How to use VXI Plug and Play Driver with Agilent VEE, C/C++, Visual Basic, LabVIEW, and LabWindows/CVI

Application Note

Why VXIplug&play and its history

There are two competing paradigms for human-instrument interaction. One is controlling an instrument through a physical front panel. The other is controlling the instrument through a software command set accessible through a standard I/O interface (e.g. RS-232, IEEE 488.2, and LAN). The first paradigm is well suited to using an instrument for individual measurements. The second is well suited for more complex sequences of measurements, possibly coordinated over several physical instruments.

To begin with, standard programming languages (e.g. C) were used to access instrument command sets. To accommodate some of the difficulties of programming instruments, HP created versions of BASIC (Rocky Mountain BASIC, HP-BASIC) which were tuned to engineering in general and instrument programming in particular. Eventually the search for more intuitive ways of creating instrument programs led HP to develop ITG (Interactive Test Generator - now obsolete), which in turn led to HP VEE.

HP VEE is HP's Visual Engineering Environment. It provides a way for non-programmers to visually program test systems without having to learn a traditional programming language. The user creates a data flow diagram using components provided with VEE. To do instrument control, VEE requires a driver (VEE ID) which can bridge the gap between VEE and the details of the instrument command set.

VEE ID's were invented to work with ITG and VEE. They provide a soft front panel to which the customer can connect data flow lines either to control the instrument or to read measurements from the instrument. These drivers are written in a proprietary VEE language. These drivers work only in ITG and VEE; they do not work in other programming environments. Since VEE's introduction, Agilent Technologies has created a library of over 450 drivers covering many of Agilent Technologies instruments.

The problem with VEE drivers is that they can only be used with VEE. In addition to programming languages like C/C++ and BASIC, there are several other programming environments for instrument programming, including National Instrument's LabWindows and LabVIEW. To add to the complication, each vendor has proprietary ways of dealing with IEEE 488.2 and VXI I/O.

The VXI Systems Alliance is an unincorporated consortium of VXI instrument manufacturers including HP, National Instruments, Tektronix, and Racal Dana, among others. One of its accomplishments is the creation of a driver standard, which will allow a single driver to work with a variety of I/O protocols and programming environments. This VXIplug&play Driver standard is now complete. The main components of the driver architecture for Windows are the driver DLL, VISA, the function panel definition, and the soft front panel.

The driver consists of multiple files. At the core is a system library (.dll for Windows). A soft front panel provides a GUI for the driver, while the function panel file allows use in graphical programming environments. A windows help file, a Visual Basic include file, and a VXI knowledge base file round out the normally supplied files.

The driver DLL is a collection of functions, which allow the programmer to access the instrument. It can be called from standard programming environments like C/C++ or Visual Basic. The DLL focuses on the functionality of the instrument, not the details of I/O. The DLL is built on top of VTL/VISA, which handles the I/O.

VTL (interim) and VISA provide I/O independence by handling 488.2 and VXI I/O. VISA implementations are specific to a particular manufacturer's protocols. For instance, if you are using a HP HP-IB card, you must have a HP version of VISA installed which understands HP-IB and SICL. However,

with the correct VISA installed, you can use a suitable VXIplug&play driver from any manufacturer to control an instrument.

The function panel allows the driver to be used with several instrumentation specific programming environments. It provides information that can be used to create a graphical window (panel) from which the .dll functions can be executed. At the present time HP VEE, LabWindows, and LabVIEW can interpret the function panel definition correctly and use it in their respective environments.

Finally, the soft front panel is an executable, which allows an instrument user to access the instrument functionality through a software front panel using the paradigm of a physical front panel.

The VXIplug&play specification places requirements on all of the above components, as well as the functions available in the driver. There are required entry points as well as rules for naming entry points. There are also rules about how the entry points tie together with the function panel.

VXIplug&play drivers can be used with several versions of VEE. VEE 3.12 is able to call a VXIplug&play driver, but the mechanism is rather awkward and does not take advantage of the function panel file. VEE 3.12 works with Windows 3.1. VEE 3.2, 4.0, 5.0, and 6.0, have a very nice interface which takes advantage of the function panel file, and work with Windows 95, Windows NT, and Windows 98. VEE 6.0 also works on Windows 2000.

How do I get it to work with VEE?

A PnP driver is simply a Windows DLL (Dynamic Link Library) and so is accessed and controlled by VEE's I/O Instrument Manager objects. First, you will need the VISA32.DLL interface library (installed in the \WINDOWS\SYSTEM directory) to allow the PnP driver to access an HPIB or GPIB interface. VISA32.DLL will not be shipped with VEE; it will be shipped with the interface card or found in one of the following URL's depending on the interface card. If you have an Agilent Card you could find them here at http://ftp.agilent.com/pub/mpusup/pc/binfiles/iop/index.html or if you have a National Instruments card you could go to www.ni.com. (Please reference Appendix A, Section I for VEE visuals).

The following instructions assume that VISA is installed and working properly, and that the instrument driver has been installed properly.

Agilent VEE Pro 6.0 works well with Universal Instrument Drivers. Agilent VEE 3.1 works with UIDs, but requires more manual activities on the user's part. These instructions assume Agilent Pro 6.0. The instructions for Agilent VEE 3.2, 4.0, 5.0 are similar. Consult the Agilent Pro 6.0 manual titled Using VXIplug&play Drivers with HP VEE, for more details.

Configuration

Start up Agilent VEE. From the menu, select I/O | Instrument Manager.. VEE will open the Instrument Manager window. If you have not used VEE with VXIplug&play drivers before this, the control labeled Instrument List should be blank. Click the Add Instrument button. This opens the Instrument properties window. This is where you place the Instrument Name, Interface, and Address. Next, click the Advanced.. | Plug&Play Driver. The first control is a drop-down list with all of the installed VXIplug&play drivers that Agilent VEE can detect. Select the one you want to work with. The bottom part of the window contains all of the parameters to the hp816x_init function except for the instrument handle. Fill in the Address appropriate resource string (e.g. GPIB0::5::INSTR), and check the Perform Reset and Perform Identification Query boxes. Normally, both of these would be checked. However, if you want to use a driver with an instrument for which it is not intended (for instance, to use utility or passthrough functions only), do not check the ID validation box. Click the OK button on the VXIplug&play Device Configuration window. The instrument should now appear as a Currently Configured Instrument in the VXIplug&play Instrument Configuration window.

Using Driver Functions

Agilent VEE helps the user by assuming some of the more mundane tasks involved in using VXIplug&play drivers. For instance, the user does not have to execute the hp816x_init function. VEE knows when driver functions need to be executed and performs the initialization before that. It also performs the hp816x_close function after completing a VEE program that uses driver functions. Finally, it keeps track of the instrument handle. The user never has to deal with it, either in the device configuration (as noted above) or for any other function. Since VEE handles these details, the user is free to concentrate on instrument functionality.

To add a driver function to the VEE program, select I/O | Instrument Manager from the VEE menu. Highlight the instrument in question i.e. hp816x and double click on the Plug&Play Driver under the Create I/O Object. VEE will allow you to place a To/From hp816x box in the programming area. One or more driver functions may be added to this box. To add a function to the box, double click on blank space in the box. The Select a Function Panel window appears. This window shows the entire function tree for the driver. Single clicking a node on the tree will display help in the bottom of the window. Double clicking a function node selects the function. When a function is selected, the Select a Function Panel window disappears and the function panel for the function is displayed. Click OK in the panel. The function panel disappears and the

function is listed in the To/From hp816x box. Any function panel controls for the function will appear on the edges of the To/From hp816x box as input and output terminals. Output controls on the function panel appear as output terminals on the right hand side of the box. All other function panel controls (except Instrument Handle, as noted above) appear as input terminals on the left hand side of the box. These terminals can then be connected to other visual elements of the VEE language as desired.

As a simple exercise, add the hp816x_reset function to the box. There are no terminals to hook up, so this program is easy. Just run it, and you're done. Next, try hooking up the hp816x_revision_query function. Note that the two output strings also appear on the right hand side of the To/From hp816x box. This is because you must allocate storage for all strings and arrays before calling driver functions. VEE Automatically allocates arrays for its specific output parameters, as opposed to other languages. At this point you are ready to run this program to query the instrument.

How do I get it to work with Visual Basic?

Configuration

Start up Microsoft Visual Basic with a Standard EXE. From the menu, select Project/Add Module.. Visual Basic will open the Add Module window. Click the Existing button. This opens the Look in window. Use normal file dialog box navigation to get to the vxipnp\winxx\include directory for the module hp816x.bas as well as the visa32.bas. When you have the file selected, click the open button. You may only be able to open one file at a time so you may have to repeat the described procedure to include both of these files. (Please reference Appendix A, Section II for Visual Basic Visual).

Using Driver Functions

To exercise the functions that the hp816x.bas has to offer double click on the Form Window. The Form Window allows you to create the windows, dialog boxes, and controls in your application. You draw and view controls on a form. You should now have opened the Form (Code) Window. Here in the code window you could develop your program to read and write to the instrument. The easiest way to find out what functions are available is to go to the hp816x.hlp help file found in the VXIPNP\win95\HP816x directory. The functions are referenced either by Alphanumeric or Hierarchical under the Referenced Information section of the help file. You may also access the function by double clicking on the hp816x.bas module. The functions of interest will be preceded by the Declare Function.

Now you are able to begin exploring the functions and running the instrument remotely. Please refer to your Visual Basic manual on proper calling of DLL's.

How do I get it to work with C?

Configuration

The following is a summary of important compiler-specific considerations for several C/C++ compiler products when developing WIN32 applications. (Please reference Appendix A, Section IV for MS Visual C++ visuals).

For Microsoft Visual C++ version 6.0 compilers select File | New and select either Win32 Application or Win32 Console Application depending on your preference. This will enable you to create an empty project. Now that the project has been created, go to Project | Dependencies...to see whether or not you have the correct project for modification. We will now go to the code generation setting to change the settings from single to multithread. VISA requires these definitions for WIN32. This setting changes the value of the /MD, /ML, and /MT compiler switches. It is usually best to use the multi-threaded options when building a DLL. If you specify a single-threaded option, your DLL will work reliably only when called by single-thread applications. The way this is done is by going to the menu bar and selecting Project | Settings...Click on the C/C++ button. Select Code Generation from the Category list box and select Multithreaded DLL from the Use run-time Libraries list box. Click the OK to close the dialog boxes.

Specify an object file or standard library (either static or import) to pass to the linker. Select Project | Settings from the menu. Click on the Link button and add visa32.lib and hp816x.lib to the Object | Library Modules list box. Separate file names with a space. Optionally, you may add the library directly to your project file. Click on OK to close the dialog boxes.

You may wish to add include file and library file to the search paths. They are set by selecting Tools | Options from the menu. Click on the Directories button to set the include file path. Select Include Files from the Show directories for: list box. Double-click in the Directories window to add the c:\VXIPNP\WINxx\INCLUDE directory. Now, select Library Files from the Show directories for: list box. Double-click in the Directories window to add the c:\VXIPNP\WINxx\LIB\MSC directory.

If using Borland C++ compilers, you may wish to add the include file and library file paths. They are set under the Options | Project menu selection. Double-click on Directories from the Topics list box and add c:\VXIPNP\WINxx\LIB\BC and c:\VXIPNP\WINxx\INCLUDE directories.

You are now able to begin programming the hp816X mainframes. Before you start the coding I would suggest taking a look at the visa examples we supply with the driver. They could be found at c:\VXIPNP\WINxx\hp816x\visa directory. Have fun.

How do I create a LabVIEW VI Library?

LabVIEW 4.0 and higher works with VXIplug&play drivers. It reads in the driver functions, creates an icon for each function, and adds the Gwin95/NT error handling terminals.

Configuration

To load a Universal Instrument Driver into LabVIEW, select Tools |Instrumentation | Update VXIplug&play Drivers... from the LabVIEW menu. The UPDATE VXIPNP DRIVERS window appears. It contains two large text boxes. The top one contains a list of installed VXIplug&play GWIN drivers, and does not concern us. The bottom one contains VXIplug&play drivers that have not been converted to LabVIEW VI libraries, or whose VI libraries are out of date. To convert (or update) one of these to a VI library for use with LabVIEW, select it and click FP Options. Once in the FP Options refer to the Appendix A, section III for the proper settings. After clicking OK, the CVI conversion status box appears. If everything goes well, it will disappear when the conversion is complete.

Selecting Driver Functions

To select a driver function, select File | Open from the LabVIEW menu. The Choose the VI to Open dialog box appears. The default directory for this box is the main LabVIEW directory. Double click the INSTR.LIB subdirectory to open it. The file hp816x.llb should appear in the file list. Select the file and click OK. LabVIEW will open the File Dialog box that lists all of the driver function VI's in alphabetical order. Select the one you want to add to your program and click OK. LabVIEW displays the GWin95/NT framework VI panel for the function. To add a driver function to a program, click the Select a VI... icon from the functions palette. The Choose the VI to Open dialog box appears, and the function is selected as described above. In this case, however, LabVIEW adds the function icon to the LabVIEW program rather than displaying the GWin95/NT framework VI panel. The icon can then be connected to other visual program elements. LabVIEW error handling terminals were designed to be connected in series. If any one function in the series fails, the downstream driver function VI's will not execute.

How do I get it to work in LabWindows/CVI?

Function panels originated in LabWindows/CVI as a way of seeing what individual driver functions would do before integrating them into CVI programs. LabWindows/CVI is Agilent's current choice for a function panel editor. It can also be used to exercise installed drivers and should compile all Agilent Technologies drivers

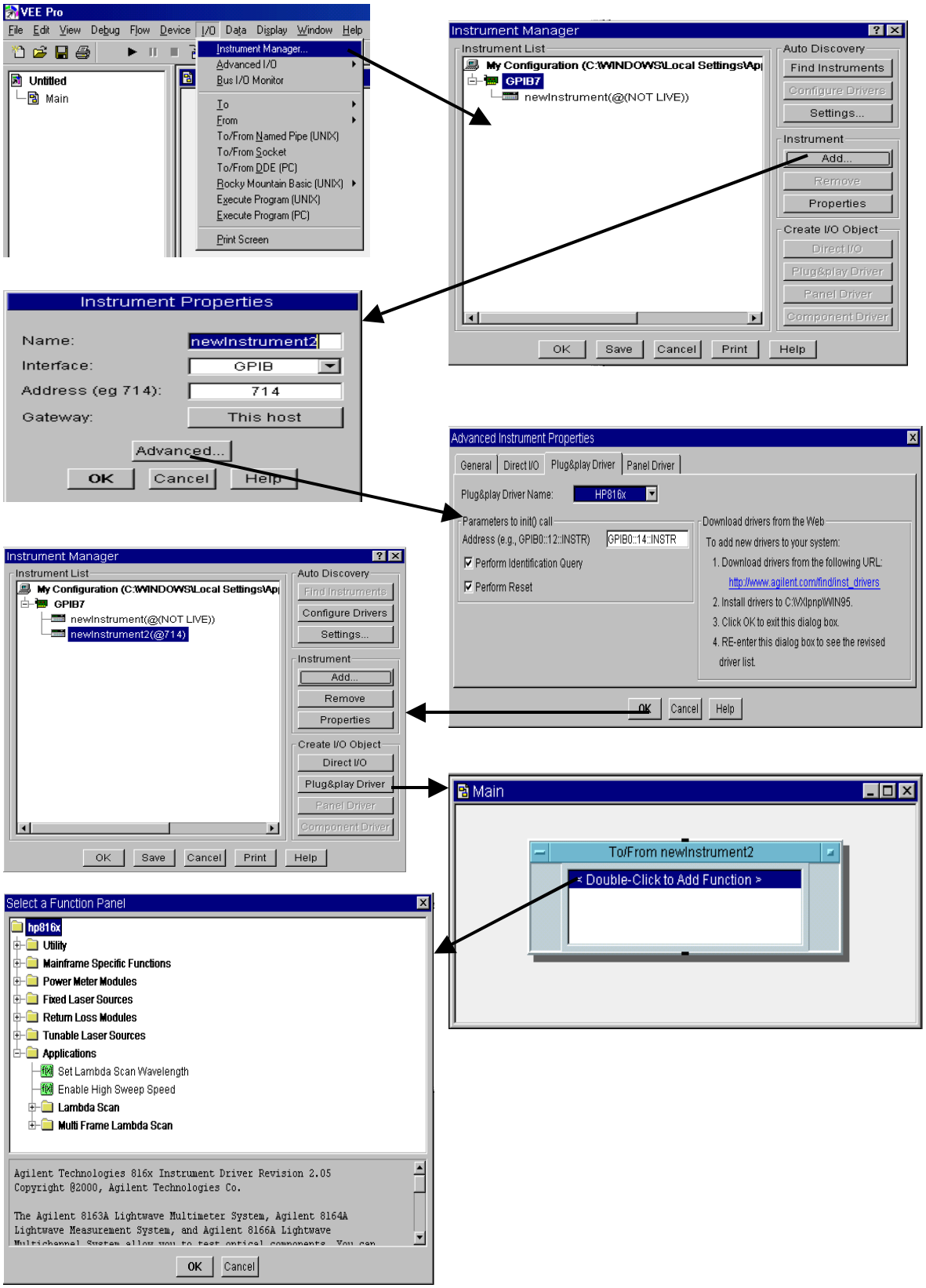
Exercising Function Panels

To exercise the function panel, select Instrument | Load... from the menu. The Load Instrument file dialog box appears. Navigate to the vxipnp|winxx|include directory and select the .fp file. Include the HP816x_32.lib file into the LabWindows/CVI project, by clicking Add Files to Project: Library (*.lib) from the Edit menu item in the project window. Make sure the .h and .dll files are in the same directory as the LabWindows/CVI project. At this point the driver is "loaded" into LabWindows. Next, select Instrument from the menu again. You will see the driver you just loaded as a choice on the top part of the list. Select the instrument. The first level of the function tree will appear in a function tree window. It should include the Initialize and Close functions, as well as the first level of function tree classes (indicated by "..." after the name, e.g. Utility... Application... etc.) To select a class or a function, double click it. If you double click the class, the function tree window display changes to the content of the class. If you double click a function, the function panel window will appear. Once a function panel is displayed, the function tree window disappears. To re-display it, click on the tree icon.

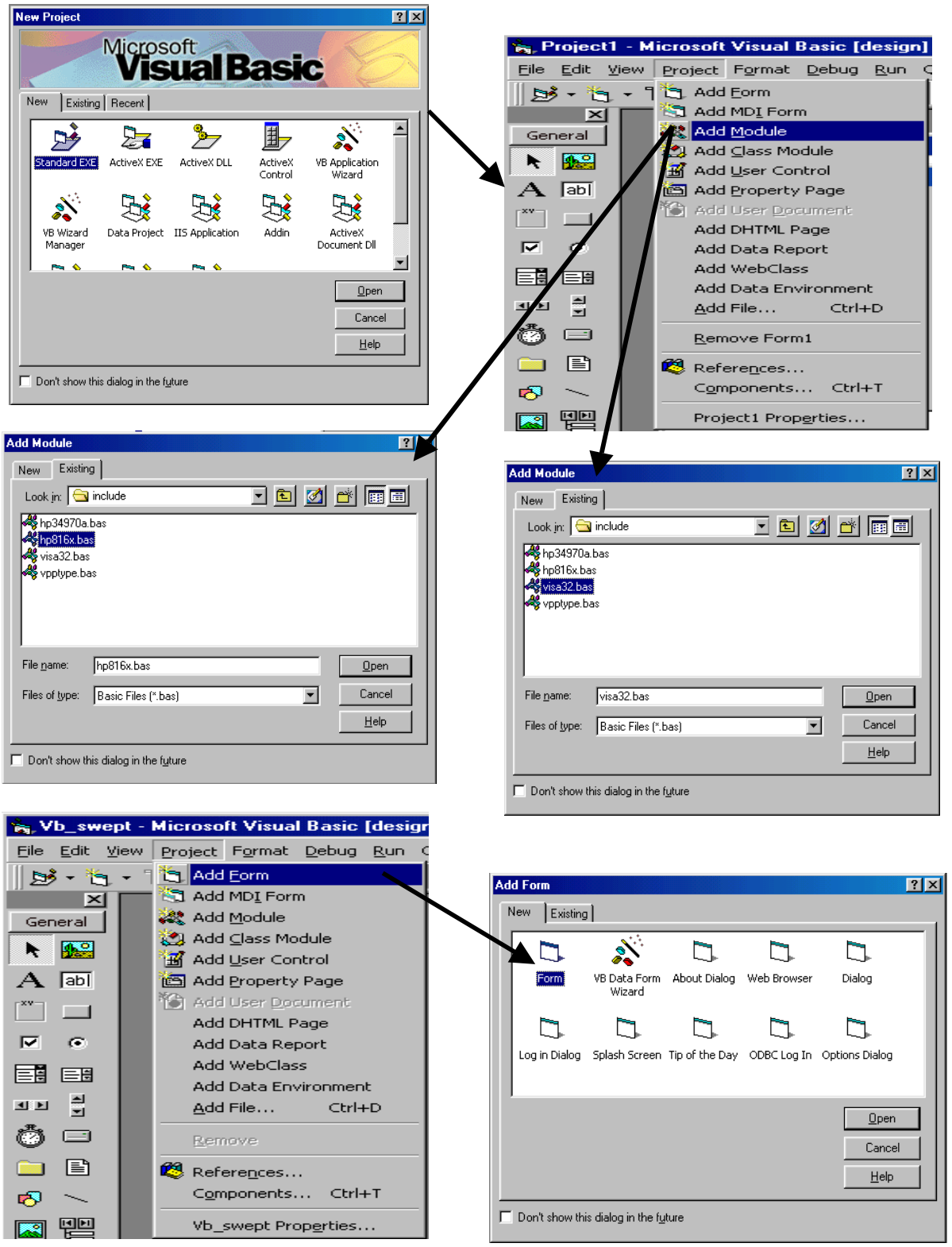
When exercising driver function panels, always start with Initialize and end with Close.

LabWindows/CVI automatically creates temporary variables with cryptic names for scalar output controls. You will need to create variables for strings and array output. You may create variables for scalars if you want meaningful names. To declare variables, click on a control to select it, then select Code | Declare Variable... from the function panel's menu. Once you have entered values for all input controls and have taken care of variables for output, run the function panel by clicking the runner icon. The status control will return a 0 if the function executes correctly.

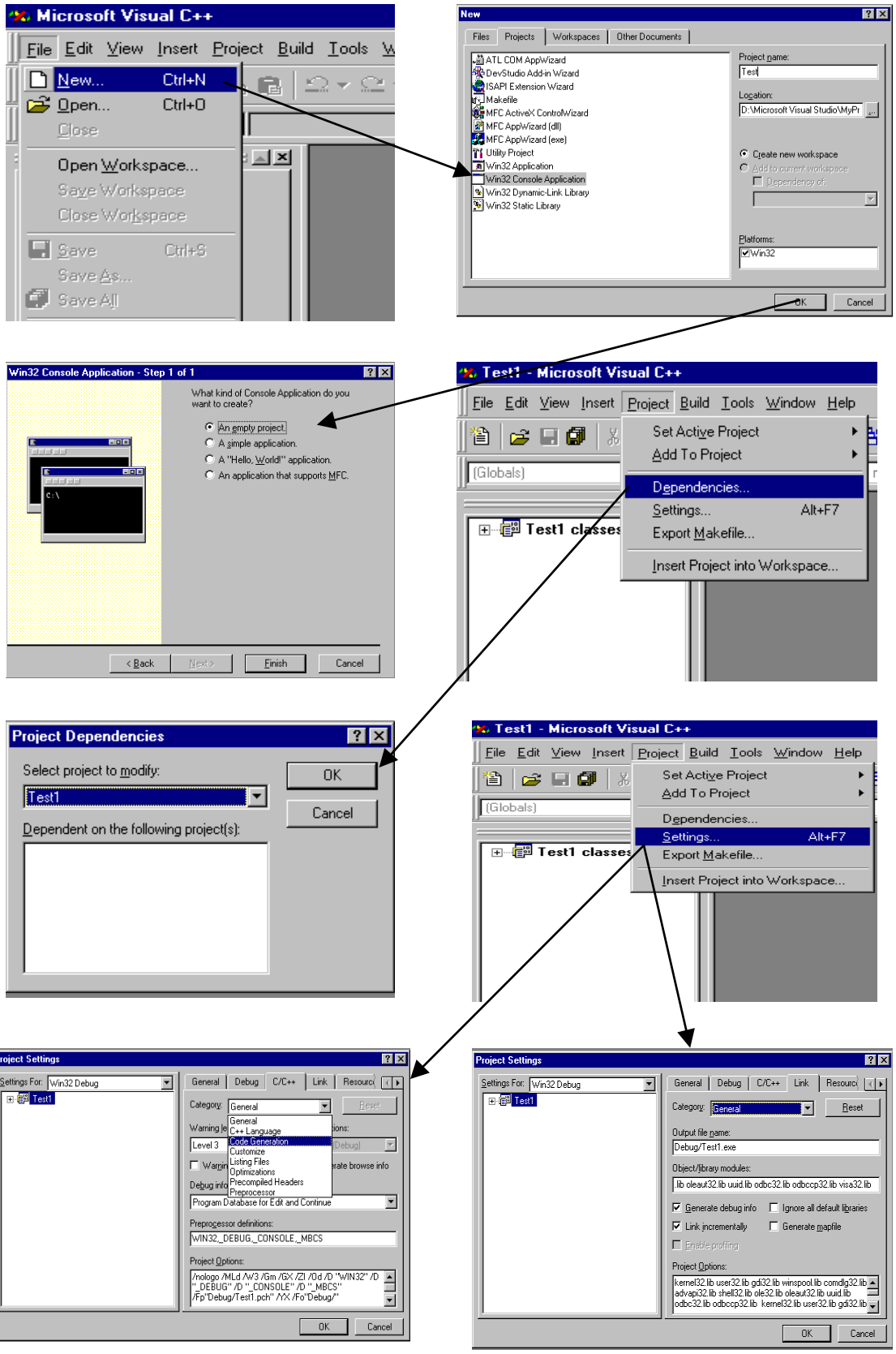
Section I: Agilent VEE Pro 6.0



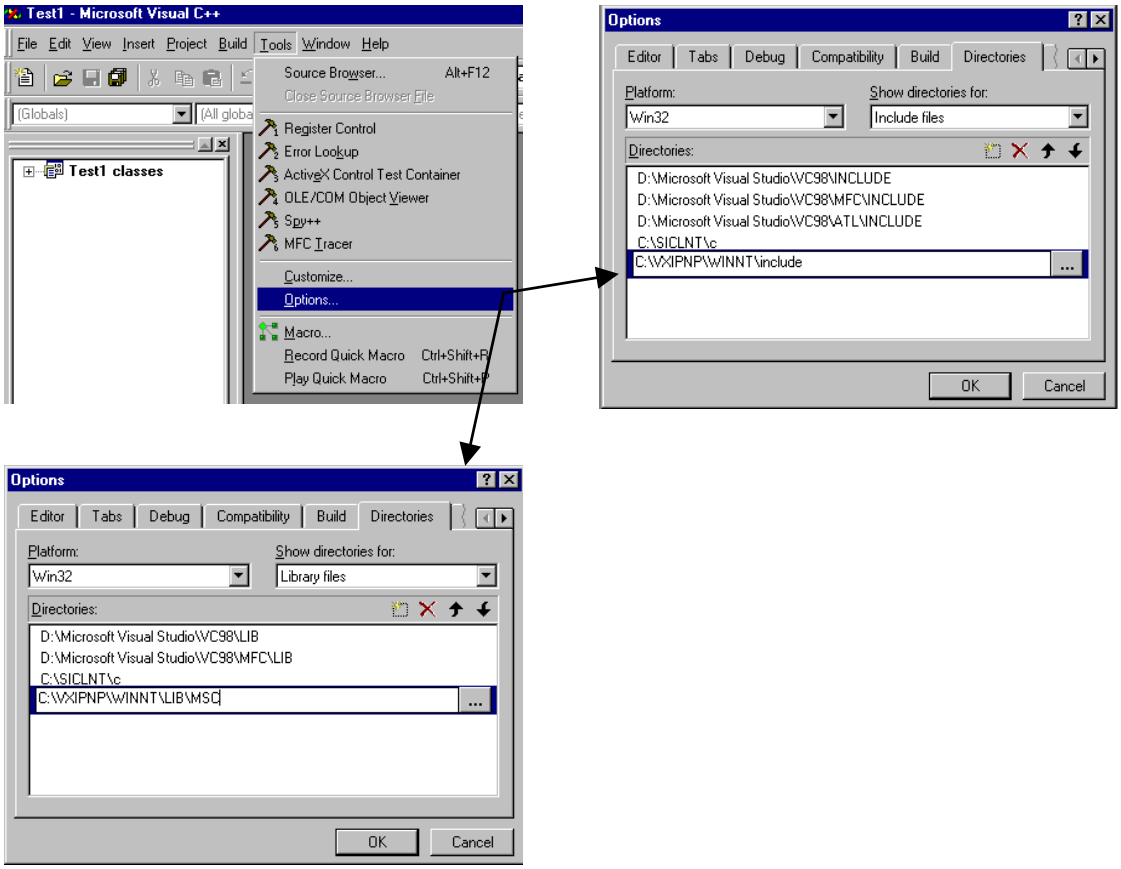
Section II: Visual Basic 6.0



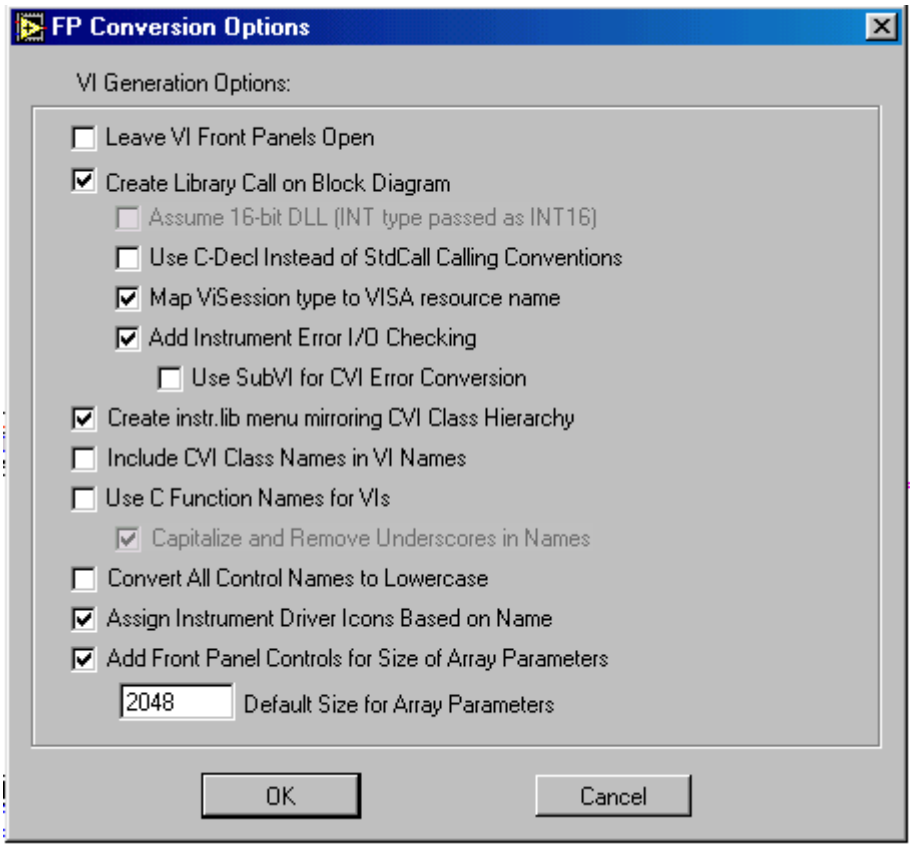
Section III: Microsoft Visual C++



Continuation...



Section IV: LabVIEW Function Panel Options



Agilent Technologies' Test and Measurement Support, Services, and Assistance

Agilent Technologies aims to maximize the value you receive, while minimizing your risk and problems. We strive to ensure that you get the test and measurement capabilities you paid for and obtain the support you need. Our extensive support resources and services can help you choose the right Agilent products for your applications and apply them successfully. Every instrument and system we sell has a global warranty. Support is available for at least five years beyond the production life of the product. Two concepts underlie Agilent's overall support policy: "Our Promise" and "Your Advantage."

Our Promise

Our Promise means your Agilent test and measurement equipment will meet its advertised performance and functionality. When you are choosing new equipment, we will help you with product information, including realistic performance specifications and practical recommendations from experienced test engineers. When you use Agilent equipment, we can verify that it works properly, help with product operation, and provide basic measurement assistance for the use of specified capabilities, at no extra cost upon request. Many self-help tools are available.

Your Advantage

Your Advantage means that Agilent offers a wide range of additional expert test and measurement services, which you can purchase according to your unique technical and business needs. Solve problems efficiently and gain a competitive edge by contracting with us for calibration, extra-cost upgrades, out-of-warranty repairs, and on-site education and training, as well as design, system integration, project management, and other professional engineering services. Experienced Agilent engineers and technicians worldwide can help you maximize your productivity, optimize the return on investment of your Agilent instruments and systems, and obtain dependable measurement accuracy for the life of those products.

By internet, phone, or fax, get assistance with all your test & measurement needs

Online assistance: www.agilent.com

Phone or Fax

United States:
(tel) 1 800 452 4844

Canada:
(tel) 1 877 894 4414
(fax) (905) 282-6495

Europe:
(tel) (31 20) 547 2323
(fax) (31 20) 547 2390

Japan:
(tel) (81) 426 56 7832
(fax) (81) 426 56 7840

Latin America:
(tel) (305) 269 7500
(fax) (305) 269 7599

Australia:
(tel) 1 800 629 485
(fax) (61 3) 9210 5947

New Zealand:
(tel) 0 800 738 378
(fax) 64 4 495 8950

Asia Pacific:
(tel) (852) 3197 7777
(fax) (852) 2506 9284

Product specifications and descriptions in this document subject to change without notice.

August 9, 2001

5988-2790EN

Agilent Technologies