored in the ID key is actually switched on and connected at the time a test is run to check the presence of instruments, “0” (FALSE) otherwise.
- Serial Port, the port to which this instrument is cabled
- Serial Port Handle, the file handle used to access the given serial port in the current session

The control provides a method to create a structure where this does not exist and populate it with default values. Other control methods will fail if the structure is not present.

Communications to and from the instruments are referenced by the instrument number, so for example with a single instrument the instrument number is always “0”.

3.3 Using the Control

3.3.01 Single Instrument Applications

In the simple case when only a single instrument is connected, your application program need do only the following;

1) Call **VerifyConnections** The first time the program is run this will fail and return "1", since the correct registry structure will not exist, so on this event the program should:
   - Call **CreateSRCommsRegKeys** to create the registry keys, and
   - Call S**canForInstruments** , which will search for the connected instrument and then update the registry keys.
2) Call **OpenSRComms(0)** This will open the communications session with Instrument 0
3) Use the **WriteReadSRInstrument** function with an instrument number parameter of 0 to send commands to the instrument and receive responses from it.
4) On program termination, call **CloseSRComms(0)** to close communications with Instrument 0.

3.3.02 Multiple Instrument Applications

When using several instruments it is useful for the user to be able to check that they

are all present and functioning from a single dialog box. The control therefore provides the **ConfigureInstruments** function to make this as easy as possible. To use this, your application program needs to do the following;

1) Call **VerifyConnections** The first time the program is run this will fail and return "1", since the correct registry structure will not exist, so on this event the program should then call **CreateSRCommsRegKeys** to create the registry keys.

2) Call **ConfigureInstruments** This will display a dialog box similar to that shown in figure 3-1 below, with no entries on the first occasion it is called.

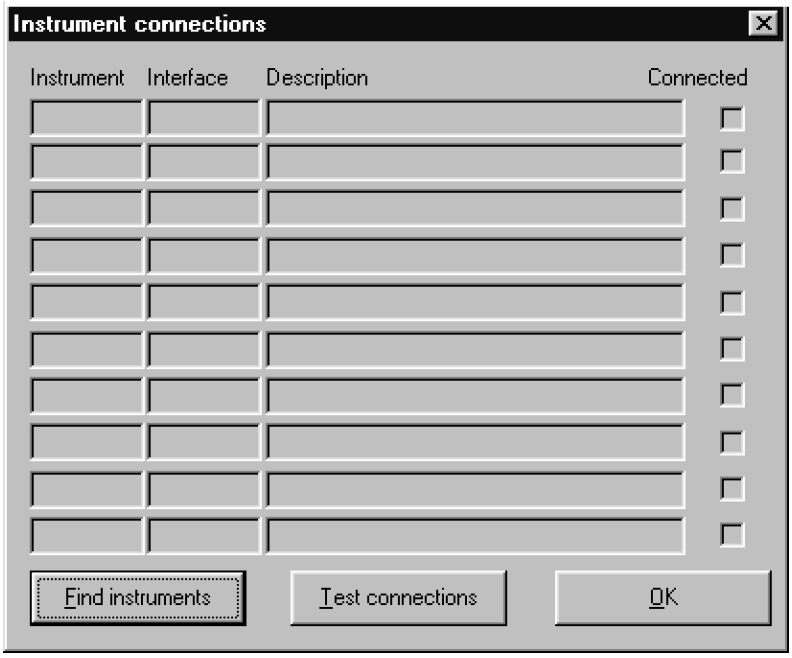

Figure 3-1, Instrument Connections Dialog Box- No Instruments Available

3) The user should then click **Find Instruments** and **OK** to proceed at the warning "This action will overwrite all instrument descriptions". This runs the **ScanForInstruments** function, which starts a search of the GPIB and then the serial ports for any compatible instruments and on completion updates the dialog box. For example, with four instruments connected the updated dialog box might appear as in figure 3-2 below.

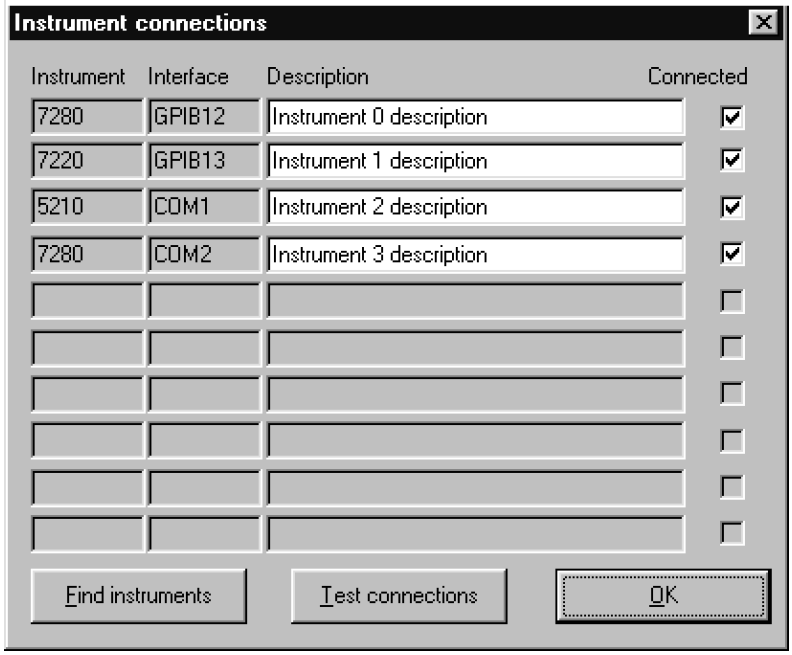

Figure 3-2, Instrument Connections Dialog Box - 4 Instruments Available

4) If desired, the default instrument descriptions can be changed so that they are meaningful in a given experiment. For example, descriptions such as "photocurrent", "sample voltage" or "temperature" could be used.

5) The column on the right hand side shows whether the listed instrument is actually present and connected. The next time this dialog box is activated, the stored connection profile is reloaded and tested, and if an instrument is not found then the relevant connected box will be unchecked. If the instrument should be present, then the user should make any necessary connections or turn it on and click **Test Connections**, which runs the **VerifyConnections** function, to re-test the connections.

6) Once the configuration process is complete, the user will click on **OK** to close the dialog box. At this point the registry structure will hold details of all the instruments that are connected and functioning.

7) Your application program or its user now needs to open communications with all the instruments listed. In order to determine the Instrument number to use, call **GetActiveInstruments** and use the arrays it returns to identify the instrument number to use for communication with a specific instrument. In the simplified case of the dialog box shown above, where all instruments listed are present, then the first instrument is Instrument 0, the second Instrument 1, and so on.

8) Call **OpenSRComms(nInstrumentNumber)** in a loop that cycles through all the required instrument numbers. This will open the communications session with all the instruments.

9) Use the **WriteReadSRInstrument** function, using the required instrument number, to send commands to the relevant instrument and receive responses from it.

10) On program termination, call **CloseSRComms** in a loop similar to that used in section 7 above to close communications with all instruments.

3.4 Methods

3.4.01 Introduction

The SRInstComms Control provides nine methods (or functions) to control containers. It does not have any user accessible properties. This section describes the methods in detail, with each method description starting on a new page to give the greatest clarity.

3.4.02 CreateSRCommsRegKeys

short CreateSRCommsRegKeys()

Description:
This function checks if SRComms registry keys exist under the HK_CURRENT_USER\Software\SIGNAL RECOVERY key in the registry and if not creates them.

The keys save 10 instrument settings (Instrument 0 to Instrument 9) and under each one saves:- the Model Number, interface type (currently GPIB or RS232 only), GPIB address (1 to 32), RS232 Port (COM1 to COM4), RS232 Baud Rate, RS232 Data Bits and RS232 Parity.

A key is also used to save whether a given instrument as actually present at that address.

Returns:
0 if successful, 1 otherwise

Notes:
This function must be run at least once on each machine that will use the SRInstComms control. The simplest method to ensure that the keys exist is to call the function at the start of your program. If they are not present then the function will create them, but if they are then the existing settings will not be affected.

Once created, the structure needs data for at least the key for Instrument 0 before communications with any instrument are possible. The **ScanForInstruments()**or **ConfigureInstruments()** functions store such information, while the **VerifyConnections()** function updates the connection status for instruments referenced in the registry keys.

3.4.03 ScanForInstruments

short ScanForInstruments()

Description:
This function searches for compatible ***SIGNAL RECOVERY*** instruments on the GPIB and serial port COM1 to COM4 interfaces and saves details of the first ten units that are found back into the SRComms registry keys. It assumes that the instrument's GPIB port parameters are set to CR with EOI or EOI only terminator and that the RS232 parameters are one of the following:-

1) 9600 Baud, 7 Data bits, 1 Stop Bit, Even Parity, Echo On, Prompt On
2) 19200 Baud, 7 Data bits, 1 Stop Bit, Even Parity, Echo On, Prompt On
3) 4800 Baud, 7 Data bits, 1 Stop Bit, Even Parity, Echo On, Prompt On
4) 4800 Baud, 8 Data bits, 0 Stop Bits, Even Parity, Echo On, Prompt On

Setting 1 is the setting implemented by an Auto Default/Default Setting operation on the models 7220, 7220BFP, 7225, 7225BFP, 7260, 7265 and 7310. It is also the factory default for the other models except the 5105 and 5106, which use Setting 4.

The instrument description comment fields for Instruments 0 to 9 in the registry are set to "Instrument 0 description" to "Instrument 9 description" respectively.

Returns:
0 if successful, 1 otherwise

Notes:
It is always advisable to check at the start of each program that instruments as saved in the keys are actually present before attempting communication with them. Use the **VerifyConnections()** function to do this.

3.4.04 ConfigureInstruments

short ConfigureInstruments()

Description:
This function displays a dialog box similar to that shown in figure 3-3 below.

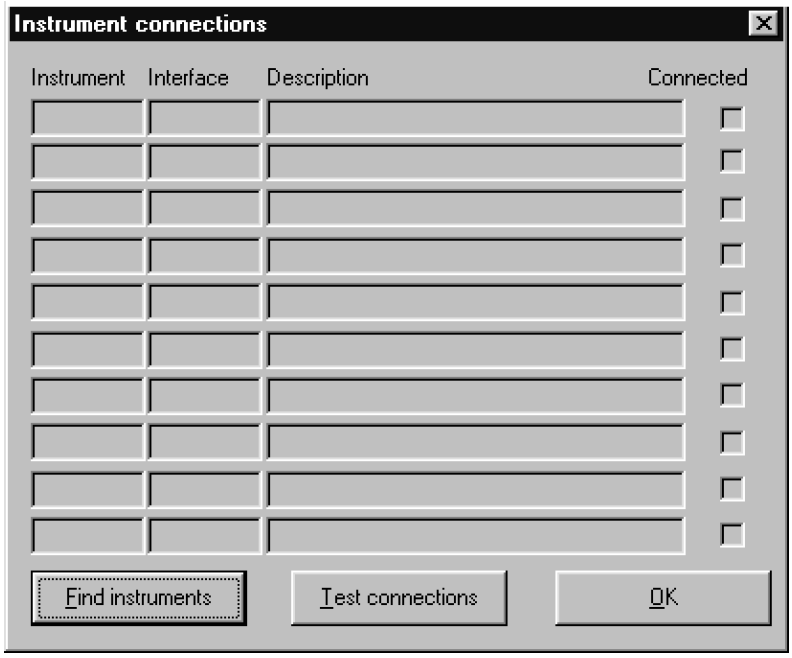

Figure 3-3, Instrument Connections Dialog Box - No Instruments Available

The **Find Instruments** button runs the **ScanForInstruments()** function, while the **Test Connections** button runs the **VerifyConnections()** function.

On completion of either function the dialog box is updated. For example, with four instruments connected the updated dialog box might appear as in figure 3-4 below.

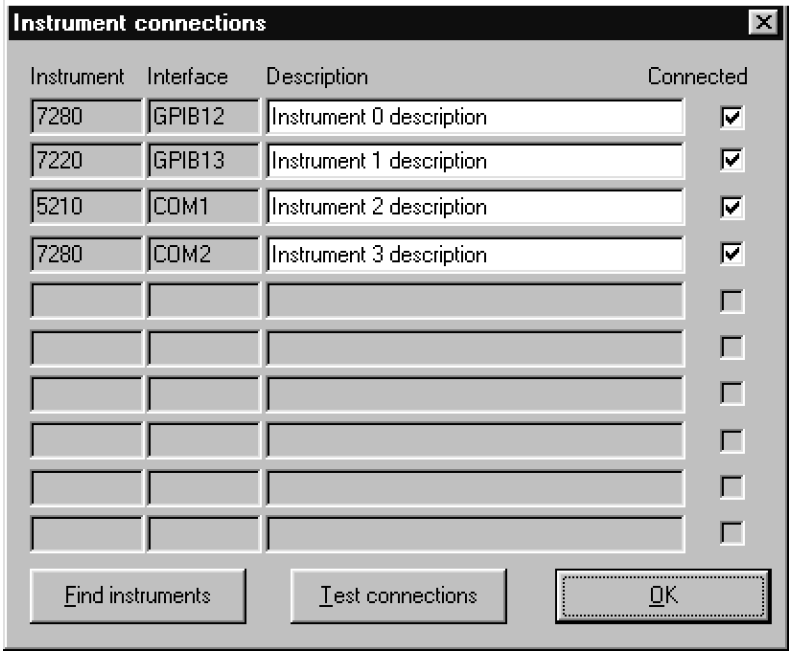

Figure 3-4, Instrument Connections Dialog Box - 4 Instruments Available

If desired, change the default instrument descriptions so that they are meaningful in the experiment. For example, descriptions such as "photocurrent", "sample voltage" or "temperature" could be used.

The column on the right hand side shows whether the listed instrument is actually present and connected. The next time this dialog box is activated, the stored connection profile is reloaded and tested, and if an instrument is not found then the relevant connected box will be unchecked. If the instrument should be present, then make any necessary connections or turn it on and click **Test Connections** to re-test the connections.

Once the configuration process is complete, click on **OK** to close the dialog box.

Returns:
0 if successful, 1 otherwise

Notes:
Use this function to give users easy access to the underlying registry keys.

3.4.05 VerifyConnections

short VerifyConnections()

Description:
This function checks each instrument number in the registry keys. If a model number is present, it attempts to open it and if successful sets the corresponding Present status key to "1"; if the instrument is not found it sets the key to "0".

Returns:
0 if successful, 1 otherwise

Notes:
It is always advisable to check at the start of each program that instruments as saved in the keys are actually present before attempting communication with them.

3.4.06 OpenSRComms

short OpenSRComms(short nInstrumentNumber)

Description:
Before communicating with an instrument the “channel” to it must be opened, in the same way as opening a file handle is required before accessing a file.

This function searches the SRComms section of the registry and uses the key under the specified instrument number parameter nInstrumentNumber. If RS232 is specified as the interface, it tries to open that serial port at the specified baud rate. If successful, it stores a handle to the open port back in the registry under the appropriate key.

If GPIB is specified as the interface, then it checks that the board is present.

If either interface exists, it then sends an ID command to the instrument to read its type and if a response is received and it matches the model number held in the key then the Present status registry key is set to "1". In other cases the Present status registry key is set to "0". It also sets the delimiter used to separate multiple instrument responses to a comma (ASCII 44), by sending the DD 44 command.

Returns:
0 if successful, 1 otherwise.

Notes:
If you open communications with an instrument using this function then you must use the corresponding function **CloseSRComms(short nInstrumentNumber**) to close the channel before terminating you program.

3.4.07 WriteReadSRInstrument

short WriteReadSRInstrument(short nInstrumentNumber,
BSTR sCommand,
VARIANT* sResponse,
VARIANT* vResponseArray,
VARIANT* nStatusByte)

Description:
This is the core write/read function for communicating with the instrument(s). The instrument to use is specified by the nInstrumentNumber parameter, and must first have been opened using the **OpenSRComms(short nInstrumentNumber)** function.

The function uses the information under the specified nInstrumentNumber instrument number in the registry keys and if RS232 is specified as the interface then the existing opened file handle is used for communications; if GPIB then the GPIB address is used.

The parameters have the following significance:

nInstrumentNumber Integer number of the opened instrument with which to communicate.

sCommand String containing the command to send to the instrument, e.g. "ID"

sResponse Variant containing string returned from the instrument as a result of command sCommand. Null if there is no response.

vResponseArray Variant array containing the results of a curve dump command, for instruments that respond to such commands. Each element of the array is a variant of type string representing one response line, so for example if a curve length of 100 points is being dumped there will be 100 elements to the array.

nStatusByte Variant containing an integer with the most recent value of the serial poll status byte or response to the ST command. The information contained by the byte can be interpreted by reference to the instrument instruction manuals.

Returns:
0 if successful, 1 otherwise

Notes:
Commands are converted to upper case and stripped of leading and trailing whitespace before being sent to the instrument.

When the GPIB interface is used, the **nStatusByte** value is the response to a serial poll carried out after reading any instrument response. When the RS232 interface is used, the **nStatusByte** value is set to "1" unless the instrument prompt following a command

is "?". In this case the function sends an ST command to determine the status byte value and returns this in the **nStatusByte** parameter.

When auto functions are called, the function behaves as follows:

RS232 Interface
Function returns when command completes, or after 60 seconds time out, whichever occurs first.

GPIB Interface
Function returns when command completes.

The function correctly handles the dump curve binary (DCB) command implemented in the 7220, 7220BFP, 7225, 7225BFP, 7265 and 7280 instruments. When the RS232 interface is in use, it switches to 8-bit data transmission before sending the command and reverts to the previous setting after it completes. It also implements "dot" (floating point) versions of this command, which are not actually supported by the instrument. When the command "DCB. n" is received, the function sends commands to determine the present sensitivity and other calibration settings to apply to the received data, sends a DCB n command to the instrument, reads the resulting data and then corrects it for the present calibration.

Note: This technique assumes that all the data being transferred was acquired using the same sensitivity, time constant and slope settings, and that these settings have not changed since acquisition..

3.4.08 CloseSRComms

short CloseSRComms(short nInstrumentNumber)

Description:
This function searches the SRComms section of the registry and uses the key under the specified instrument number parameter nInstrumentNumber. If RS232 is specified as the interface, it then closes the specified serial port handle. If GPIB is specified as the interface, then it sends an interface clear.

Returns:
0 if successful, 1 otherwise.

Notes:
Use **OpenSRComms(short nInstrumentNumber)** to open a communications channel.

All open channels must be closed before terminating your program.

3.4.09 WriteCommsRegKeyComment

short WriteCommsRegKeyComment(BSTR sComment)

Description:
This function writes a comment string sComment to the registry key for the last instrument opened with the **OpenSRComms(short nInstrumentNumber)** function.

The comment field makes it easier for the user to identify the purpose of each instrument rather than the alternative, which would be to refer to instruments by connection type/address/port number.

Returns:
0 if successful, 1 otherwise.

3.4.10 GetActiveInstruments

short GetActiveInstruments(VARIANT* vActiveInstruments,
VARIANT* nNoOfActiveInstruments,
VARIANT* vActiveInstrumentsIndex)

Description:
This function is provided to allow application programs to determine which instruments are connected to the system and currently active. When called, it checks the SRComms registry keys for all ten instruments, Instrument 0 to Instrument 9 and returns an string array containing information about the active instruments, an integer between 0 and 10 which is the present total number of active instruments, and a ten-element integer array that contains the instrument number corresponding to each item in the string array.

The parameters have the following significance:

vActiveInstruments	Variant string array containing between zero and ten elements, each of which contains the model number and comment string, e.g. "7280 'Instrument 0 description'"
nNoOfActiveInstruments	Variant containing an integer representing the total number of presently active instruments, and hence the dimension of the vActiveInstruments variant array
vActiveInstrumentsIndex	Variant array containing ten elements, each of which is a variant of type integer identifying the instrument number corresponding to the given position in the vActiveInstruments variant array.

Hence, for example, if there are three instruments listed in the registry structure, but only two of them, Instrument 0 and Instrument 2, are connected, then the **vActiveInstruments** array will have two elements, e.g. "7280 'Instrument 0 description'" and "7265 'Instrument 2 description'", while the corresponding first two **vActiveInstrumentsIndex** array elements will be 0 and 2.

Returns:
0 if successful, 1 otherwise

Notes:
This function allows an application program to implement dropdown combo or list box showing instruments that are available and presently active.

Demonstration Files

Chapter 4

4.1 Introduction

The SRInstComms control is supplied with demonstration files showing how it can be used in with Visual C++, Visual Basic, VBScript, LabVIEW 6.0 and Excel 97. These files are installed by default in subfolders of the c:\Program Files\Signal Recovery\SRInstComms folder.

4.2 Demonstration Files

4.2.01 Visual C++

The files supplied form a workspace which may be opened in Visual C++ version 6.0 When compiled, the result is a simple dialog box application in the form of a "static interface" allowing the user to type a command in a text box, click the **Send Command** button and monitor the response. When several instruments are connected to the system then a list box allows one on these to be selected as the instrument to use. Figure 4-1 shows the main dialog box of this application.

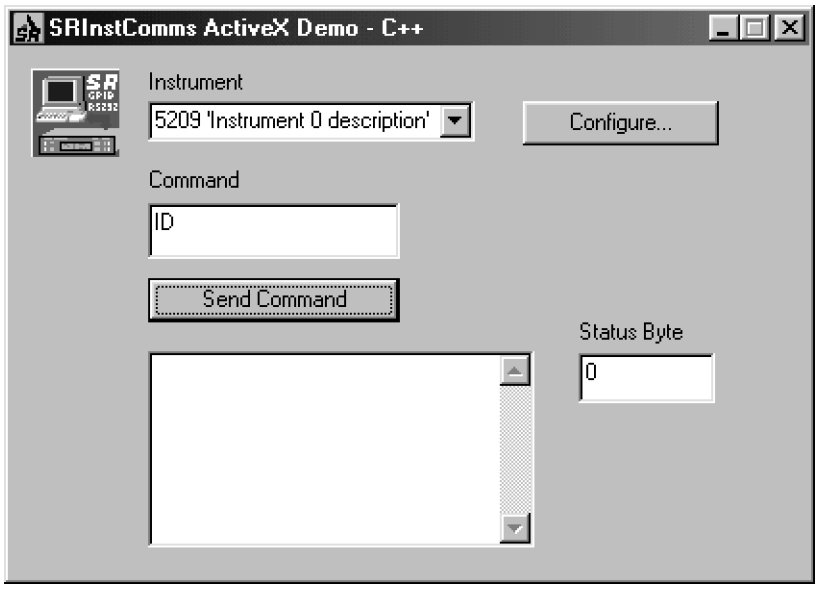

Figure 4-1 SRInstComms.exe C++ Main Dialog Box

Click the **Configure** button to open the **Instrument Connections** dialog box which shows the instruments connected to the system. It also allows the instrument descriptions to be edited.

4.2.02 Visual Basic

The files supplied form a workspace which may be opened in Visual Basic version 6.0 When compiled, the result is similar to the application generated by the Visual C++ demonstration files. A compiled version of this project, SRInstComms.exe, is also

included as an executable file that can be run “as supplied”.

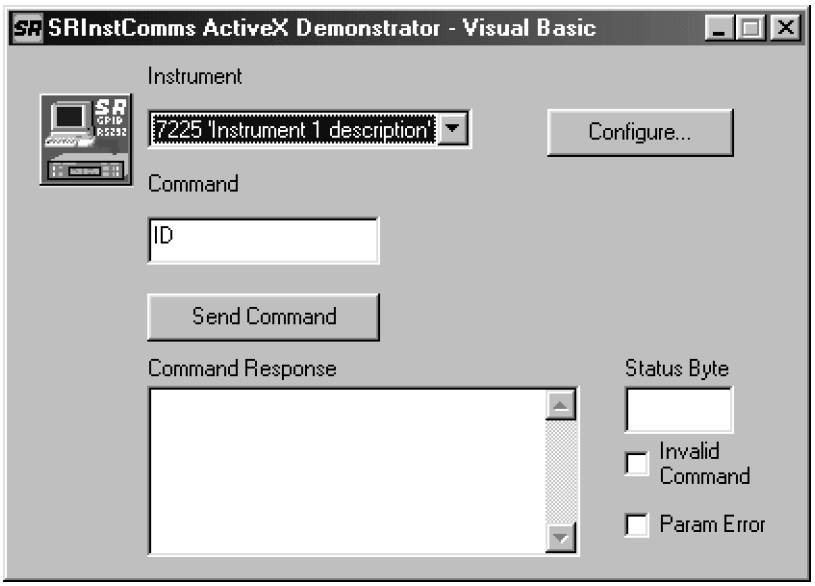

Figure 4-2 SRInstComms.exe Visual Basic Main Dialog Box

Click the **Configure** button to open the **Instrument Connections** dialog box which shows the instruments connected to the system. It also allows the instrument descriptions to be edited.

4.2.03 VBScript (HTML)

The supplied file can be run under Internet Explorer and provides a simple interface routine to communicate with the instrument.

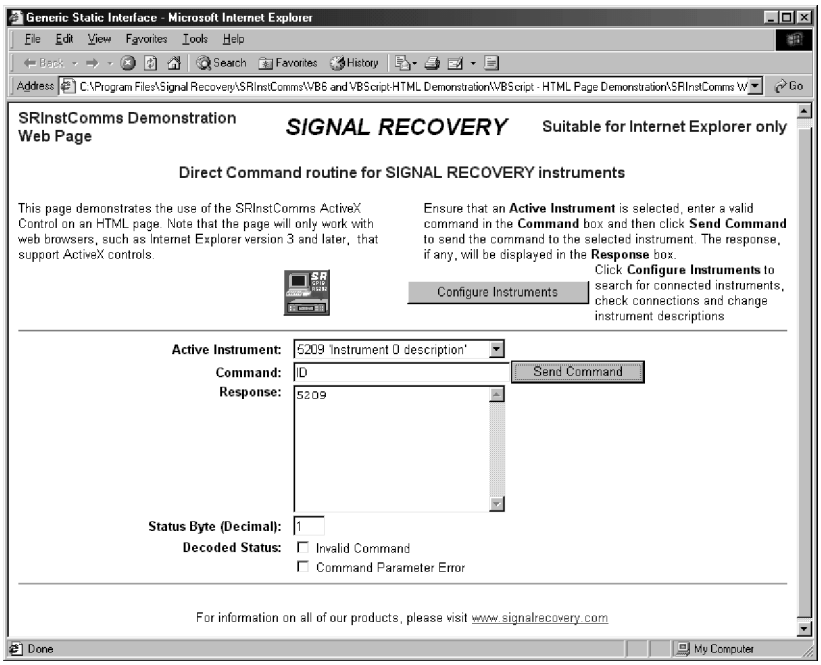

Figure 4-3 SRInstComms HTML Demonstration File

4.2.04 Excel 97

The demonstration workbook contains one worksheet. The sheet has similar "static interface" capabilities to the Visual C++ and Visual Basic projects, but in addition shows how the curve buffer found in the 7220, 7220BFP, 7225, 7225BFP, 7265 and 7280 can be programmed to acquire a number of points and then display the result graphically.

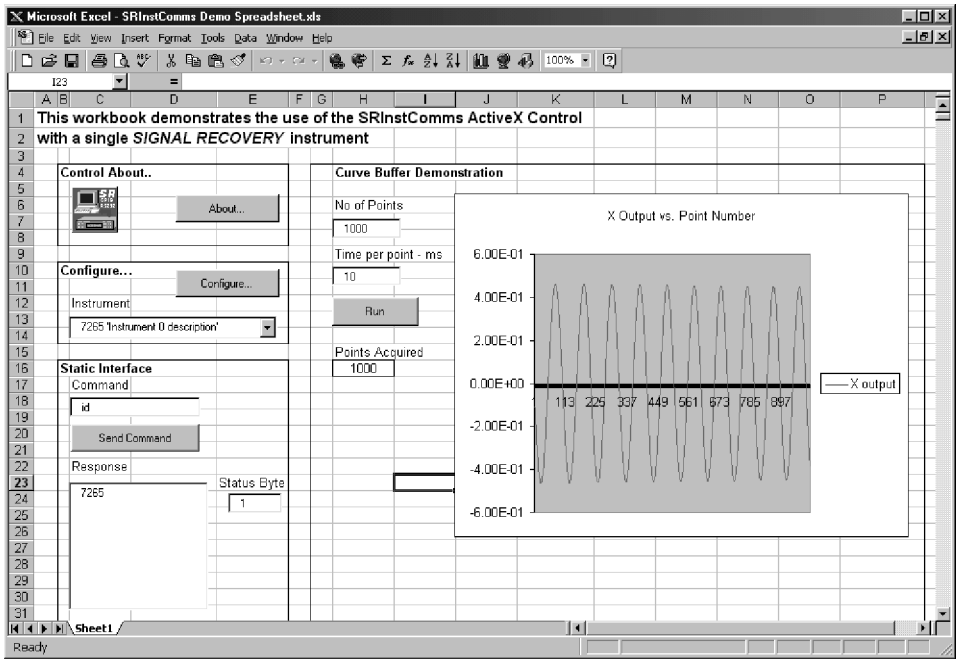

Figure 4-3 SRInstComms Demo Spreadsheet.xls Excel 97 Workbook

Click the **Configure** button to open the **Instrument Connections** dialog box which shows the instruments connected to the system. It also allows the instrument descriptions to be edited.

Click the **About** button to show the control's About dialog box.

4.2.05 LabVIEW 6.0

The demonstration VI offers similar functionality to that of the Excel 97 workbook.

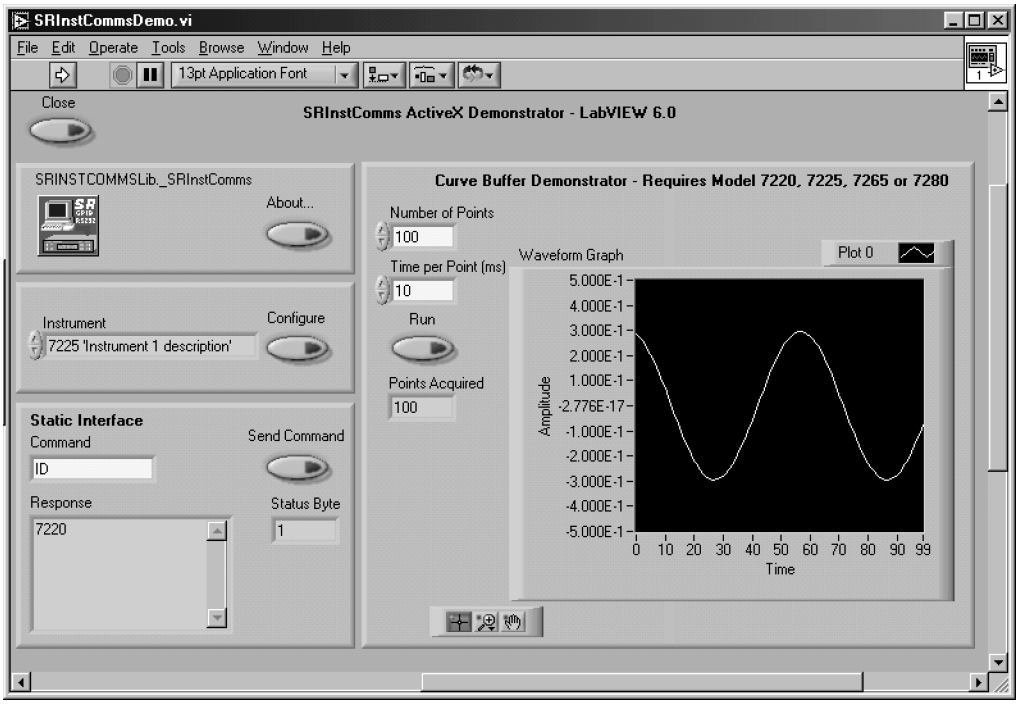

Figure 4-4 SRInstCommsDemo.vi LabVIEW VI - Front Panel

Click the **Configure** button to open the **Instrument Connections** dialog box which shows the instruments connected to the system. It also allows the instrument descriptions to be edited.

Click the **About** button to show the control's About dialog box.

Cable Diagrams

Appendix A

A.1 Introduction

RS232 serial ports may be configured as one of two types, these being DTE (data terminal equipment) and DCE (data communications equipment). All ***SIGNAL RECOVERY*** lock-in amplifiers are configured as DTE devices, which is the same as most personal computers. Consequently, in order to satisfactorily connect a lock-in amplifier to a computer it is necessary to use a *Null Modem* (also known as a *Modem Eliminator*) cable.

The model 5113 preamplifier is however configured as a DCE and hence requires a straight through cable, rather than a null modem type.

For historical reasons units are fitted with one of three types of RS232 serial interface connectors, which when taken with the fact that the serial port on the computer may be fitted with either a 9-pin or 25-pin connector, means that one of eight types of cable will be needed.

This appendix shows the connection diagrams for all eight types of cable, which should prove of use to readers with reasonable practical skills who would like to assemble their own interconnecting cables.

A.2 Diagrams

Models 5105 and 5106 to computers with 9-pin serial connectors.

These units require a 9-pin female to 9-pin female null modem cable.

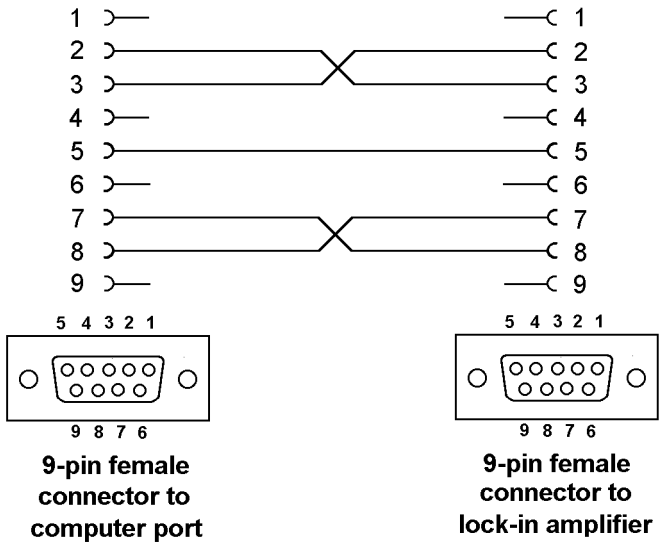

Figure A-1, Interconnecting RS232 Cable for Models 5105 & 5106 to computers with 9-pin Serial Interface Connectors

Models 5105 and 5106 to computers with 25-pin serial connectors.

These units require a 25-pin female to 9-pin female null modem cable.

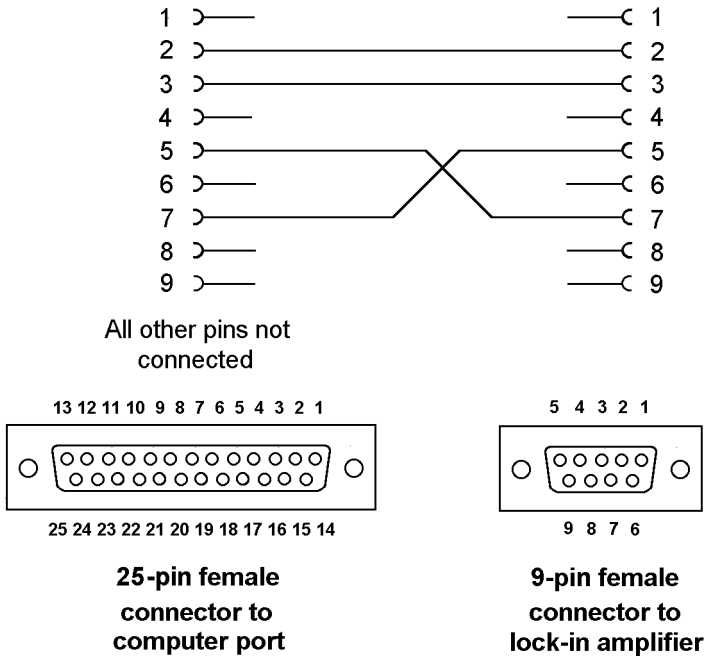

Figure A-2, Interconnecting RS232 Cable for Models 5105 & 5106 to Computers with 25-pin Serial Interface Connectors

Models 5109, 5110, 5209, 5210 and 5302 to computers with 9-pin serial connectors.

These units require a 9-pin female to 25-pin male null modem cable.

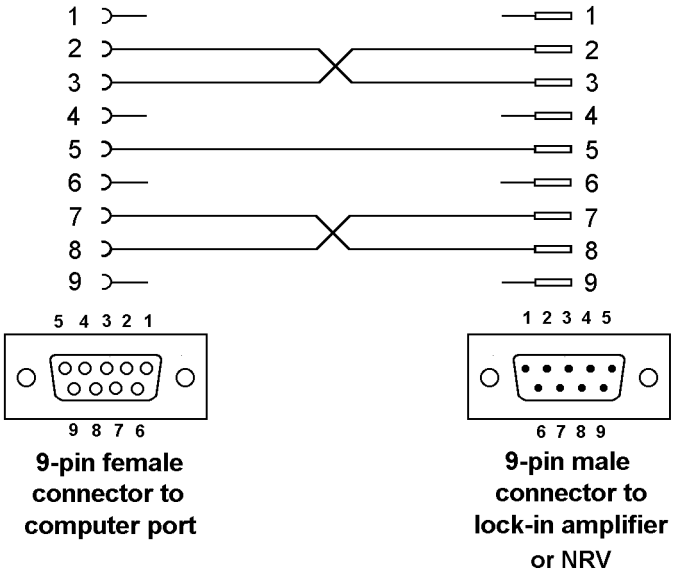

Figure A-3, Interconnecting RS232 Cable for Models 5109, 5110, 5209, 5210 & 5302 to Computers with 9-pin Serial Interface Connectors

Models 5109, 5110, 5209, 5210 and 5302 to computers with 25-pin serial connectors.

These units require a 25-pin female to 25-pin male null modem cable.

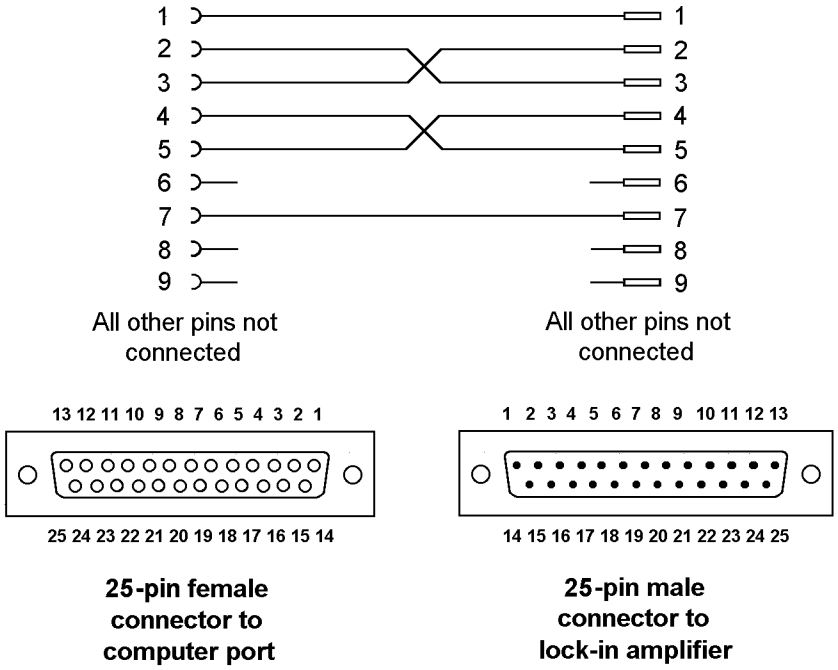

Figure A-4, Interconnecting RS232 Cable for Models 5109, 5110, 5209, 5210 & 5302 to Computers with 25-pin Serial Interface Connectors

Models 7220, 7220BFP, 7225, 7225BFP, 7260, 7265, 7280 and 7310 to computers with 9-pin serial connectors.

These units require a 9-pin female to 9-pin male null modem cable.

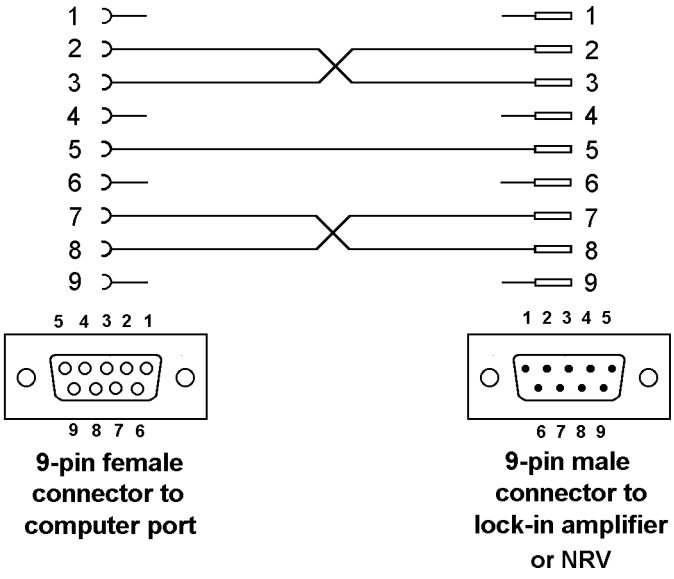

Figure A-5, Interconnecting RS232 Cable for Models 7220, 7220BFP, 7225, 7225BFP, 7260, 7265, 7280 and 7310 to Computers with 9-pin Serial Interface Connectors

Models 7220, 7220BFP, 7225, 7225BFP, 7260, 7265, 7280 and 7310 to computers with 25-pin serial connectors.

These units require a 25-pin female to 9-pin male null modem cable.

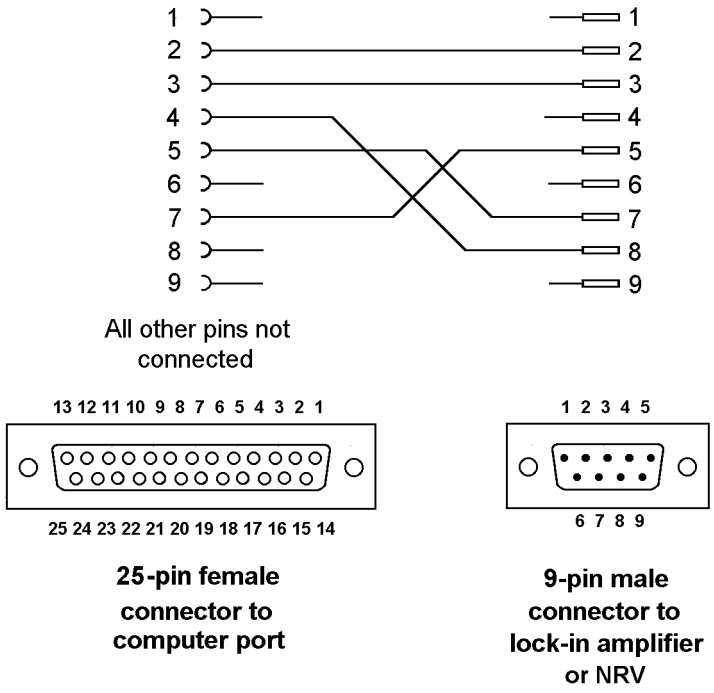

Figure A-6, Interconnecting RS232 Cable for Models 7220, 7220BFP, 7225, 7225BFP, 7260, 7265, 7280 and 7310 to Computers with 25-pin Serial Interface Connectors

Model 5113 to computers with 9-pin serial connectors.

This unit requires a 9-pin female to 25-pin male cable.

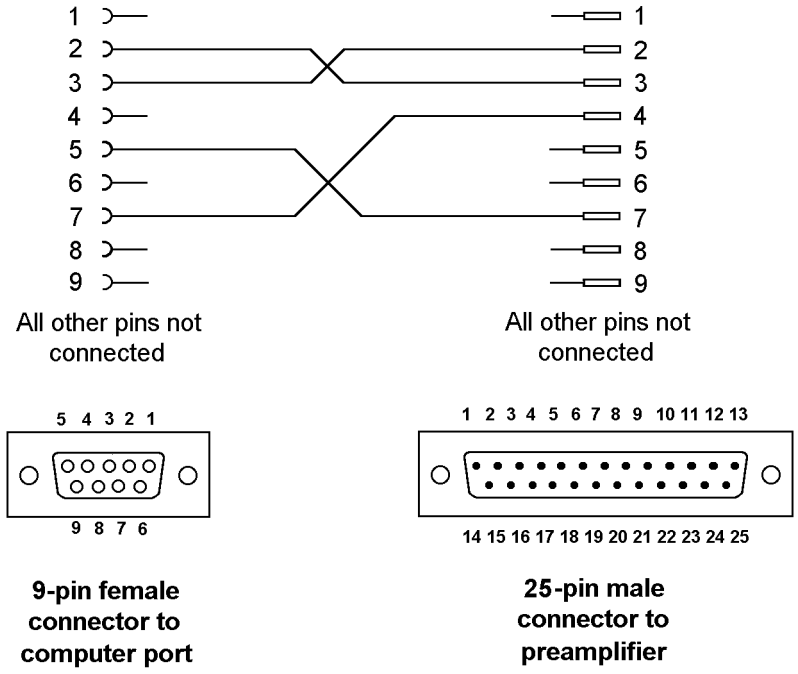

Figure A-7, Interconnecting RS232 Cable for Model 5113 to Computers with 9-pin Serial Interface Connectors

Model 5113 to computers with 25-pin serial connectors.

This unit requires a 25-pin female to 25-pin male cable.

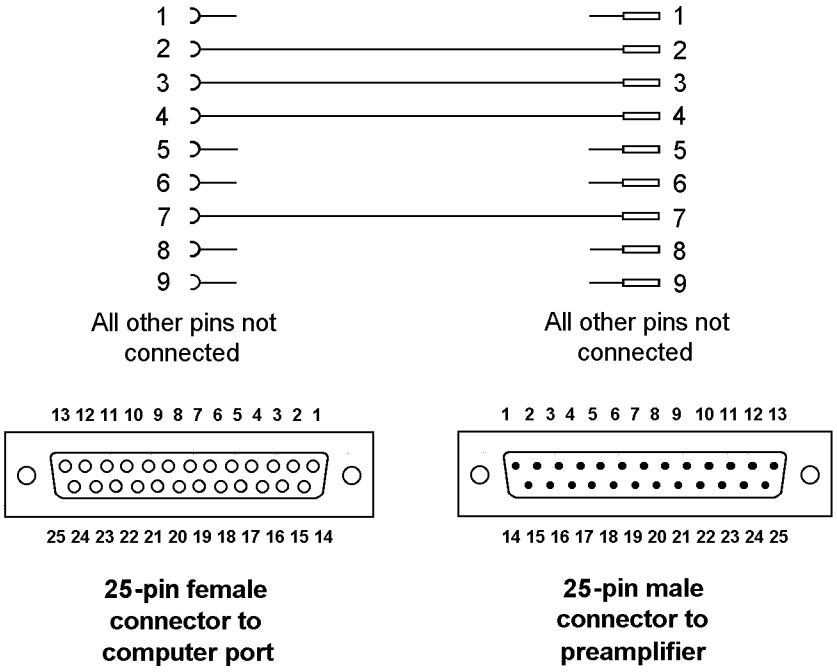

Figure A-8, Interconnecting RS232 Cable for Model 5113 to Computers with 25-pin Serial Interface Connectors

Software License & Support

Appendix B

B.1 Introduction

This appendix provides details of the software license and support policy which applies to the SRInstComms™ Active X control and discusses how you may obtain help in using it from us. It also discusses how you may help us by reporting any errors you discover and making suggestions for new features.

SIGNAL RECOVERY is granting you a non-exclusive license to use the software program and associated documentation (collectively, the "Software") for your internal use in accordance with the following terms:

B.2 License

You may:

a) use the Software (including any subsequent updates) on a single computer system;

b) copy the Software into any machine readable or printed form for backup or modification purposes in support of your use of the Software on a single computer, provide that you make only one such copy for such purposes;

c) modify the Software and/or merge it into another program for **your personal use only on a single machine** (Any portion of the Software merged into another program will continue to be subject to the terms and conditions of this Agreement);

d) transfer the Software and this license to a third party if the other party agrees to accept all the terms and conditions of this Agreement. If you transfer the Software, you must at the same time notify the name and address of the third party to ***SIGNAL RECOVERY*** and either transfer all copies, whether in printed or machine readable form, to the same party or destroy all copies not transferred; this includes all modifications and all portions of the Software contained in or merged into other programs.

You must reproduce and include the copyright notice on any copy, modification or portion merged into another program.

B.2.01 Updates and Upgrades

If this Software is an update or upgrade, then the purchaser must have the original Software and license Agreement which becomes part of this Agreement. The original Software may not be transferred to another computer or party except as part of this product.

You may not use, copy, modify, or transfer this Software or any copy, modification or merged portion of it, in whole or in part except as expressly provided for in this license

If you transfer possession of any copy, modification or merged portion of this Software to another party, your license is automatically terminated.

B.2.02 Term

The license is effective until terminated. You may terminate it at any time by destroying the Software together with all copies, modifications and merged portions in any form. It will also terminate upon conditions set forth elsewhere in this Agreement. You agree upon such termination to destroy the Software together with all copies, modifications and merged portions in any form.

B.2.03 General

You may not sublicense, assign or transfer the license or the Software except as expressly provided in this Agreement. Any attempt otherwise to sublicense, assign or transfer any of the rights, duties or obligations hereunder is void.

If you wish to incorporate the control into your own application program for sale, distribution or use by anyone other than yourself you must obtain a separate license from us to cover this distribution.

This Agreement will be governed by the laws of the state of Tennessee, USA.

B.3 Software Registration

SIGNAL RECOVERY will provide technical support in the use of the SRInstComms™ Active X control to one registered, named, user for each copy purchased. In order to allow this to operate you must therefore complete the product registration form located on our website at **www.signalrecovery.com/register**

Once registered, you may then telephone, fax or e-mail your query to the ***SIGNAL RECOVERY*** office in Bracknell, UK. You must give the product serial number (located on the supplied Setup Disk), together with the registered user's name, on each occasion you contact us, or support will be denied.

If at all possible, when telephoning, please be in front of the computer and lock-in amplifier you are using, so that you may immediately try our suggestions. You must also be able to tell us the type of computer and computer interface you are using, the version of the SRInstComms™ Active X control software and the version of Windows.

Registered users of the software will also be informed of future product upgrades and enhancements.

B.4 Software Errors

The SRInstComms™ Active X control has been extensively tested prior to release, but it is possible that some errors remain which were not revealed by the testing process.

SIGNAL RECOVERY would very much appreciate receiving, by e-mail to **service@signalrecovery.com** or by post or fax to the Bracknell, UK, office, written notification of any problems you discover.

When reporting problems please try to be as specific as possible. Include details of the type of instrument used, the revision level of the instrument firmware and the type of interface and computer used.

If you feel there is a specific feature which would add to the software's functionality then please let us know. Naturally we cannot commit ourselves to adding any given new feature, but even so such suggestions are useful in identifying those areas in which the product may be improved.

Index

WARRANTY

AMETEK SIGNAL RECOVERY, a part of AMETEK Advanced Measurement Technology, Inc. warrants to the original user of this product that it shall be free of defects resulting from faulty manufacture of the product or its components. AMETEK Advanced Measurement Technology, Inc. makes no warranties regarding either the satisfactory performance of the software encoded on this product or the fitness of the software for any particular purposes. Defects covered by this warranty shall be corrected either by repair or by replacement, as determined by AMETEK Advanced Measurement Technology, Inc.

AMETEK Advanced Measurement Technology, Inc. reserves the right to make revisions to the software program(s) without incurring any obligation to install same on software previously sold or supplied.

Licensee hereby acknowledges that the computer programs/software, including, without limitation the design thereof, documentation and other information relating thereto are trade secrets and continue to be proprietary to AMETEK Advanced Measurement Technology, Inc. Licensee agrees not to take any action which might encumber or expose the computer programs/software to any claims, transfers, conversion or other forms of encumbrance.

THERE ARE NO WARRANTIES WHICH EXTEND BEYOND THE DESCRIPTION HEREIN. THIS WARRANTY IS IN LIEU OF AND EXCLUDES ANY AND ALL OTHER WARRANTIES OR REPRESENTATIONS EXPRESSED IMPLIED OR STATUTORY INCLUDING MERCHANTABILITY AND FITNESS FOR PURPOSE AS WELL AS ANY AND ALL OTHER OBLIGATIONS OR LIABILITIES OF AMETEK ADVANCED MEASUREMENT TECHNOLOGY, INC. INCLUDING BUT NOT LIMITED TO SPECIAL OR CONSEQUENTIAL DAMAGES. NO PERSON, FIRM OR CORPORATION IS AUTHORISED TO ASSUME FOR AMETEK ADVANCED MEASUREMENT TECHNOLOGY, INC ANY ADDITIONAL OBLIGATION OR LIABILITY NOT EXPRESSLY PROVIDED FOR HEREIN EXCEPT IN WRITING DULY EXECUTED BY AN OFFICER OF AMETEK ADVANCED MEASUREMENT TECHNOLOGY, INC.

If you do not agree with the provisions of this license agreement then you should return this product unopened.

We reserve the right to deny customer service, including but not limited to software updates and telephone support to any person or organization using software not registered withAMETEK ***SIGNAL RECOVERY***

SHOULD YOU REQUIRE SUPPORT

A. Contact ***SIGNAL RECOVERY*** at the address below.

B. We will need the following information.

1. Serial number of the software and the model number of the instrument being used
2. Your name (instrument user)
3. Your address
4. Your telephone number and extension or e-mail address
5. Symptoms (in detail, including control settings)

C. For support on this product please contact:

SIGNAL RECOVERY Service
AMETEK Advanced Measurement Technology
Astro House
Brants Bridge,
Bracknell
RG12 9HW
UNITED KINGDOM

Phone: +44 (0)1344 477900
Fax: +44 (0)1344 477931
E-mail: service@signalrecovery.com
Wbsite: www.signalrecovery.com