

IVI™ DRIVER TOOLSET

Version 1.1

This document contains installation directions and lists changes to the IVI class specifications that might affect your programs.

Contents

Contents	1
Introduction	1
Installation	2
Changes Affecting the IVI API	4
Class Driver Inherent Attribute and Function Changes	5
Specific Driver Inherent Attribute Changes	6
IviDCPwr Changes	7
IviDmm Changes	9
IviFgen Changes	13
IviScope Changes	18
IviSwtch Changes	25
IVI Soft Front Panels	27
Class Simulation Drivers	27
Updates to NI Spy	28

Introduction

The IVI Driver Toolset 1.1 is a maintenance release of the IVI Driver Library 1.0. This release includes necessary updates for conformance with the IVI specifications. The following summary lists the changes between versions 1.0 and 1.1:

- The name of the product changed from IVI Driver Library to IVI Driver Toolset to ensure consistent naming with add-on products for LabVIEW and Measurement Studio, which includes LabWindows/CVI and measurement tools for Visual C++ and Visual Basic.

- All class drivers are updated to conform to revision 2.0 of the IVI Foundation class specifications for the following classes: DMM, function generator, DC power supply, scope, switch. The DC power supply class replaces the power supply class that was part of the previous release. An AC power supply class specification is currently under development in the Foundation, and will be included in a future release of this product contingent upon the specification's completion.
- The IVI Driver Toolset includes improved interchangeability checking capability, detecting and reporting interchange warnings at the time the warning occurs.
- NI-Spy is improved to display attribute names and error names instead of their hexadecimal equivalents.
- In addition to this release of the toolset, National Instruments is distributing a runtime version of the IVI class drivers, which is available on your CD or from the Instrument Driver Network at ni.com

Installation

Consider the following issues before installing the IVI Driver Toolset:

- If you already have the IVI Driver Library 1.0 installed, you must uninstall it first and then proceed with the installation of the IVI Driver Toolset 1.1.
- If you plan to use the IVI Driver Toolset with LabVIEW and/or LabWindows/CVI, make sure that you have already installed LabVIEW version 5.1 or greater and/or LabWindows/CVI version 5.0.1 or greater.
- National Instruments recommends that you close all other applications before you install the IVI Driver Toolset.

To use the IVI Driver Toolset for Windows, you must have the following minimum system requirements:

- Windows 2000/NT/9x or later
- PC using at least a 133 MHz Pentium class or higher microprocessor
- Minimum of 32 MB of memory
- 40 MB of free hard disk space

To install the IVI Driver Toolset, perform the following steps:

1. Insert the IVI Driver Toolset Installation CD into the CD-ROM drive.
2. Wait several seconds. Normally, the CD automatically launches the IVI Driver Toolset Installation CD dialog box. If the IVI Driver Toolset Installation CD dialog box does not appear, click **Start»Run** in the

Windows taskbar and enter `d:\setup` in the input box, where *d* is the letter of your CD-ROM drive. Click **OK** to begin installation.

3. Follow the directions that appear on screen.

Uninstall Considerations

To uninstall the IVI Driver Toolset, use the Add/Remove Programs item located in Control Panel. Use the same method to uninstall the other components installed by the IVI Driver Toolset, such as the IVI Engine, NI-DAQ 6.7, and NI-488.2. To uninstall specific drivers, run the uninstall programs located in the Start menu under the `VXIpnp\Uninstall Drivers` folder.

Installed Components

The IVI Driver Toolset installs the following components:

- IVI Soft Front Panels (VirtualBench 2.6). If you are currently using an earlier version of VirtualBench you must do one of the following:
 - Run the VirtualBench uninstall program.
 - Install IVI Soft Front Panels in a different location by selecting the **Component** button for the **Utilities** option in the IVI Driver Toolset installation program, and then click the **Browse** button located next to the IVI Soft Front Panels option.
- The IVI Engine, the LabWindows/CVI Run-Time Engine, NI-VISA, NI-DAQ, and Measurement & Automation Explorer. Installation disk images for these components are located on the IVI Driver Toolset CD in the following subdirectories:

Component	Drivers Directory
IVI Engine 1.6	<code>iviengine</code>
LabWindows/CVI Run-Time Engine 5.5	<code>cvirte</code>
NI-VISA 2.01	<code>visa</code>
NI-DAQ 6.7	<code>daq</code>
Measurement & Automation Explorer	<code>max</code>

The IVI Driver Toolset CD contains the installation disk image for NI-488.2 for Windows version 1.6 in the `gpiB` subdirectory.

- If you select to install IVI Soft Front Panels, the IVI Driver Toolset installs a minimal installation of NI-DAQ 6.7. For a complete installation of NI-DAQ, run the DAQ Designer in the DAQ directory.

- If LabWindows/CVI is installed on your computer, the IVI Driver Toolset installs the following instrument driver files for each specific driver in the `cvi\instr` directory:
 - Source file (*.c)
 - Include file (*.h)
 - Import library file (*.lib)
 - Function panel files (*.fp and *.sub)
 - Help file (*.hlp)
 - Readme file (*.txt)
- If LabWindows/CVI is not installed on your computer, the IVI Driver Toolset installs the above files in the `VXIplug&play` directories. The instrument driver dynamic link library (*.dll) file is always installed in either the `vxipnp\win95\bin` or `vxipnp\winnt\bin` directory.
- If you use Windows 95 and do not have DCOM installed, you will encounter the following error message upon launching `setup.exe`:
"A required .DLL file, OLEPRO32.DLL, was not found."
Please install DCOM by running the DCOM installer from the `\DCOM98` directory on the IVI Driver Toolset CD.

Changes Affecting the IVI API

Since the initial release of the IVI class specifications by the IVI Foundation in August of 1998, the membership has reviewed those specifications and determined that significant modifications and updates were needed. This update was required to better reflect the behavior of the instruments that the specifications encompass, and to include input from several large instrument vendors that joined the Foundation subsequent to the initial release of the specifications.

As a result of the change in the IVI class specifications, National Instruments has updated our IVI Driver Toolset to conform to the IVI Foundation's new class API specifications. This portion of the release notes summarizes the changes that are incorporated into the class specifications and describes their effect on all the components of the toolset. In addition, this section covers the guidelines to update your applications to use the new version of the toolset, which is compliant with the new IVI class specifications.

Class Driver Inherent Attribute and Function Changes

As a result of the changes in the Inherent Capabilities specified by the IVI Foundation, the following attributes and functions were changed in all class drivers. The following functions and attributes were added, modified, or removed.



Note Identical changes apply to the LabVIEW VIs and properties of the same name.

1. The following attributes were renamed in the class drivers:

Replace	With
<PREFIX>_ATTR_CLASS_PREFIX	<PREFIX>_ATTR_CLASS_DRIVER_PREFIX
<PREFIX>_ATTR_CLASS_MAJOR_VERSION	<PREFIX>_ATTR_CLASS_DRIVER_MAJOR_VERSION
<PREFIX>_ATTR_CLASS_MINOR_VERSION	<PREFIX>_ATTR_CLASS_DRIVER_MINOR_VERSION
<PREFIX>_ATTR_CLASS_REVISION	<PREFIX>_ATTR_CLASS_DRIVER_REVISION
<PREFIX>_ATTR_SPECIFIC_PREFIX	<PREFIX>_ATTR_SPECIFIC_DRIVER_PREFIX
<PREFIX>_ATTR_MODULE_PATHNAME	<PREFIX>_ATTR_SPECIFIC_DRIVER_LOCATOR
<PREFIX>_ATTR_DRIVER_MAJOR_VERSION	<PREFIX>_ATTR_SPECIFIC_DRIVER_MAJOR_VERSION
<PREFIX>_ATTR_DRIVER_MINOR_VERSION	<PREFIX>_ATTR_SPECIFIC_DRIVER_MINOR_VERSION
<PREFIX>_ATTR_DRIVER_REVISION	<PREFIX>_ATTR_SPECIFIC_DRIVER_REVISION

2. The following attributes were added to the class drivers:

<PREFIX>_ATTR_CLASS_DRIVER_VENDOR
<PREFIX>_ATTR_CLASS_DRIVER_DESCRIPTION
<PREFIX>_ATTR_CLASS_DRIVER_CLASS_SPEC_MAJOR_VERSION
<PREFIX>_ATTR_CLASS_DRIVER_CLASS_SPEC_MINOR_VERSION
<PREFIX>_ATTR_SPECIFIC_DRIVER_VENDOR
<PREFIX>_ATTR_SPECIFIC_DRIVER_DESCRIPTION
<PREFIX>_ATTR_SPECIFIC_DRIVER_CLASS_SPEC_MAJOR_VERSION

```

<PREFIX>_ATTR_SPECIFIC_DRIVER_CLASS_SPEC_
    MINOR_VERSION
<PREFIX>_ATTR_INSTRUMENT_FIRMWARE_REVISION
<PREFIX>_ATTR_INSTRUMENT_MANUFACTURER
<PREFIX>_ATTR_INSTRUMENT_MODEL
<PREFIX>_ATTR_SUPPORTED_INSTRUMENT_MODELS
<PREFIX>_ATTR_IO_SESSION_TYPE
    (Not available in LabVIEW)
<PREFIX>_ATTR_IO_SESSION (Not available in LabVIEW)

```

- The following attributes were removed from the class drivers:

```

<PREFIX>_ATTR_ENGINE_MAJOR_VERSION
<PREFIX>_ATTR_ENGINE_MINOR_VERSION
<PREFIX>_ATTR_ENGINE_REVISION

```
- The following functions were added to the class drivers:

```

<prefix>_InitWithOptions
<prefix>_GetSpecificDriverCHandle
    (Not available in LabVIEW)

```

Specific Driver Inherent Attribute Changes

As a result of the changes in the Inherent Capabilities specified by the IVI Foundation, the following attributes were changed in all specific drivers. The following attributes were added, modified or removed.



Note Identical changes apply to the LabVIEW VIs and properties of the same name.

- The following attributes were renamed in the specific drivers:

Replace	With
<PREFIX>_ATTR_SPECIFIC_PREFIX	<PREFIX>_ATTR_SPECIFIC_DRIVER_PREFIX
<PREFIX>_ATTR_DRIVER_MAJOR_VERSION	<PREFIX>_ATTR_SPECIFIC_DRIVER_MAJOR_VERSION
<PREFIX>_ATTR_DRIVER_MINOR_VERSION	<PREFIX>_ATTR_SPECIFIC_DRIVER_MINOR_VERSION
<PREFIX>_ATTR_DRIVER_REVISION	<PREFIX>_ATTR_SPECIFIC_DRIVER_REVISION

2. The following attributes were added to the specific drivers:

```
<PREFIX>_ATTR_SPECIFIC_DRIVER_VENDOR
<PREFIX>_ATTR_SPECIFIC_DRIVER_DESCRIPTION
<PREFIX>_ATTR_SPECIFIC_DRIVER_CLASS_SPEC_
    MAJOR_VERSION
<PREFIX>_ATTR_SPECIFIC_DRIVER_CLASS_SPEC_
    MINOR_VERSION
<PREFIX>_ATTR_INSTRUMENT_FIRMWARE_REVISION
<PREFIX>_ATTR_INSTRUMENT_MANUFACTURER
<PREFIX>_ATTR_INSTRUMENT_MODEL
<PREFIX>_ATTR_SUPPORTED_INSTRUMENT_MODELS
<PREFIX>_ATTR_IO_SESSION_TYPE
    (Not available in LabVIEW)
<PREFIX>_ATTR_IO_SESSION (Not available in LabVIEW)
```

3. The following attributes were removed from the specific drivers:

```
<PREFIX>_ATTR_ENGINE_MAJOR_VERSION
<PREFIX>_ATTR_ENGINE_MINOR_VERSION
<PREFIX>_ATTR_ENGINE_REVISION
```

IviDCPwr Changes

In the initial release of the IVI specifications, both AC and DC power supplies were grouped together in one class called IviPower. This class specification was very generic and did not represent the instruments in that class very well. As a result, the IviPower specification was obsoleted and replaced by two separate specifications: IviDCPwr and IviACPwr. Currently, only the IviDCPwr specification is finalized and consequently the IVI Driver Toolset contains a class driver that complies with this specification.

Example Programs

The example programs that ship with the toolset are updated to reflect these changes. The LabVIEW example program for the IviDCPwr class is found in the `labview\examples\instr\iviclass.llb` directory:

- **IviDCPwr [MSR]- Output DC Volts.vi**—Instructs the instrument to output a DC voltage. The VI then checks to see if an overvoltage or overcurrent condition occurred.

The LabWindows/CVI sample program is found in the `cvi\samples\iviclass` directory:

- **IviDCPwr.prj**—Demonstrates basic use of the IviDCPwr class driver by instructing the instrument to output a DC voltage.

Guidelines for Updating Your Application

Since the IviDCPwr class is a new class in this release of the toolset and it does not map very well to the old IviPower class, we do not recommend that you update your application. Although it might be possible to do so in simple cases, it is generally not recommended. To bring your application up to date, we recommend that you familiarize yourself with the new IviDCPwr API by reading the appropriate sections in the *IVI Driver Toolset User Manual* and the online help, and then recreate your application using the new drivers.



Note Since many of the inputs to the LabVIEW driver are modified from constants to enumerated types and vice versa, it is imperative that you recreate front panel controls or indicators, or block diagram constants that you created using the automatic “Create Control” (or Indicator or Constant). In some cases, even the enumerations and values of the enumerated types are changed, so even if a control looks identical, it might not have the same values.

Instrument-Specific Drivers Affected

To use these new class drivers you also must use updated instrument-specific drivers as well. These are updated as part of the installation of the IVI Driver Toolset. The old instrument-specific drivers are updated to work with the new IviDCPwr class driver. Below is a list of all current IviDCPwr drivers:

Manufacturer	Driver	Comments
American Reliance	arpps	Replaces IviPower obsolete
Hewlett Packard	hp6610xa	Replaces IviPower obsolete
Hewlett Packard	hp662xa	Replaces IviPower obsolete
Hewlett Packard	hp66xxa	Replaces IviPower obsolete
Hewlett Packard	hp66xxbc	New
Hewlett Packard	hpe363xa	Replaces IviPower obsolete
Keithley	ke230x	New, split from IviPower obsolete ke230x
Keithley	ke230y	New, split from IviPower obsolete ke230x
Stanford Research	srps3xx	Replaces IviPower obsolete
Tektronix	tk252xg	Replaces IviPower obsolete

Co-Existing IviPower with IviDCPwr

National Instruments discourages installing the new IviDCPwr class on a system that contains the obsolete IviPower class. Under such circumstances, both the IviDCPwr drivers and the IviPower drivers might behave unexpectedly. However, it might be necessary for the two classes to coexist on the same system. If this is the case, the following issues must be carefully considered for the drivers to operate properly.

- Most of the new IviDCPwr-specific driver installers install files with the same name as the obsolete IviPower drivers. Because of this, running a new IviDCPwr-specific driver installer replaces the corresponding IviPower driver already installed on the system. Therefore, before installing a new IviDCPwr-specific driver, the corresponding IviPower driver must be renamed.
- The `ivi.ini` file contains configuration information on all the drivers installed in the system. Since most of the new IviDCPwr-specific drivers have the same prefixes as the obsolete IviPower-specific drivers, installing a new IviDCPwr-specific driver overwrites the corresponding IviPower-specific driver's configuration information in the `ivi.ini` file. To prevent this, either rename the `ivi.ini` file before installing IVI Driver Toolset 1.1 or rename the sections in `ivi.ini` that contains configuration information for the obsolete IviPower-specific drivers.
- Before you install IVI Driver Toolset 1.1, IVI Driver Library 1.0 must be uninstalled from the system. Uninstalling IVI Driver Library 1.0 deletes the IviPower class driver from the system. To preserve the IviPower class driver, copy the `ivipower.dll` file into the `VXIPNP\Win95 (or WinNT)\BIN` directory.

IviDmm Changes

As a result of the changes that resulted in the IviDmm 2.0 specification, the following attributes and functions were added, modified or removed in the IviDmm class driver.



Note Identical changes apply to the LabVIEW VIs and properties of the same name.

1. The IviDmm 2.0 specifications no longer define a “miscellaneous extension group.” Instead, an extension capability group is created for each attribute that existed in the “miscellaneous extension group.” A high-level function is provided to access each attribute. You can change your application to use these high-level functions to replace

set/get calls for these attributes. The affected attributes and the corresponding function are as follows:

Replace	With
IVIDMM_TRIGGER_SLOPE	IviDmm_ConfigureTriggerSlope
IVIDMM_AUTO_ZERO	IviDmm_ConfigureAutoZeroMode
IVIDMM_AUTO_RANGE_VALUE	IviDmm_GetAutoRangeValue
IVIDMM_POWERLINE_FREQ	IviDmm_ConfigurePowerLineFrequency

- Diode, Continuity, Conductance, and Charge extended measurement types have been removed from the IviDmm 2.0 specifications. You must substitute an occurrence of the following values in your code by the specific driver-defined values:

IVIDMM_VAL_DIODE
 IVIDMM_VAL_COULOMBS
 IVIDMM_VAL_CONTINUITY
 IVIDMM_VAL_SIEMENS

- Substitute the following defined values for IVIDMM_POWERLINE_FREQ by the explicit real value:

Replace	With
IVIDMM_VAL_50_HERTZ	50.0
IVIDMM_VAL_60_HERTZ	60.0
IVIDMM_VAL_400_HERTZ	400.0

- You must make several changes to the IviDmm Base Capabilities group. First, substitute a call to IviDmm_Configure with IviDmm_ConfigureMeasurement and IviDmm_ConfigureACBandwidth. ConfigureMeasurement specifies resolution in **Absolute Digits** rather than digits of precision. You must first convert the resolution in digits of precision to absolute units using the following formulas:

$$Digits = \log_{10} \left(\frac{MeasurementRange}{Absolute Resolution} \right)$$

$$Absolute Resolution = \left(\frac{MeasurementRange}{(10)^{Digits}} \right)$$

For example, substitute the first line of code with the next two calls:

```
IviDmm_Configure (vi, IVIDMM_VAL_AC_VOLTS, 100.0,  
    IVIDMM_VAL_4_5_DIGITS, 20.0, 1000.0);  
IviDmm_ConfigureMeasurement (vi,  
    IVIDMM_VAL_AC_VOLTS, 100.0, 0.001);  
IviDmm_ConfigureACBandwidth (vi, 20.0, 1000.0);
```

Next, change all occurrences of `ATTR_RESOLUTION` to `ATTR_RESOLUTION_ABSOLUTE`. Verify that the value you set this attribute to is in absolute units of precision.

- You only call `IviDmm_ConfigureACBandwidth` if your measurement function is set to an AC Volts or Current measurement.
 - The resolution parameter is ignored if the value of the ‘range’ parameter is `IVIDMM_VAL_AUTO_RANGE_ON`.
5. Fahrenheit temperature measurements are removed from the `IviDmm 2.0` specifications. To obtain readings in Fahrenheit, you must use the `IVIDMM_VAL_TEMPERATURE` defined value for the `IVIDMM_ATTR_FUNCTION` attribute. After you use one of the measurement functions to obtain a reading, convert the result from degrees Celsius to Fahrenheit in your application.
 6. Substitute all occurrences of `IVIDMM_VAL_SW_TRIG_FUNC` with `IVIDMM_VAL_SOFTWARE_TRIG`. This is the new defined value for a software trigger source for the `ATTR_TRIGGER_SOURCE` and `ATTR_SAMPLE_TRIGGER` attributes. Change all calls to `IviDmm_SendSWTrigger` to `IviDmm_SendSoftwareTrigger`.
 7. The `IviDmm_CalculateAccuracy` function is removed from the class specifications. You must perform accuracy calculations in your application or call the specific driver’s `CalculateAccuracy` function if it implements one. The following error codes also are removed. You must not check for these errors in your code:

```
IviDmm_ERROR_ACCURACY_UNKNOWN  
IviDmm_ERROR_ACCURACY_UNKNOWN_WHILE_AUTORANGING
```

Example Programs

The example programs that ship with the toolset are updated to reflect these changes. The LabVIEW example programs for the `IviDmm` class is found in the `labview\examples\instr\iviclass.11b` directory:

- **IviDmm - Single-Point Measurement.vi**—Uses immediate triggering to read a single measurement from the instrument.
- **IviDmm [MP] - Multi-point Measurement.vi**—Reads multiple measurements from the instrument based on multiple triggers and multiple measurements per trigger. This VI uses immediate triggering for both the measurement trigger and the sample trigger.

The LabWindows/CVI sample program is found in the `cvi\samples\iviclass` directory:

- **ividdmm.prj**—Demonstrates basic use of the IviDmm class driver. This project uses immediate triggering to read a single measurement from the instrument.

Updating Your Application

To update your application to be compliant with the new version of the IVI Driver Toolset, you must start by familiarizing yourself with the various changes that are implemented in the driver by reading the previous section. Once you do that, you can assess what functions or attributes you use in your application that have either been modified or removed from the driver. You then need to substitute each of those functions with the appropriate replacement(s). Here is an example of a code snippet update to the new IviDmm class driver:

```
IviDmm_Init ("Logical_Name", VI_TRUE, VI_TRUE, &dmm);
IviDmm_Configure (dmm, IVIDMM_AC_VOLTS, 100.0,
    IVIDMM_VAL_4_5_DIGITS, 20.0, 1000.0);
IviDmm_Read (dmm, MaxTime, &measurement);
IviDmm_Close (dmm);
```

By comparing the code above to the changes listed earlier in this section, notice that the `IviDmm_Configure` function is replaced with `IviDmm_ConfigureMeasurement` and `IviDmm_ConfigureACBandwidth` and that you must convert the resolution into absolute units. Therefore, modify the code above to replace the `IviDmm_Configure` function with the following:

```
IviDmm_ConfigureMeasurement (dmm, IVIDMM_AC_VOLTS,
    100.0, 0.001);
IviDmm_ConfigureACBandwidth (dmm, 20.0, 1000.0);
```



Note Since many of the inputs to the LabVIEW driver are modified from constants to enumerated types and vice versa, it is imperative that you recreate front panel controls or indicators, or block diagram constants that you created using the automatic “Create Control” (or Indicator or Constant). In some cases, even the enumerations and values of the enumerated types are changed, so even if a control looks identical, it might not have the same values.

Instrument-Specific Drivers Affected

If you use these new class drivers, you also must use updated instrument-specific drivers. These are updated as part of the installation of the IVI Driver Toolset. All of the old instrument-specific drivers are

updated to work with the new IviDmm class driver. Below is a list of all current IviDmm drivers:

Manufacturer	Driver	Comments
Fluke	fl45*	Updated to IviDmm specification version 2.0
Fluke	fl8840a	Updated to IviDmm specification version 2.0
Hewlett Packard	hp34401a	Updated to IviDmm specification version 2.0
Hewlett Packard	hp34420a	New
Hewlett Packard	hp3458a	Updated to IviDmm specification version 2.0
Hewlett Packard	hpe1410a	New
Hewlett Packard	hpe1411b	Updated to IviDmm specification version 2.0
Keithley	ke2000	Updated to IviDmm specification version 2.0
Fluke/Philips	pm2534	Updated to IviDmm specification version 2.0
Tektronix	tktxdmm	Updated to IviDmm specification version 2.0
Wavetek	wt1362s	Updated to IviDmm specification version 2.0

* The prefix for this driver was changed from uppercase FL45 to lowercase fl45 for consistency with all other drivers. All direct calls to this driver must be corrected. This change does not affect the LabVIEW driver.

IviFgen Changes

As a result of the changes in the IviFgen 2.0 specification, the following attributes and functions were added, modified, or removed in the IviFgen class driver.



Note Identical changes apply to the LabVIEW VIs and properties of the same name.

1. Replace all calls to `IviFgen_EnableOutput` and `IviFgen_DisableOutput` with calls to `IviFgen_ConfigureOutputEnabled`.

Replace	With
<code>IviFgen_EnableOutput(vi, ch)</code>	<code>IviFgen_ConfigureOutputEnabled(vi, ch, VI_TRUE);</code>
<code>IviFgen_DisableOutput(vi, ch)</code>	<code>IviFgen_ConfigureOutputEnabled(vi, ch, VI_FALSE);</code>

- When accessing the `IVIFGEN_ATTR_OUTPUT_IMPEDANCE` attribute (either from the `IviFgen_Set`, `Get`, or `Check` attribute functions or from `IviFgen_ConfigureOutputImpedance`), make the following replacements:

Replace	With
<code>IVIFGEN_VAL_50_OHMS</code>	<code>50.0</code>
<code>IVIFGEN_VAL_75_OHMS</code>	<code>75.0</code>
<code>IVIFGEN_VAL_1_MEGA_OHM</code>	<code>0.0</code>

- When accessing the `IVIFGEN_ATTR_REF_CLOCK_SOURCE` attribute (either from the `IviFgen_Set`, `Get`, or `Check` attribute functions or from `IviFgen_ConfigureRefClockSource`), make the following replacements:

Replace	With
<code>IVIFGEN_VAL_INTERNAL</code>	<code>IVIFGEN_VAL_REF_CLOCK_INTERNAL</code>
<code>IVIFGEN_VAL_EXTERNAL</code>	<code>IVIFGEN_VAL_REF_CLOCK_EXTERNAL</code>
<code>IVIFGEN_VAL_RTSM_CLOCK</code>	<code>IVIFGEN_VAL_REF_CLOCK_RTSM_CLOCK</code>

- Changes to the trigger model:

The `IVIFGEN_ATTR_OPERATION_MODE` attribute is added to the fundamental capabilities group. Because it is part of the fundamental capabilities group, you must set this attribute in your application. If your application does not trigger the function generator, set this attribute to `IVIFGEN_VAL_OPERATE_CONTINUOUS`. You can then remove all calls to `IviFgen_ConfigureTrigger` used to set the trigger source to immediate.

If your application triggers the function generator, set this attribute to `IVIFGEN_VAL_OPERATE_BURST`. Replace calls to `IviFgen_ConfigureTrigger` with calls to `IviFgen_ConfigureTriggerSource` and `IviFgen_ConfigureBurstCount`.

The `IVIFGEN_ATTR_TRIGGER_COUNT` attribute is replaced with the `IVIFGEN_ATTR_BURST_COUNT` attribute.

The defined value `IVIFGEN_VAL_GENERATE_CONTINUOUS` for the `IVIFGEN_ATTR_TRIGGER_COUNT` attribute is removed. To generate a signal continuously, set the `IVIFGEN_ATTR_OPERATION_MODE` attribute to `IVIFGEN_VAL_OPERATE_CONTINUOUS`. To generate a signal continuously after receiving a trigger, set the

IVIFGEN_ATTR_BURST_COUNT attribute to the instrument-specific defined value for continuous generation.

- Replace all calls to `IviFgen_EnableAM` and `IviFgen_DisableAM` with calls to `IviFgen_ConfigureAMEnabled` and `IviFgen_ConfigureAMSource`.

Replace	With
<code>IviFgen_EnableAM(vi, ch, src)</code>	<code>IviFgen_ConfigureAMEnabled(vi, ch, VI_TRUE);</code> <code>IviFgen_ConfigureAMSource(vi, ch, src);</code>
<code>IviFgen_DisableAM(vi, ch)</code>	<code>IviFgen_ConfigureAMEnabled(vi, ch,</code> <code>VI_FALSE);</code>

- When accessing the `IVIFGEN_ATTR_AM_SOURCE` attribute (either from the `IviFgen_Set-`, `Get-`, or `Check Attribute` functions or from `IviFgen_ConfigureAMSource`), make the following replacements:

Replace	With
<code>IVIFGEN_VAL_INTERNAL</code>	<code>IVIFGEN_VAL_AM_INTERNAL</code>
<code>IVIFGEN_VAL_EXTERNAL</code>	<code>IVIFGEN_VAL_AM_EXTERNAL</code>

- When accessing the `IVIFGEN_ATTR_AM_INTERNAL_WAVEFORM` attribute (either from the `IviFgen_Set-`, `Get-`, or `Check Attribute` functions or from `IviFgen_ConfigureAMInternal`), make the following replacements:

Replace	With
<code>IVIFGEN_VAL_WFM_SINE</code>	<code>IVIFGEN_VAL_AM_INTERNAL_SINE</code>
<code>IVIFGEN_VAL_WFM_SQUARE</code>	<code>IVIFGEN_VAL_AM_INTERNAL_SQUARE</code>
<code>IVIFGEN_VAL_WFM_TRIANGLE</code>	<code>IVIFGEN_VAL_AM_INTERNAL_TRIANGLE</code>
<code>IVIFGEN_VAL_WFM_RAMP_UP</code>	<code>IVIFGEN_VAL_AM_INTERNAL_RAMP_UP</code>
<code>IVIFGEN_VAL_WFM_RAMP_DOWN</code>	<code>IVIFGEN_VAL_AM_INTERNAL_RAMP_DOWN</code>

- Replace all calls to `IviFgen_EnableFM` and `IviFgen_DisableFM` with calls to `IviFgen_ConfigureFMEnabled` and `IviFgen_ConfigureFMSource`.

Replace	With
IviFgen_EnableFM(vi, ch, src)	IviFgen_ConfigureFMEnabled(vi, ch, VI_TRUE); IviFgen_ConfigureFMSource(vi, ch, src);
IviFgen_DisableFM(vi, ch)	IviFgen_ConfigureFMEnabled(vi, ch, VI_FALSE);

9. When accessing the IVIFGEN_ATTR_FM_SOURCE attribute (either from the IviFgen_Set-, Get-, or Check Attribute functions or from IviFgen_ConfigureFMSource), make the following replacements:

Replace	With
IVIFGEN_VAL_INTERNAL	IVIFGEN_VAL_FM_INTERNAL
IVIFGEN_VAL_EXTERNAL	IVIFGEN_VAL_FM_EXTERNAL

10. When accessing the IVIFGEN_ATTR_FM_INTERNAL_WAVEFORM attribute (either from the IviFgen_Set-, Get-, or Check attribute functions or from IviFgen_ConfigureFMInternal), make the following replacements:

Replace	With
IVIFGEN_VAL_WFM_SINE	IVIFGEN_VAL_AM_INTERNAL_SINE
IVIFGEN_VAL_WFM_SQUARE	IVIFGEN_VAL_AM_INTERNAL_SQUARE
IVIFGEN_VAL_WFM_TRIANGLE	IVIFGEN_VAL_AM_INTERNAL_TRIANGLE
IVIFGEN_VAL_WFM_RAMP_UP	IVIFGEN_VAL_AM_INTERNAL_RAMP_UP
IVIFGEN_VAL_WFM_RAMP_DOWN	IVIFGEN_VAL_AM_INTERNAL_RAMP_DOWN

11. Substitute all occurrences of IVIFGEN_VAL_SW_TRIG_FUNC with IVIFGEN_VAL_SOFTWARE_TRIG. This is the new defined value for a software trigger source for the IVIFGEN_ATTR_TRIGGER_SOURCE attribute. Change all calls to IviFgen_SendSWTrigger to IviFgen_SendSoftwareTrigger.
12. The trigger source value IVIFGEN_VAL_GPIB_GET is removed. Replace all instances of it with the instrument-specific defined value.
13. The IviFgen_InitiateGeneration and IviFgen_AbortGeneration functions no longer must be called. If your application does not need to explicitly start and stop signal generation, you do not need to call these functions. However, if your

application does call them, you do *not* need to remove them for your application to continue behaving correctly.

Example Programs

The example programs that ship with the toolset are updated to reflect these changes. The following LabVIEW example programs for the `IviFgen` class are found in the `labview\examples\instr\iviclass.llb` directory:

- **IviFgen [ARB] - Generate Arb Wfm.vi**—Instructs the instrument to generate an arbitrary waveform.
- **IviFgen [SEQ] - Generate Arb Sequence.vi**—Defines a sequence that contains two arbitrary waveforms of equal length and then instructs the instrument to generate the sequence.
- **IviFgen [STD] Generate Std Wfm.vi**—Instructs the instrument to generate a standard waveform.

The LabWindows/CVI sample program is found in the `cvi\samples\iviclass` directory:

- **ivifgen.prj**—Demonstrates basic use of the `IviFgen` class driver by generating a standard waveform.

Guidelines for Updating Your Application

To update your application to be compliant with the new version of the IVI Driver Toolset, you need to start by familiarizing yourself with the various changes that are implemented in the driver by reading the previous section. Once you do that, you can assess what functions or attributes you use in your application that have either been modified or removed from the driver. You then need to substitute each of those functions with the appropriate replacement(s). Here is an example of a code snippet update to the new `IviFgen` class driver:

```
IviFgen_Init ("Logical_Name", VI_TRUE, VI_TRUE, &fgen);
IviFgen_ConfigureStandardWaveform (fgen, chanName,
    IVIFGEN_VAL_WFM_SINE, amplitude, offset, frequency,
    startPhase);
IviFgen_EnableOutput (fgen, chanName);
IviFgen_InitiateGeneration (fgen);
IviFgen_DisableOutput (fgen, chanName);
IviFgen_Close (fgen);
```

By comparing the code above to the changes listed earlier in this section, notice that the `IviFgen_EnableOutput` and `IviFgen_DisableOutput` functions are replaced with `IviFgen_ConfigureOutputEnabled`.

```

Therefore, modify the code above to replace the section that begins with the
IviFgen_EnableOutput function with the following:
IviFgen_ConfigureOutputEnabled (fgen, chanName,
    VI_TRUE);
IviFgen_InitiateGeneration (fgen);
IviFgen_ConfigureOutputEnabled (fgen, chanName,
    VI_FALSE);

```



Note Since many of the inputs to the LabVIEW driver are modified from constants to enumerated types and vice versa, it is imperative that you recreate front panel controls or indicators, or block diagram constants that you created using the automatic “Create Control” (or Indicator or Constant). In some cases, even the enumerations and values of the enumerated types changed, so even if a control looks identical, it might not have the same values.

Instrument-Specific Drivers Affected

If you use these new class drivers, you also must use updated instrument-specific drivers. These are updated as part of the installation of the IVI Driver Toolset. All of the old instrument-specific drivers are updated to work with the new `IviFgen` class driver. Below is a list of all current `IviFgen` drivers:

Manufacturer	Driver	Comments
Fluke	fl6061a	Updated to IviFgen specification version 2.0
Hewlett Packard	hp33120a	Updated to IviFgen specification version 2.0
Stanford Research	srds345	New
Tektronix	tkvx4790	New
Wavetek	wt1375	Updated to IviFgen specification version 2.0
Wavetek	wt1396	Updated to IviFgen specification version 2.0
Wavetek	wt395	Updated to IviFgen specification version 2.0
Yokogawa	ykfg3x0	New

IviScope Changes

As a result of the changes in the IviScope 2.0 specification, the following attributes and functions were changed in the IviScope class driver. The following functions and attributes were added, modified, or removed.



Note Identical changes apply to the LabVIEW VIs and properties of the same name.

1. Replace the `IviScope_ConfigureVertical` with the `IviScope_ConfigureChannel` function calls. The list of parameters remains the same.
2. The name of the `IVISCOPE_ATTR_BANDWIDTH` changed to `IVISCOPE_ATTR_MAX_INPUT_FREQUENCY`. The new attribute name better reflects the original intent of this attribute. The attribute describes the maximum input frequency that the oscilloscope can acquire without attenuating the signal by more than 3 dB. Consequently, the parameter name in the accessor function `IviScope_ConfigureChanCharacteristics` changed from `bandwidth` to `maxInputFrequency`.
3. The following attributes, which defined the position of the first point in the waveform record with respect to the trigger event point, are removed from the specification:

```
IVISCOPE_ATTR_HORZ_RECORD_REF_POSITION
IVISCOPE_ATTR_TRIGGER_DELAY_TIME
```

They have been replaced with the following attribute:

```
IVISCOPE_ATTR_ACQUISITION_START_TIME
```

Consequently, the `IviScope_ConfigureHorizontal` and `IviScope_ConfigureTriggerSource` functions had to change. Both functions are removed from the specification and the instrument drivers. The replacement functions better describe the programmer's approach to the configuration of the oscilloscopes. See the function changes below for detailed descriptions of mentioned changes.

4. The Horizontal and Acquisition subsystems of the oscilloscope instrument driver are replaced with a single subsystem—the acquisition subsystem. The acquisition type attribute remained the main configuration parameter that you should start your configuration procedure with. The accessor function for configuring the `IVISCOPE_ATTR_ACQUISITION_TYPE` changed from `IviScope_ConfigureAcquisition` to `IviScope_ConfigureAcquisitionType`.

Instead of using the `IviScope_ConfigureHorizontal` function, you use the `IviScope_ConfigureAcquisitionRecord` function. The prototypes of the old and new functions are as follows:

```
//Prototype using IviScope ConfigureHorizontal
ViStatus IviScope_ConfigureHorizontal (ViSession vi,
    ViReal64 timePerRecord, ViInt32
    minimumRecordLength, ViReal64 recordRefPosition);
//Prototype of IviScope_ConfigureAcquisitionRecord
ViStatus IviScope_ConfigureAcquisitionRecord
    (ViSession vi, ViReal64 timePerRecord, ViInt32
    minimumRecordLength, ViReal64 acqStartTime);
```

5. The `acqStartTime` parameter specifies the time of the first point in the waveform record relative to the trigger event point. The value of the `acqStartTime` is calculated from the `recordRefPosition`, `timePerRecord`, and `triggerDelayTime` parameters using the following formula:

$$\text{acqStartTime} = \text{triggerDelayTime} - \text{timePerRecord} * \text{recordRefPosition} / 100$$

The description of the triggering subsystem has undergone major changes. Besides the removed attribute

`IVISCOPE_ATTR_TRIGGER_DELAY_TIME`, the trigger type and trigger source attributes are redesigned to better model the actual instruments. Immediate triggering is no longer considered a trigger source, but rather a trigger type, which also is true for AC Line triggering. Software triggering, as described by the IVI specifications as the exclusive triggering mode, does not appear to be implemented in the majority of the oscilloscopes. Therefore, the software trigger is removed as a possible trigger type.

The following values are no longer valid for the `IVISCOPE_ATTR_TRIGGER_SOURCE` attribute:

- `IVISCOPE_VAL_IMMEDIATE`
- `IIVISCOPE_VAL_GPIB_GET`
- `IVISCOPE_VAL_SW_TRIG_FUNC`
- `IVISCOPE_VAL_AC_LINE`

The valid values for the `IVISCOPE_ATTR_TRIGGER_SOURCE` attribute are as follows:

- `IVISCOPE_VAL_EXTERNAL`
- `IVISCOPE_VAL_TTL0-7`
- `IVISCOPE_VAL_ECL0-1`
- `IVISCOPE_VAL_PXI_STAR`
- `IVISCOPE_VAL_RTSI_0-6`

The following table lists invalid and valid values for the `IVISCOPE_ATTR_TRIGGER_SOURCE` attribute:

Replace	With
<code>IVISCOPE_VAL_EDGE</code>	<code>IVISCOPE_VAL_EDGE_TRIGGER</code>
<code>IVISCOPE_VAL_GLITCH</code>	<code>IVISCOPE_VAL_GLITCH_TRIGGER</code>
<code>IVISCOPE_VAL_RUNT</code>	<code>IVISCOPE_VAL_RUNT_TRIGGER</code>
<code>IVISCOPE_VAL_WIDTH</code>	<code>IVISCOPE_VAL_WIDTH_TRIGGER</code>
<code>IVISCOPE_VAL_TV</code>	<code>IVISCOPE_VAL_TV_TRIGGER</code>

Replace	With
IVISCOPE_VAL_STATE	IVISCOPE_VAL_IMMEDIATE_TRIGGER
IVISCOPE_VAL_PATTERN	IVISCOPE_VAL_AC_LINE_TRIGGER

The functions that configure the trigger type and trigger source are reorganized. Common attributes such as `IVISCOPE_ATTR_TRIGGER_COUPLING` are removed from the edge trigger-specific function to a fundamental function that all extension groups can use. The new organization of functions is as follows:

Replace

```
ViStatus IviScope_ConfigureTriggerSource (ViSession
vi, ViConstString triggerSource, ViInt32
triggerType, ViReal64 triggerDelay, ViReal64
holdoff);
```

With

```
ViStatus IviScope_ConfigureTrigger (ViSession vi,
ViInt32 triggerType, ViReal64 holdoff);
```

Replace

```
ViStatus IviScope_ConfigureEdgeTrigger (ViSession
vi, ViReal64 level, ViInt32 triggerCoupling,
ViInt32 slope);
```

With

```
ViStatus IviScope_ConfigureEdgeTriggerSource
(ViSession vi, ViConstString source, ViReal64
level, ViInt32 slope);
```

in conjunction with

```
ViStatus IviScope_ConfigureTriggerCoupling
(ViSession vi, ViInt32 triggerCoupling);
```

Similar changes apply to the other trigger types. Please refer to the IviScope class specification located at www.ivifoundation.org

- The TV Triggering model has changed. The following attributes are removed from the specification and instrument drivers and replaced with a better model:

Replace	With
IVISCOPE_ATTR_TV_TRIGGER_SIGNAL_TYPE	IVISCOPE_ATTR_TV_TRIGGER_SIGNAL_FORMAT
IVISCOPE_ATTR_TV_TRIGGER_FIELD	IVISCOPE_ATTR_TV_TRIGGER_EVENT
IVISCOPE_ATTR_TV_TRIGGER_LINE	IVISCOPE_ATTR_TV_TRIGGER_LINE_NUMBER

The values for these attributes have also changed:

IVISCOPE_ATTR_TV_TRIGGER_SIGNAL_FORMAT supports the same values as IVISCOPE_ATTR_TV_TRIGGER_SIGNAL_TYPE did.

The IVISCOPE_ATTR_TV_TRIGGER_EVENT specifies the type of the TV trigger event. Replace the following values with the new TV types.

Replace	With
IVISCOPE_VAL_TV_FIELD1	IVISCOPE_VAL_TV_EVENT_FIELD1
IVISCOPE_VAL_TV_FIELD2	IVISCOPE_VAL_TV_EVENT_FIELD2
IVISCOPE_VAL_TV_ANY_FIELD	IVISCOPE_VAL_TV_EVENT_ANY_FIELD
Added Value	
IVISCOPE_VAL_TV_EVENT_ANY_LINE	IVISCOPE_VAL_TV_EVENT_LINE_NUMBER

If the application sets the IVISCOPE_ATTR_TV_TRIGGER_EVENT to IVISCOPE_VAL_LINE_NUMBER, then the application also must configure the IVISCOPE_ATTR_TV_TRIGGER_LINE_NUMBER attribute either directly or via the accessor function IviScope_ConfigureTVTriggerLineNumber.

- The value IVISCOPE_VAL_INFINITE is no longer allowed for the IVISCOPE_ATTR_NUM_ENVELOPES and IVISCOPE_ATTR_NUM_AVERAGES attributes. This value is removed from the drivers.
- The defined values for input impedance are removed from the drivers. The application uses the actual real values instead:

Replace	With
IVISCOPE_VAL_50_OHMS	50.0
IVISCOPE_VAL_75_OHMS	75.0
IVISCOPE_VAL_1_MEG_OHM	1.0E+06

Example Programs

The example programs that ship with the toolset are updated to reflect these changes. The LabVIEW example program for the IviScope class is in the labview\examples\instr\iviclass.llb directory:

- IviScope - Acq Wfm Edge Triggered**—Configures the instrument for a single-channel waveform acquisition. It uses edge triggering to trigger the acquisition.

- **IviScope - Acq Dual Wfm Edge Triggered**—Configures the instrument for a dual-channel waveform acquisition. It uses edge triggering on the first channel to trigger the acquisition.

The LabWindows/CVI sample program is found in the `cvi\samples\iviclass` directory:

- **iviscope.prj**—Demonstrates basic use of the IviScope class driver. This project configures the instrument for a single-channel waveform acquisition. It uses edge triggering to trigger the acquisition.

Guidelines for Updating Your Application

To update your application to be compliant with the new version of the IVI Driver Toolset, familiarize yourself with the various changes that are implemented in the driver by reading the previous section. Once you do that, you can assess what functions or attributes you use in your application that have either been modified or removed from the driver. Then, substitute each of those functions with the appropriate replacement(s). Here is an example of a code snippet update to the new IviScope class driver:

```
IviScope_init ("Logical_Name", VI_TRUE, VI_TRUE,
    &scope);
IviScope_ConfigureAcquisition (scope, acqType);
IviScope_ConfigureVertical (scope, channelName,
    vertRange, offset, IVISCOPE_VAL_DC, probeAttn,
    VI_TRUE);
IviScope_ConfigureChanCharacteristics (scope, channel,
    inputImpedance, bandwidth);
IviScope_ConfigureHorizontal (scope, timePerRecord,
    minRecordLen, recordRefPos);
IviScope_ConfigureTriggerSource (scope, triggerSource,
    triggerType, triggerDelay, holdoff);
IviScope_ConfigureEdgeTrigger (vi, level, coupling,
    slope);
IviScope_ReadWaveform (scope, instrumentHandle,
    channelName, waveformSize, maxTime, waveformArray,
    &actPoints, &initialX, &XIncrement);
IviScope_close (scope);
```

By comparing the code above to the changes listed earlier in this section, you notice many changes: the `IviScope_ConfigureAcquisition` function is replaced with `IviScope_ConfigureAcquisitionType`, the `IviScope_ConfigureVertical` function is replaced with `IviScope_ConfigureChannel`, the bandwidth input to the `IviScope_ConfigureChanCharacteristics` function is changed to `maxInputFrequency`, the `IviScope_ConfigureHorizontal` function is replaced with `IviScope_ConfigureAcquisitionRecord` with the new `acqStartTime` input, and the

IviScope_ConfigureTriggerSource and IviScope_ConfigureEdgeTrigger functions are replaced with IviScope_ConfigureTrigger, IviScope_ConfigureEdgeTriggerSource, and IviScope_ConfigureTriggerCoupling. Therefore, replace the code above with the following:

```
IviScope_init ("Logical_Name", VI_TRUE, VI_TRUE,
    &scope);
IviScope_ConfigureAcquisitionType (scope, acqType);
IviScope_ConfigureChannel (scope, channelName,
    vertRange, offset, IVISCOPE_VAL_DC, probeAttn,
    VI_TRUE);
maxInputFrequency = bandwidth ;
IviScope_ConfigureChanCharacteristics (scope, channel,
    inputImpedance, maxInputFrequency);
acqStartTime = triggerDelayTime -
    minRecordLen*recordRefPos/100
IviScope_ConfigureAcquisitionRecord (scope,
    timePerRecord, minRecordLen, acqStartTime);
IviScope_ConfigureTrigger (scope, triggerType,
    holdoff);
IviScope_ConfigureEdgeTriggerSource (scope,
    triggerSource, level, slope);
IviScope_ConfigureTriggerCoupling (vi, coupling);
IviScope_ReadWaveform (scope, instrumentHandle,
    channelName, waveformSize, maxTime, waveformArray,
    &actPoints, &initialX, &XIncrement);
IviScope_close (scope);
```



Note Since many of the inputs to the LabVIEW driver are modified from constants to enumerated types and vice versa, it is imperative that you recreate front panel controls or indicators, or block diagram constants that you created using the automatic “Create Control” (or Indicator or Constant). In some cases, even the enumerations and values of the enumerated types are changed, so even if a control looks identical, it might not have the same values.

Instrument-Specific Drivers Affected

Keep in mind that to use these new class drivers you also must use updated instrument-specific drivers as well. These are updated as part of the installation of the IVI Driver Toolset. All of the old instrument-specific drivers are updated to work with the new IviScope class driver. Below is a list of all current IviScope drivers:

Manufacturer	Driver	Comments
Hewlett Packard	hp548xxa	New
LeCroy	lclc5x4a	Discontinued
Tektronix	tkds30xx	Updated to IviScope specification version 2.0
Tektronix	tktds754	Updated to IviScope specification version 2.0
Tektronix	tktds2x0	Updated to IviScope specification version 2.0
Tektronix	tktds3x0	Updated to IviScope specification version 2.0

IviSwTch Changes

As a result of the changes in the IviSwTch 2.0 specification, the following attributes and functions were added, modified, or removed in the IviSwTch class driver.



Note Identical changes apply to the LabVIEW VIs and properties of the same name.

1. Substitute the following defined values for `IVISWTCH_ATTR_WIRE_MODE` by the explicit real value:

Replace	With
<code>IVISWTCH_VAL_1_WIRE</code>	1
<code>IVISWTCH_VAL_2_WIRE</code>	2
<code>IVISWTCH_VAL_3_WIRE</code>	3
<code>IVISWTCH_VAL_4_WIRE</code>	4

2. Substitute all occurrences of `IVISWTCH_VAL_SW_TRIG_FUNC` with `IVISWTCH_VAL_SOFTWARE_TRIG`. This is the new defined value for a software trigger source for the `ATTR_TRIGGER_INPUT` attribute. Remove all occurrences of `IVISWTCH_VAL_GPIB_GET` since this value is no longer supported for the `ATTR_TRIGGER_INPUT` attribute.

Change all calls to `IviSwTch_SendSWTrigger` to `IviSwTch_SendSoftwareTrigger`.

Example Programs

The example programs that ship with the toolset are updated to reflect these changes. The LabVIEW example program for the `IviSwTch` class is found in the `labview\examples\instr\iviclass.llb` directory:

- **IviSwTch - Path Operation Test.vi**—Interactively connects, disconnects, and checks the path between switch channels.

The LabWindows/CVI sample program is found in the `cvi\samples\iviclass` directory:

- **iviswTch.prj**—Demonstrates basic use of the `IviSwTch` class driver by connecting, disconnecting, and checking the path between switch channels.

Guidelines for Updating Your Application

To update your application to be compliant with the new version of the IVI Driver Toolset, you must start by familiarizing yourself with the various changes that are implemented in the driver by reading the previous section. Once you do that, you can assess what functions or attributes you use in your application that are either modified or removed from the driver. You then need to substitute each of those functions with the appropriate replacement(s). Here is an example of a code snippet update to the new `IviSwTch` class driver:

```
IviSwTch_Init ("Logical_Name", VI_TRUE, VI_TRUE,
    &swTch);
IviSwTch_ConfigureScanList (swTch, scanList,
    IVISWITCH_VAL_BREAK_AFTER_MAKE);
IviSwTch_ConfigureScanTrigger (swTch, scanDelay,
    IVISWITCH_VAL_SOFTWARE_TRIG, IVISWITCH_VAL_NONE);
IviSwTch_InitiateScan (swTch);
IviSwTch_SendSWTrigger (swTch);
IviSwTch_Close (swTch);
```

By comparing the code above to the changes listed earlier in this section, you notice that the `IviSwTch_SendSWTrigger` function is replaced with `IviSwTch_SendSoftwareTrigger`. Therefore, you would modify the code above to replace the `IviSwTch_SendSWTrigger` function with the following:

```
IviSwTch_SendSoftwareTrigger (swTch);
```



Note Since many of the inputs to the LabVIEW driver are modified from constants to enumerated types and vice versa, it is imperative that you recreate front panel controls or indicators, or block diagram constants that you created using the automatic “Create

Control” (or Indicator or Constant). In some cases, even the enumerations and values of the enumerated types are changed, so even if a control looks identical, it might not have the same values.

Instrument-Specific Drivers Affected

To use these new class drivers, you also must use updated instrument-specific drivers as well. These are updated as part of the installation of the IVI Driver Toolset. All of the old instrument-specific drivers are updated to work with the new `IviSwTch` class driver. Below is a list of all current `IviSwTch` drivers:

Manufacturer	Driver	Comments
Ascor	as701x	New
Hewlett Packard	hpe1460a	New
Hewlett Packard	hpe1463a*	Updated to IviSwTch specification version 2.0
Hewlett Packard	hpe8462a	New
Racal	rd126030	Updated to IviSwTch specification version 2.0
Racal	rd126040	Updated to IviSwTch specification version 2.0
* The prefix for this driver was changed from uppercase HPE1463a to lowercase hpe1463a for consistency with all other drivers. All direct calls to this driver must be corrected. This change does not affect the LabVIEW driver.		

IVI Soft Front Panels

The IVI Driver Toolset installs IVI soft front panels, which are applications that enable you to interactively operate your instruments. Use them to ensure that your instruments are operating correctly and to make simple, interactive measurements. These soft front panels are very useful when you want to debug your instruments and interactively control them to verify their functionality. These soft front panels are updated to comply with the new revision of the IVI class API specifications.

Class Simulation Drivers

The IVI Driver Toolset installs a simulation driver for each IVI class driver. The IVI Driver Toolset installs the C source files for the simulation drivers so that you can modify the drivers to meet your specific simulation requirements. If you modify the simulation drivers, do not change the simulation driver function prototypes. These class simulation drivers are updated to comply with the new revision of the IVI class API

specifications. Also included is a class simulation driver for the IviDcpwr class, which replaces the IviPower driver.

Updates to NI Spy

The IVI interface in NI-Spy has also been improved. When Spying on IVI function calls, NI-Spy now displays attribute names and error names instead of their hexadecimal equivalents. The help is now available on IVI attributes inside the Set/Get/Check attribute function call properties.



322382B-01

Jun00