IVI Instrument Driver Programming Guide (Setup Edition)

June 2012 Revision 2.0

1- IVI Instrument Driver Overview

1-1 IVI-C vs. IVI-COM

IVI Instrument Driver is an instrumentation middle-ware conforming to the industrial standard that is defined by IVI Foundation (www.ivifoundation.org). Instrument Driver itself is not an application software and unable to execute alone, but provides encapsulation for instrument I/O control that is needed by test & measurement applications.

Based on current IVI specification, an instrument driver is provided as Windows DLL format. Normally, it comes with a specific setup program providing not only DLLs, but online help document and others are also provided. Plus, an instrument driver is provided for each instrument model (or model series) separately.

The IVI specifications define the following 3 types of API specs for instrument driver.

Table 1-1 IVI Driver API specs

type	**description**
IVI-C	What expanded from the legacy VXI Plug&Play Instrument Driver. The drivers of this type are provided as Windows DLL format. Suitable for use with LabWindows/CVI, LabVIEW.
IVI-COM	An in-process COM (Component Object Model) server DLL, which utilizes Microsoft COM technologies. Suitable for use with Microsoft Office VBA, Visual Basic 6, C++. Also suitable for use with .NET languages accessing through interop assemblies.
IVI.NET	Assemblies specific to .NET languages. (Currently we don't provide this type).

1-2 Interchangeability

One of features of IVI instrument drivers is the interchangeability 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.

To utilize the interchangeability 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.

Instrument Class is what categorised by picking up fundamental functionalities of similar instrument types. For example, our KikusuiPwx IVI Instrument Driver(Kikusui PWX series DC supply) belongs to the IviDCPwrclass. Therefore, when an application that uses KikusuiPwx driver is designed considering interchangeability, even if exchanging the instrument driver with AgilentE36xx IVI Instrument Driver(Agilent Technologies E3600 series DC supply) for example, the application can continue working without rebuilding.

As an important notice, an application that considers interchangeability cannot utilize instrument specific functionalities. For example, the IviDCPwr class interface defines generic functions commonly available on DC supplies, but does not define special functions particular to specific instrument models. Furthermore, an application that utilizes interchangeability feature should not use instrument drivers for specific models. Instead, the application must use class interfaces (IVI-COM and IVI,NET cases) or class drivers provided by National Instrument for indirectly accessing the instruments.

Notes:

- Interchangeability feature does not make instrument specific features exchangeable. It makes sense only when the application uses fundamental and common features that are supported by drivers of the instrument class.
- It is possible for the application that considers interchangeability to utilizes instrument specific functionality as need. However you must be careful whether the actual connected instrument actually supports the functionalities..
- Every IVI instrument driver provides pass-through functions that allows to send/receive arbitrary SCPI commands. By using them, it is possible to send/receive commands freely regardless what instrument driver is used.

1-3 Interoperability

IVI Instrument Drivers that do not belong to any instrument classes cannot provide interchangeability, however it is not true that each driver is designed by its own specific architecture. The IVI spec defines standard API style to which every instrument driver shall confirm regardless of belonging to class. For example, Initialize or Initialize With Options functions is used for initiating instrument connection, Reset function is used for resetting the equipment, ErrorQuery function is used for inquiring errors that might have occur, etc. Usability and operations of these inherent capabilities are common for every IVI instrument driver (regardless vendors, models). 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.

1-5 Setup For x86 and x64 Editions

Windows has 2 platform editions -- 32bit(x86) edition and 64bit (x64) edition. The SETUP program of IVI instrument drivers has 2 types for each platform edition..

Table 1-2 Installer Type

type	**description**
x86 Edition SETUP	An installer package for 32bit Windows (XP, Vista, 7). Deploys components required for development and execution of 32bit applications.
x64 Edition SETUP	An installer package for 64bit bit Windows (Vista, 7). Deploys components required for development and execution of 32bit and 64bit applications.

2- Setup

2-1 VISA Library Setup

IVI instrument drivers require VISA library. One of the following VISA libraries must be installed in advanced. Also, do not install multiple versions at the same time.

- NI-VISA VER5.0 or later, or
- Agilent VISA IO Libraries Suite 16.0 or later, or
- KI-VISA VER5.0 or later

By installing VISA library , Microsoft .NET Framework 2.0, IVI Shared Components 2.x, VISA Shared Components, and USBTMC device driver are also automatically installed..

2-2 IVI Instrument Driver Setup

Notes:

- This guidebook shows an example for installing KikusuiPwx IVI Instrument Driver (KIKUSUI PWX series DC supply) on Windows7 (x64 edition). You can install IVI instrument drivers for other models in the same manner.
- Each IVI instrument driver provides separate setup program for each instrument or instrument series.

Every IVI instrument driver downloaded at Kikusui web site has the file name rule such as KikusuiXXXX_1_0_0_0_(x64).msi (or .exe) or similar. (The number part are different depending on the driver version).

Launch the setup program and you will see the following Welcome screen.. If a prompt message asking for elevating to administrator privilege is shown, follow the asking.

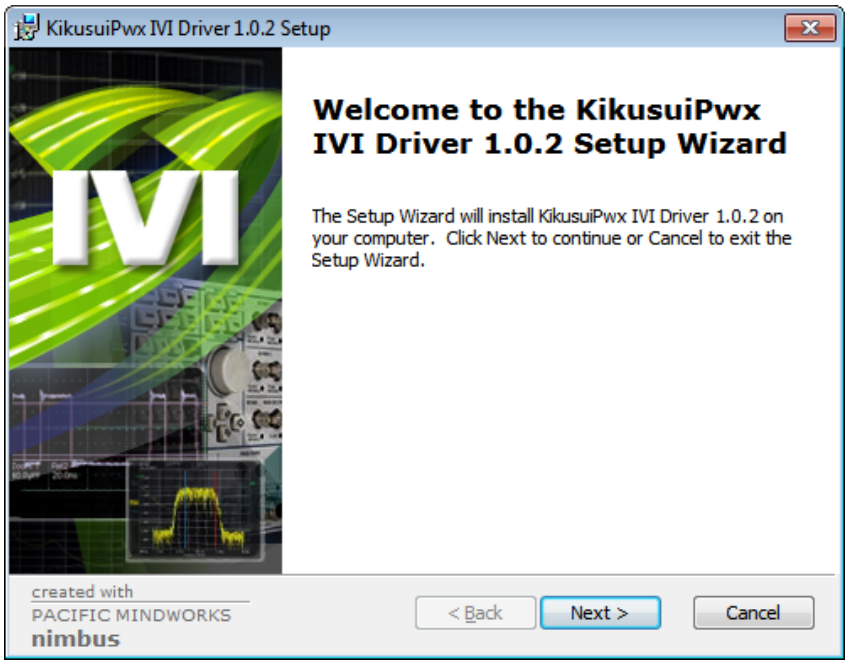

Figure 2-1 IVI Driver Welcome Screen

Clicking the **Next** button will show you the License Agreement. Be sure read it. The license of our IVI instrument driver grants you to install the driver to unlimited number of PCs.

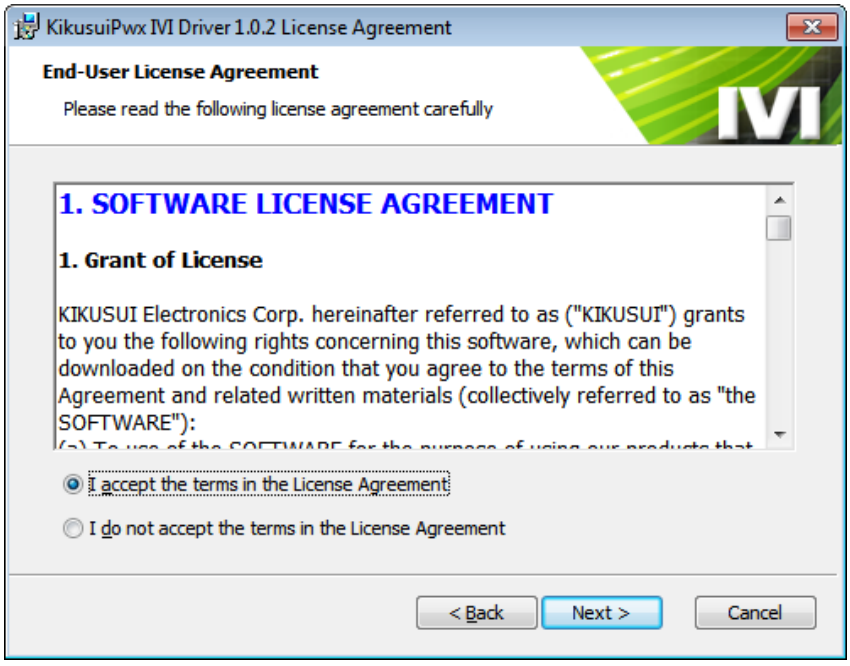

Figure 2-2 IVI Driver License Agreement Screen

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

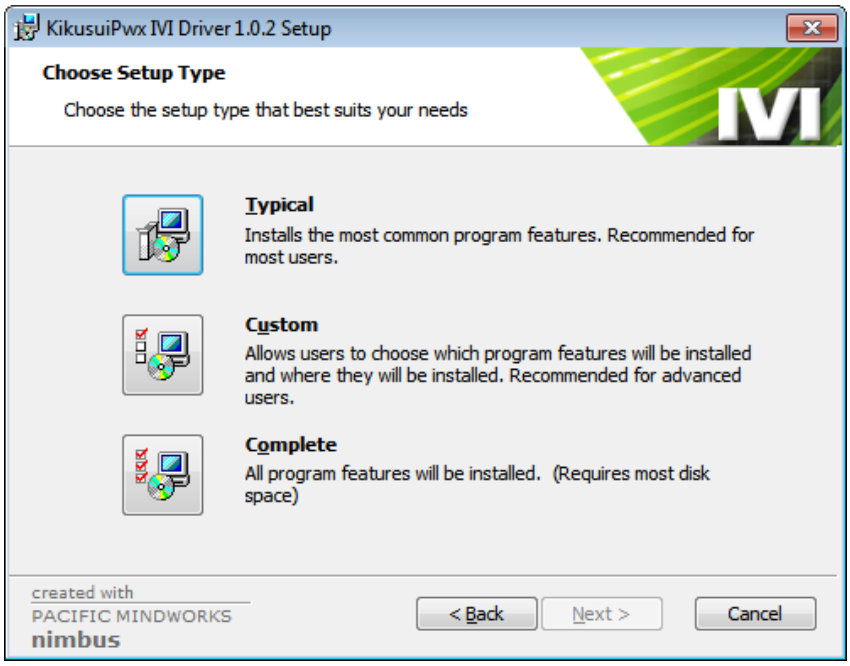

Figure 2-3 IVI Driver Choose Setup Type Screen

On the Setup Type screen there are Typical, Custom, and Complete, but here select **Complete**.

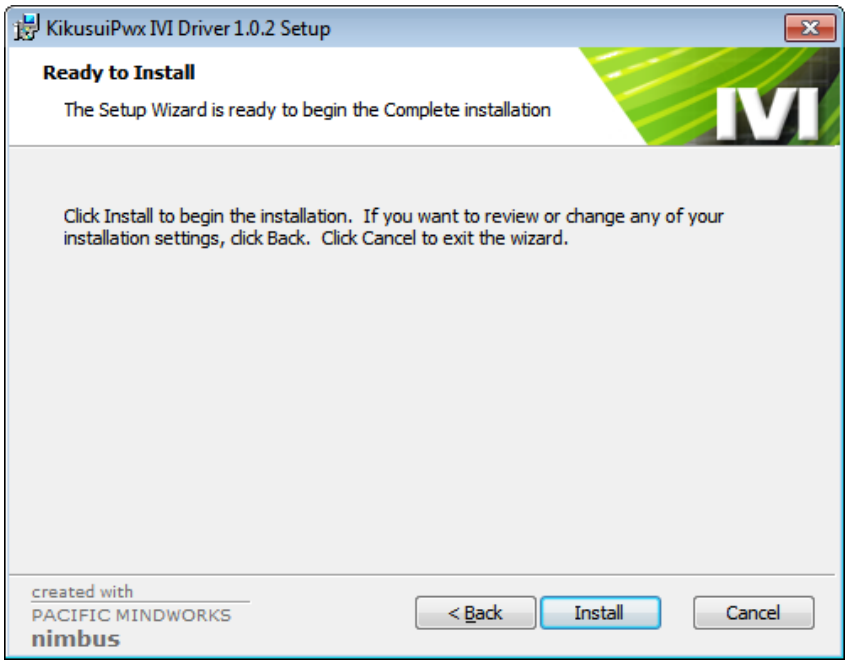

Figure 2-4 IVI Driver Install Starting Screen

Clicking the **Install** button will start the installation. When the installation has completed, shortcut menus will be created at **Start➔All Programs➔Kikusui➔KikusuiPwx**, allowing to access Readme and online help.