IVI-COM Instrument Driver Programming Guide (Setup Edition)

Dec 2003 Revision 1.0

1- What Is IVI-COM Instrument Driver

1-1 Types Of IVI Instrument Drivers

IVI-COM drivers are COM-based instrument drivers that comply with the IVI instrument driver specifications defined by the IVI Foundation (www.ivifoundation.org).

There are two types of IVI instrument driver APIs defined by the IVI specifications. One is called IVI-C, which is the expanded version of pre-existing VXI Plug&Play instrument drivers. This type of drivers is provided as a conventional Windows DLL. Another type is what called IVI-COM, which utilises Microsoft COM (Component Object Model) technologies. This type of drivers is provided as an in-process COM server (Windows COM DLL). All the IVI instrument drivers provided by Kikusui are IVI-COM.

IVI instrument driver specifications also define two driver types other than the categorisation of IVI-C and IVI-COM. One is what called IVI class-compliant driver, which belongs to particular instrument class. Another type is what called IVI custom driver, which does not belong to any instrument class. As instrument classes, IviScope, IviDmm, IviFgen, IviDCPwr, IviSwitch, IviPwrMeter, IviSpecAn, and IviRFSigGen are currently defined. A driver for the instrument that can be categorised to one of these classes is normally implemented as an IVI class-compliant driver; otherwise it is implemented as an IVI custom driver. Drivers provided by Kikusui can be either IVI class-compliant driver or IVI custom driver depending on the instrument type.

1-2 Interchangability

One of features of IVI instrument drivers is the interchangability function. This functionality provides a capability that an automation system does not have to be recompiled or re-linked to continue to work even if an instrument has been swapped with other model. How the interchangability can be realised is different between IVI-C and IVI-COM. Since IVI instrument drivers provided by Kikusui are all IVI-COM type, we explain about IVI-COM only.

To utilise the interchangability function, there must be IVI-COM instrument drivers provided for both pre-swapped and post-swapped instruments. Plus, they both must be IVI-COM class-compliant drivers that comply with the same instrument class. Also, an application that utilises the interchangability function must indirectly access the instrument drivers through the class interfaces, not through the driver-supplied specific interfaces.

For instance, Kikusui4800 IVI-COM instrument driver (for Kikusui PIA4800 series DC Power Supply Controller) complies with the IviDCPwr class. Therefore, if an application using the Kikusui4800 driver is designed with considering interchangability, you can replace the instrument driver with, for example, AgilentE36xx IVI-COM instrument drivers (for Agilent Technologies E3600 series DC Power Supply). The application can continue to work without being rebuilt. On the other hand, such application that considers interchangability cannot utilise instrument-specific features. For example, the IviDCPwr class interfaces define

functions for generic features of DC power supplies, however, do not define instrument-specific advanced functions.

1-3 Interoperability

IVI class-compliant drivers equip interchangability features. In contrast, IVI custom drivers do not. Custom drivers are, however, not completely designed with custom and free architecture. IVI specifications define standard API style that all the IVI instrument drivers shall comply with, regardless driver types. For instance, the Initialize function establishes instrument I/O connectivity, the Reset function resets the instrument, and the ErrorQuery function queries instrument errors. These inherent capabilities and developer's experiences are common for all the IVI instrument drivers (from any vendors, for any models). Therefore if you have learned how to use one instrument driver once, you can easily use other IVI instrument drivers of other models of other vendors. Like this, IVI instrument drivers are designed with considering interoperability.

1-4 Operational Performance

IVI instrument drivers have better operational performance. IVI specifications define RangeCheck, Cache, Simulate, QueryInstrStatus, RecordCoercions, and Interchange Check capabilities, which can be configured as inherent default operations. In particular, Cache and QueryInstrStatus settings provide mechanisms that will make your debugging work easier and make the final system better for performance.

Cache is a functionality, which omits wasting instrument I/O communications. (Default is TRUE.) For example, immediately after setting a voltage value, setting the same value again is redundant. When the Cache functionality is enabled, the instrument driver stores the setting value into the cache. Hereafter, if the application attempts to set the same value, the driver does not perform instrument I/Os if the cache is valid. By this mechanism, the application performance can be increased.

QueryInstrStatus is a functionality, which queries the instrument for its error register every time after performing instrument settings. If an error has occurred, the instrument driver treats it as an error. (Default is FALSE.) Normally property values being set or parameters passed to a method are validated by the RangeCheck functionality for the value range, however, range checking operations are not always perfect especially on complex instruments. Sending such values to the instrument occasionally generates an instrument error. Another example is a case that some instrument functions are temporarily disabled depending on the instrument status. On these situations, you can check if the instrument has accepted the setting values by querying the instrument for the error register after sending the setting values. When this functionality is set to TRUE, the application performance gets slow. However, these kinds of errors are normally generated when the instrument setting instructions are inappropriate such as by incorrect setting order, and normally it is when debugging time. Once the correct setting instructions are made (or once the instrument driver has stop generating errors), you can switch this functionality to FALSE and then the application performance will be increased.

As a performance-concerned feature other than above, there is the multi-threaded capability. All the IVI instrument drivers are considered for use under multi-threaded environment. In case of IVI-COM drivers, the threading model is stamped as "both" in the registry. IVI drivers will play the best operational performance with both multi-threaded and single-threaded applications.

2- Setup

2-1 .NET Framework Setup

If you develop or run .NET applications, **Microsoft .NET Framework** must be installed first.

If the IVI Shared Components and each of IVI-COM instrument drivers, when being set up, have detected the existence of .NET Framework, the setup program also installs required Primary Interop Assemblies (PIA). The PIAs are required when you use IVI-COM instrument drivers under .NET environment. You will have to re-install the IVI Shared Components and/or an IVI-COM driver to make sure to install the PIA modules, if you have installed them before installing the .NET Framework

2-2 VISA Library Setup

IVI-COM instrument drivers perform instrument I/Os by using the **VISA Library** (VISA COM software). You can use either NI-VISA (VER3.0 or later), Agilent VISA (IO Lib M01.00 or later), or KI-VISA (VER2.2.x or later) as a VISA COM enabled version of the VISA library. Mind that the VISA must be installed prior to installing IVI drivers. As for how to set up the VISA library, refer to the documentation for each version of VISA libraries.

2-3 Kikusui IVI Config Utility Setup

All IVI-COM instrument drivers require **IVI Shared Components**. You can obtain this component at the IVI Foundation web site or at the Kikusui web site as a stand-alone setup program. At the Kikusui web site, you can also obtain the combo version that also includes **Kikusui IVI Config Utility**. This utility software is a tool for configuring interchangability features.

Now we assume that you set up **Kikusui IVI Config Utility With IVI Shared Components**. The setup program, when the install is done, also launches the setup program for IVI Shared Components as need.

The download file at the Kikusui web site (KiIviConfig.exe) is a self-extracting setup program. As you launch the setup program, the Welcome screen shown below will appear.

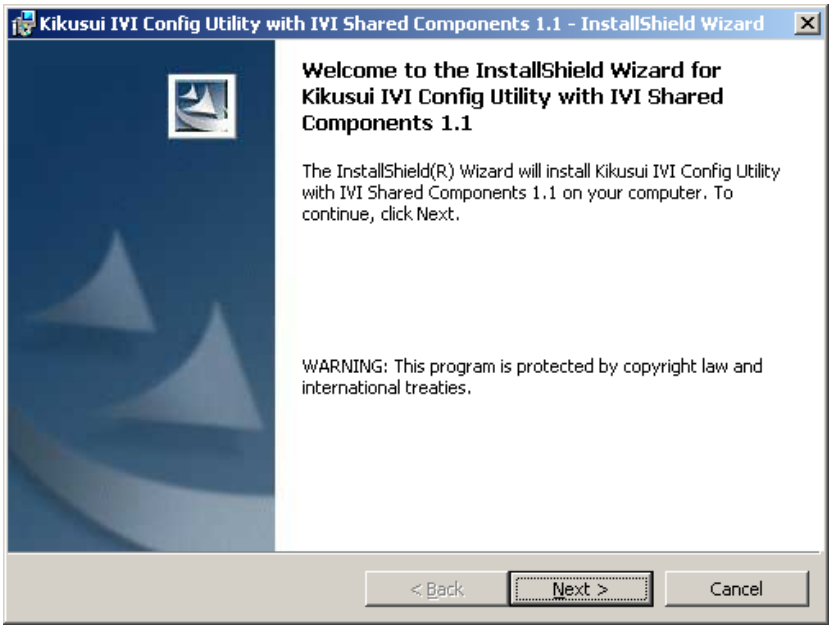

Figure 2-1 Kikusui IVI Config Utility (Welcome Screen)

Clicking the **Next** button will show you the License Agreement. Be sure to read it.

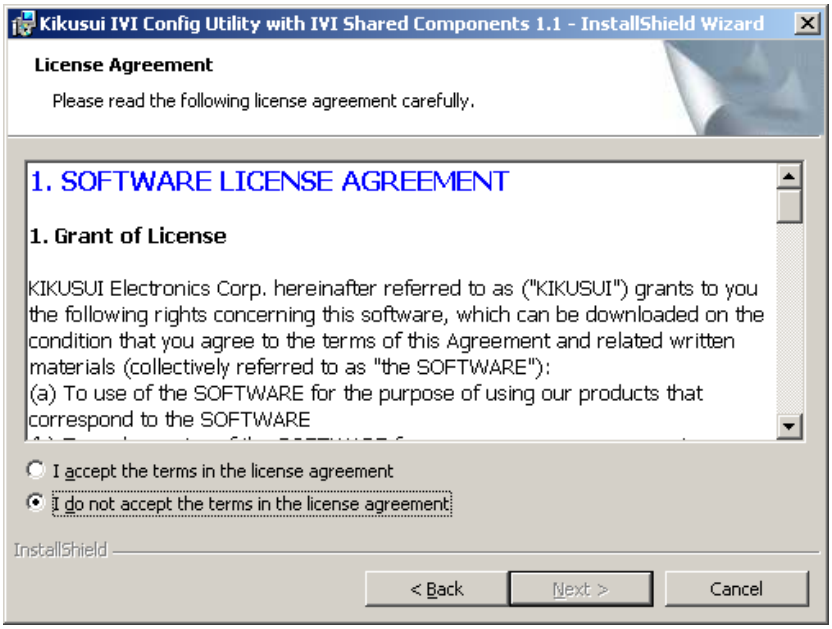

Figure 2-2 Kikusui IVI Config Utility (License Agreement)

Clicking the **Next** button further shows the Ready To Install screen. Click the **Install** button to start the setup.

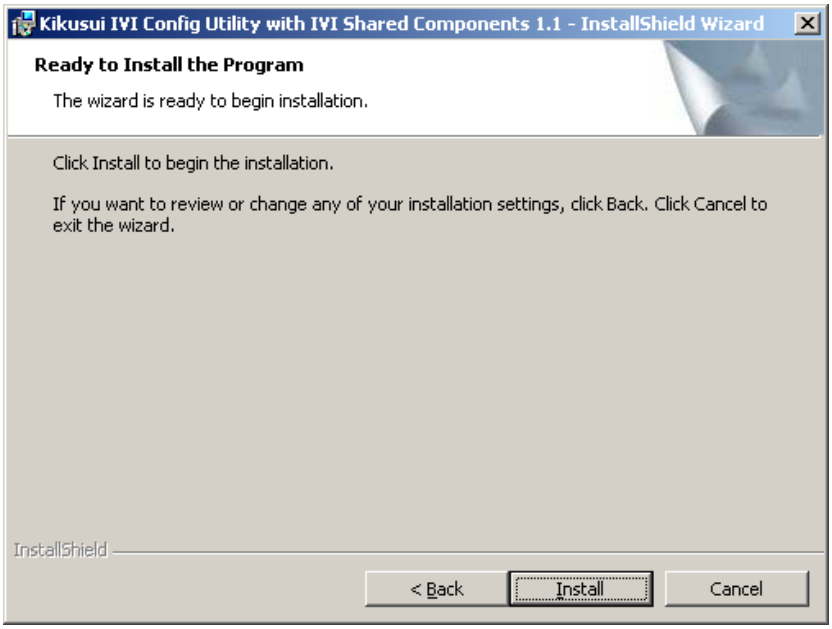

Figure 2-3 Kikusui IVI Config Utility (Ready To Install)

When the setup has completed, the **Launch IVI Shared Components Installer** checkbox will appear. Make sure to check it and then click the **Finish** button. Then continue the setup for the IVI Shared Components.

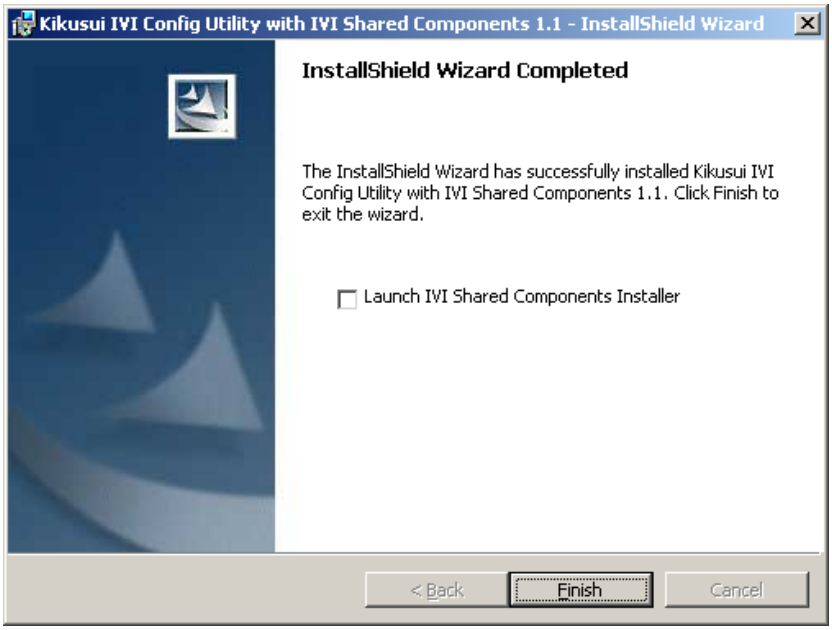

Figure 2-4 Completed Screen

Notes:

The **Launch IVI Shared Components Installer** checkbox will be shown only when the IVI Shared Components is not installed yet or when the version that is already installed is older.

You can also launch the setup program for the IVI Shared Components later. Choosing **[Start] button➔Programs➔IVI➔IVI Shared Components➔Install IVI Shared Components** menu will launch the setup program.

2-4 IVI Shared Components Setup

As you start the setup program for the IVI Shared Components, you will see the Welcome screen.

Figure 2-5 IVI Shared Components (Welcome Screen)

Clicking the **Next** button will show you the Licence Agreement. Be sure to read it.

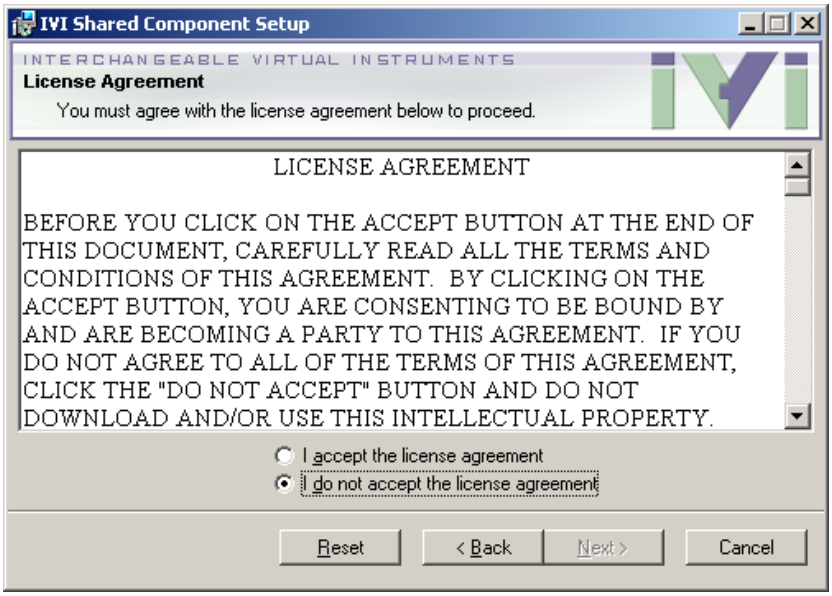

Figure 2-6 IVI Shared Components (License Agreement)

By clicking the **Next** button further, you will see the selection screen for the IVI Standard Root Directory.

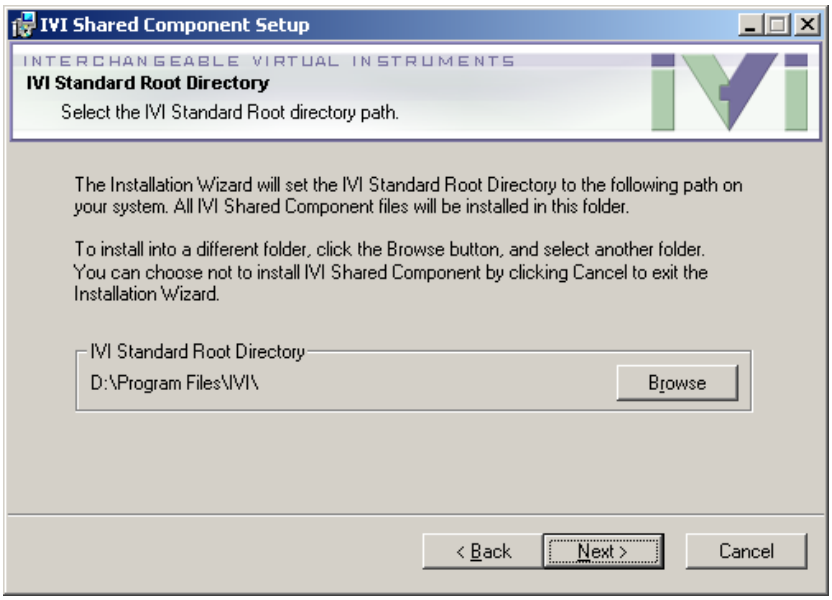

Figure 2-7 IVI Shared Components (IVI Standard Root Directory Selection)

The IVI Shared Components goes to the /Program Files/IVI directory as default. Because the destination directory you decide here is stamped in the registry, all the IVI-COM instrument drivers being installed hereafter also go to that directory. By clicking the **Next** button further, you will see the Ready To Install screen.

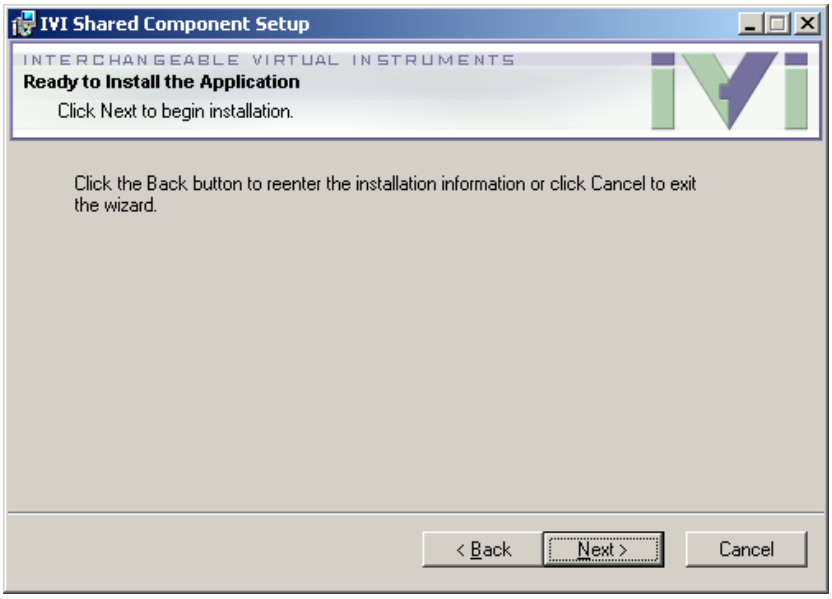

Figure 2-8 IVI Shared Components (Ready To Install)

Click the **Next** button to start the installation.

3- IVI-COM Instrument Driver Setup

Notes:

This guidebook assumes that you use the IVI-COM Kikusui4800 Instrument Drivers (for Kikusui PIA4800 series DC Power Supply Controller). You can also use other versions of IVI-COM instrument drivers in the same manner.

After completing the setup for the VISA Library (VISA COM Software) and the IVI Shared Components, finally you install the IVI-COM instrument driver. Normally IVI-COM instrument drivers provide separate setup programs for each model.

IVI-COM instrument driver files downloaded at the Kikusui web site all have a KikusuiXXXX_1_0_0_0.msi file name. (The numeric parts indicate the version.) The setup program for the IVI-COM Kikusui4800 driver shown in the following picture has the file name Kikusui4800_1_0_1_0.msi. You can immediately execute it as long as Windows Installer 2.0 is already installed on your PC.

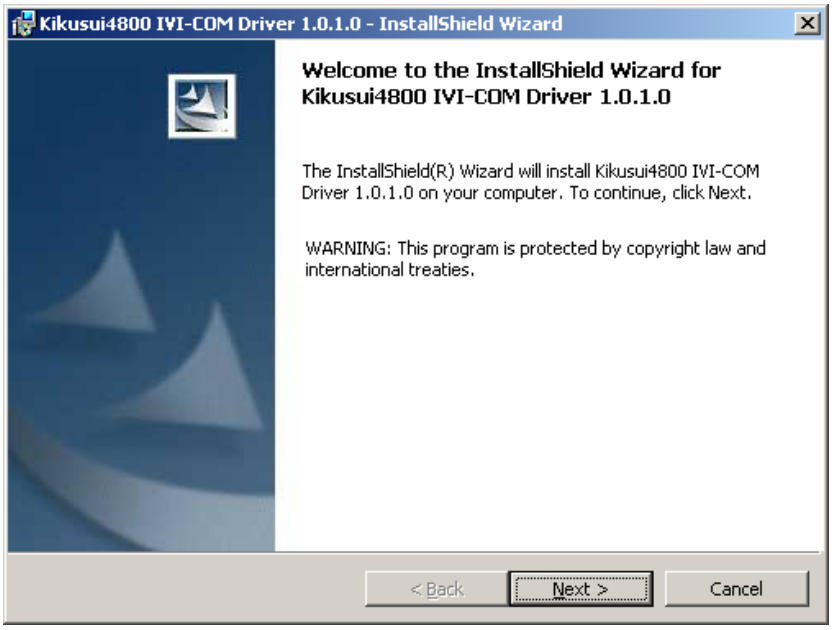

Figure 3-1 IVI-COM Driver (Welcome Screen)

Clicking the **Next** button will show you the License Agreement. Be sure read it.

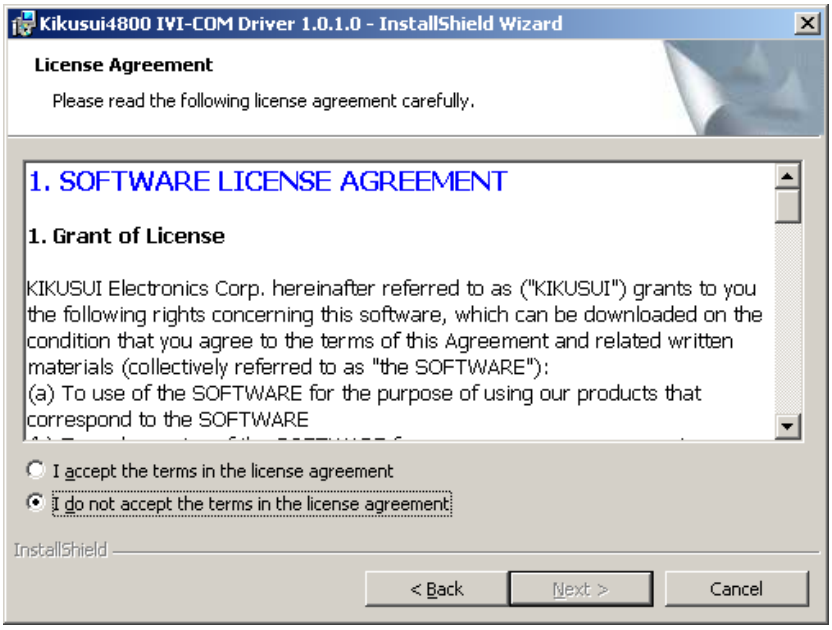

Figure 3-2 IVI-COM Driver (License Agreement)

By clicking the **Next** button, you will see the Customer Information screen.

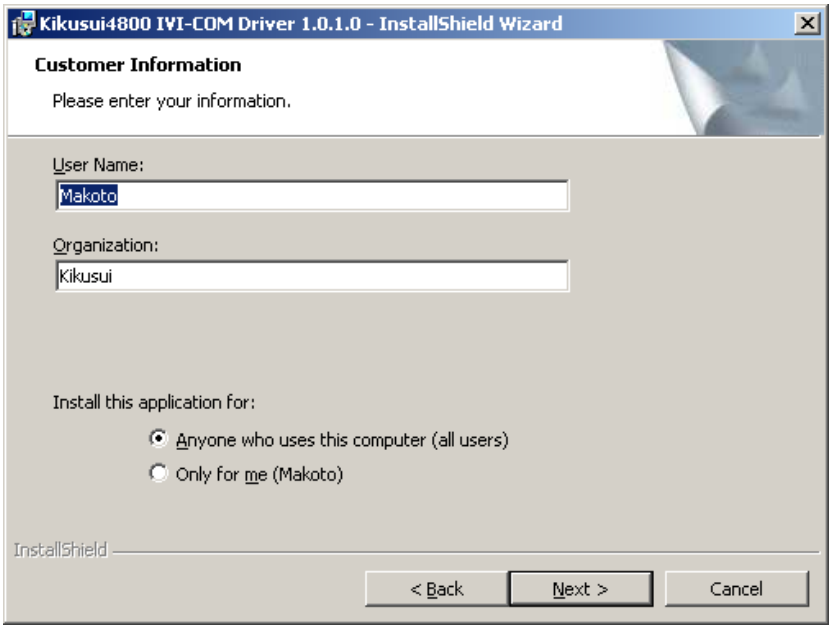

Figure 3-3 IVI-COM Driver (Customer Information Confirmation)

By clicking the **Next** button further, you will see the Setup Type selection screen.

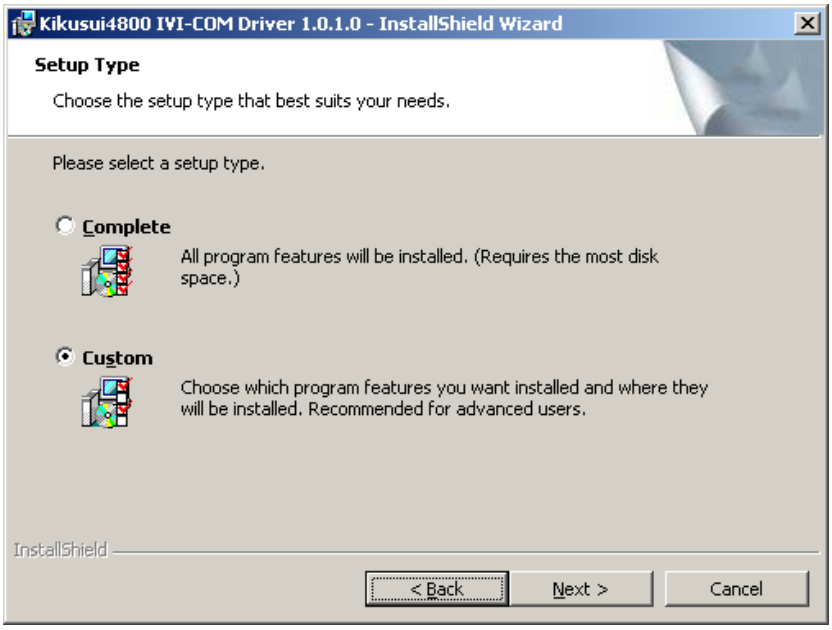

Figure 3-4 IVI-COM Driver (Setup Type Selection)

On the Setup Type selection screen, there are two choices, Complete and Custom. However the IVI-COM instrument drivers provided by Kikusui enforces you to select the **Custom** setup. Clicking the **Next** button show you the Custom Setup screen.

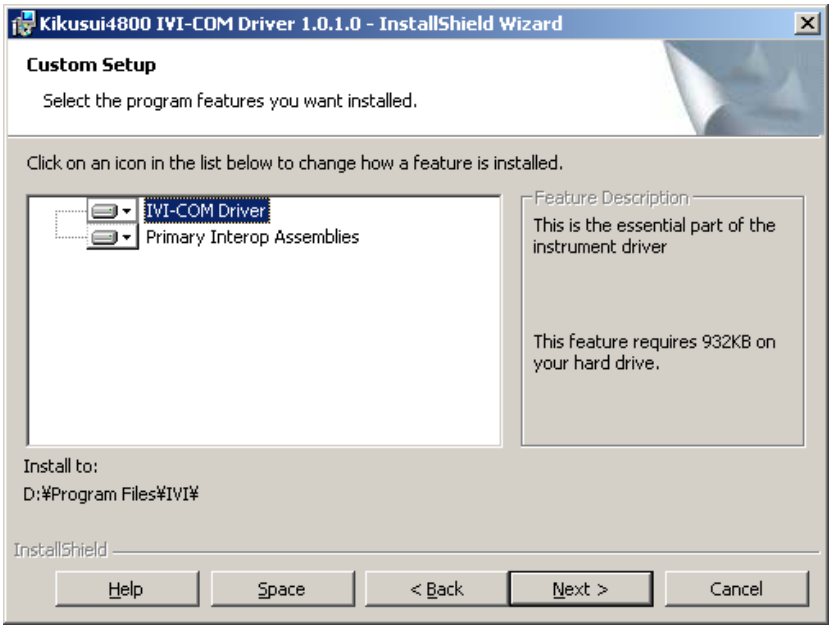

Figure 3-5 IVI-COM Driver (Custom Setup)

On the Custom Setup screen, you can select if the Primary Interop Assemblies are to be installed or not. If the .NET Framework has been detected, the **Primary Interop Assemblies** is selected as default, otherwise the selection is cancelled. The destination directory is represented as **Install To**. Because the IVI Standard Root Directory is already decided by the IVI Shared Components setup, you cannot change it here. Clicking the **Next** button will show you the Ready To Install screen.

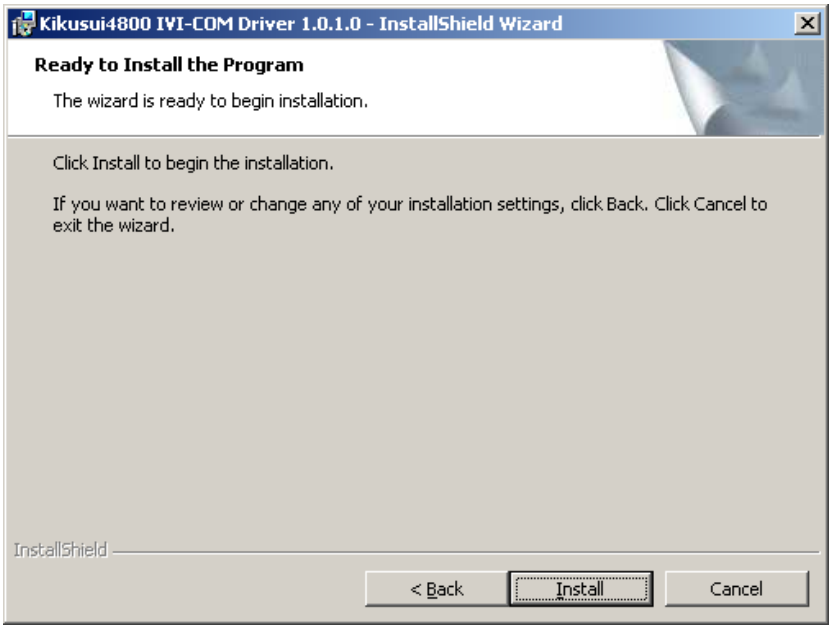

Figure 3-6 IVI-COM Driver (Ready To Install)

Clicking the **Install** button will start the installation. When the installation has completed, a folder is created at [**Start] button➔Programs➔IVI**, thereby you can access the Readme document, online help, and sample programs.

IVI-COM Instrument Driver Programming Guide