



IVI-4.4: IviDCPwr Class Specification

April, 2002
Revision 2.0

Important Information

The IviDCPwr Class Specification (IVI-4.4) is authored by the IVI Foundation member companies. For a vendor membership roster list, please visit the IVI Foundation web site at www.ivifoundation.org, or contact the IVI Foundation at 2515 Camino del Rio South, Suite 340, San Diego, CA 92108.

The IVI Foundation wants to receive your comments on this specification. You can contact the Foundation through email at ivilistserver@ivifoundation.org, through the web site at www.ivifoundation.org, or you can write to the IVI Foundation, 2515 Camino del Rio South, Suite 340, San Diego, CA 92108.

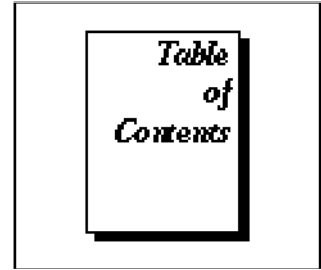
Warranty

The IVI Foundation and its member companies make no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The IVI Foundation and its member companies shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Trademarks

Product and company names listed are trademarks or trade names of their respective companies.

No investigation has been made of common-law trademark rights in any work.



1	Overview of the IviDCPwr Specification.....	7
1.1	Introduction.....	7
1.2	IviDCPwr Class Overview	7
1.3	References.....	7
1.4	Definitions of Terms and Acronyms	8
2	IviDCPwr Class Capabilities	9
2.1	Introduction.....	9
2.2	IviDCPwr Group Names	9
2.3	Repeated Capability Names	9
2.3.1	OutputChannel	9
3	General Requirements.....	10
3.1	Minimum Class Compliance.....	10
3.1.1	Disable.....	10
3.2	Capability Group Compliance.....	10
4	IviDCPwrBase Capability Group	11
4.1	Overview	11
4.2	IviDCPwrBase Attributes.....	13
4.2.1	Current Limit.....	14
4.2.2	Current Limit Behavior	15
4.2.3	Output Enabled	16
4.2.4	OVP Enabled.....	17
4.2.5	OVP Limit.....	18
4.2.6	Voltage Level.....	19
4.2.7	OutputChannel Count.....	20
4.2.8	OutputChannel Item (COM only)	21
4.2.9	OutputChannel Name (COM only).....	22
4.3	IviDCPwrBase Functions.....	23
4.3.1	Configure Current Limit.....	24
4.3.2	Configure Output Enabled (IVI-C only)	25
4.3.3	Configure Output Range	26
4.3.4	Configure OVP	28
4.3.5	Configure Voltage Level (IVI-C only)	29
4.3.6	Get OutputChannel Name (IVI-C only)	30
4.3.7	Query Current Limit Max.....	31
4.3.8	Query Voltage Level Max.....	32
4.3.9	Query Output State	33
4.3.10	Reset Output Protection	35

4.4	IviDCPwrBase Behavior Model	36
5	IviDCPwrTrigger Extension Group.....	37
5.1	IviDCPwrTrigger Overview.....	37
5.2	IviDCPwrTrigger Attributes	37
5.2.1	Trigger Source.....	38
5.2.2	Triggered Current Limit	41
5.2.3	Triggered Voltage Level.....	42
5.3	IviDCPwrTrigger Functions.....	43
5.3.1	Abort.....	44
5.3.2	Configure Triggered Current Limit (IVI-C only)	45
5.3.3	Configure Triggered Voltage Level (IVI-C only)	46
5.3.4	Configure Trigger Source (IVI-C only).....	47
5.3.5	Initiate.....	48
5.4	IviDCPwrTrigger Behavior Model.....	49
6	IviDCPwrSoftwareTrigger Extension Group	50
6.1	IviDCPwrSoftwareTrigger Overview.....	50
6.2	IviDCPwrTrigger Functions.....	50
6.2.1	Send Software Trigger.....	50
6.3	IviDCPwrSoftwareTrigger Behavior Model.....	50
6.4	IviDCPwrSoftwareTrigger Compliance Notes	50
7	IviDCPwrMeasurement Extension Group.....	51
7.1	IviDCPwrMeasurement Overview.....	51
7.2	IviDCPwrMeasurement Functions	51
7.2.1	Measure	52
7.3	IviDCPwrMeasurement Behavior Model.....	54
8	IVIDCPwr Attribute ID Definitions	55
9	IVIDCPwr Attribute Value Definitions	56
10	IviDCPwr Function Parameter Value Definitions.....	59
11	Error and Completion Code Value Definitions	61
12	IviDCPwr Hierarchies.....	62
12.1	IviDCPwr COM Hierarchy.....	62
12.1.1	IviDCPwr COM Interfaces.....	62
12.1.2	IviDCPwr COM Interface Reference Properties	63
12.1.3	IviDCPwr COM Category.....	63
12.2	IviDCPwr C Function Hierarchy	64
12.3	IviDCPwr C Attribute Hierarchy.....	65

Appendix A	Specific Driver Development Guidelines.....	66
A.1	Introduction.....	66
A.2	Disabling Unused Extensions	66
A.3	Special Considerations for Configure Output Range	67
A.4	Special Considerations for Initiate.....	67
A.5	Special Considerations for Current Limit and Current Limit Behavior.....	68
Appendix B.	Interchangeability Checking Rules	69
B.1	Introduction.....	69
B.2	When to Perform Interchangeability Checking	69
B.3	Interchangeability Checking Rules.....	69
Appendix C.	ANSI C Include File	71
Appendix D.	COM IDL File	76
D.1	IviDCPwrTypeLib.idl.....	76
D.2	IVIDCPwr.idl	76
D.3	IviDCPwrEnglish.idl	85

IviDCPwr Class Specification

IviDCPwr Revision History

This section is an overview of the revision history of the IviDCPwr specification.

Table 1. IviDCPwr Class Specification Revisions

Revision Number	Date of Revision	Revision Notes
Revision 1.0 draft	October 1, 1999	Original draft.
Revision 2.0a draft	April 31, 2001	First draft to include COM requirements
Revision 2.1vc1 draft	July 30, 2001	Voting candidate 1 review draft.
Revision 2.1vc2 draft	August 30, 2001	Voting candidate 2 review draft.
Revision 2.1vc3 draft	November, 2001	Voting candidate 3 review draft.
Revision 2.1vc5 draft	December, 2001	Voting candidate 5 review draft. Changes based on discussion at December IVI meeting.
Revision 2.0vc6 draft	January, 2002	Voting candidate 6 review draft. Changes based on comments from NI.
Revision 2.0vc7 draft	March, 2002	Added latest IDL.
Revision 2.0vc7 draft	March, 2002	IVI_CLASS_PUBLIC_ATTR_BASE changed to IVI_CLASS_ATTR_BASE.
Revision 2.0 Final	March, 2002	Approved Specification

1 Overview of the IviDCPwr Specification

1.1 Introduction

This specification defines the IVI class for DC power supplies. The IviDCPwr class is designed to support the typical DC power supply as well as common extended functionality found in more complex instruments. This section summarizes the *IviDCPwr Specification* itself and contains general information that the reader may need in order to understand, interpret, and implement aspects of this specification. These aspects include the following:

- ? IviDCPwr Class Overview
- ? The definitions of terms and acronyms
- ? References

1.2 IviDCPwr Class Overview

This specification describes the IVI class for DC power supplies. The IviDCPwr class is designed to support the typical DC Power supply as well as common extended functionality found in more complex instruments. The IviDCPwr class conceptualizes a DC power supply as an instrument capable of generating a DC power signal.

The IviDCPwrBase capability group provides the capability to configure a power supply for basic signal output. This includes setting the output range, the output voltage level, the over-voltage protection level, the current limit, enabling or disabling over-voltage protection, setting the current limit behavior, and enabling or disabling outputs. The IviDCPwrBase capability group is described in section 4, *IviDCPwrBase Capability Group*.

The IviDCPwrTrigger extension provides the capability to make changes to the output signal based on a trigger event. The driver can configure the trigger source, the triggered voltage level, and the triggered current limit. This extension group is described in section 5, *IviDCPwrTrigger Extension Group*.

The IviDCPwrSoftwareTrigger extension provides the capability to make changes to the output signal based on a software trigger event. Sending a software trigger to the instrument causes it to change the output signal. This extension group is described in section 6, *IviDCPwrSoftwareTrigger Extension Group*.

The IviDCPwrMeasurement extension provides the capability to take measurements on the output signal. With this extension, the driver can take immediate measurements of values like the DC Voltage and DC Current. This extension group is described in section 7, *IviDCPwrMeasurement Extension Group*.

1.3 References

Several other documents and specifications are related to this specification. These other related documents are the following:

- ? IVI Charter Document
- ? IVI-3.1 - Driver Architecture Specification
- ? IVI-3.2 - Inherent Capabilities Specification
- ? IVI-3.3 – Standard Cross Class Capabilities
- ? VPP-3.x - VXi*plug&play* Instrument Driver Specifications
- ? VPP-4.x - Virtual Instrument Software Architecture (VISA) Specifications

1.4 Definitions of Terms and Acronyms

Refer to *IVI-5: Glossary* for a description of the terms and acronyms used in this specification. This specification does not define any additional terms.

2 IviDCPwr Class Capabilities

2.1 Introduction

The IviDCPwr specification divides DC power supply capabilities into a base capability group and multiple extension capability groups. Each capability group is discussed in a separate section. This section defines names for each capability group and gives an overview of the information presented for each capability group.

2.2 IviDCPwr Group Names

The capability group names for the IviDCPwr class are defined in the following table. The group name is used to represent a particular capability group and is returned as one of the possible group names from the Group Capabilities attribute.

Table 2-1. IviDCPwr Group Names

Group Name	Description
IviDCPwrBase	Base capabilities of the IviDCPwr specification. This group supports the ability to generate a DC power signal, to specify output limits, and to control the behavior of the power supply when the output is greater than or equal to one of the limits.
IviDCPwrTrigger	This group supports the ability to make changes to the output signal based on a trigger event.
IviDCPwrSoftwareTrigger	This group supports the ability to make changes to the output signal based on a software trigger event.
IviDCPwrMeasurement	This group supports the ability to query the instrument for the measurement characteristics of the output signal.

Refer to Section 15, *Class Specification Layout*, in *IVI-3.4: API Style Guide* for a description of the Capability Group Section Layout.

2.3 Repeated Capability Names

The IviDCPwr Class Specification defines one repeated capability. Refer to the sections of *IVI-3.1, Driver Architecture Specification* that deal with repeated capabilities. The relevant sections are Section 2.7, *Repeated Capabilities*, Section 4.1.9, *Repeated Capabilities*, Section 4.2.5, *Repeated Capabilities*, and Section 5.8, *Repeated Capability Identifiers and Selectors*.

? OutputChannel

2.3.1 OutputChannel

In the configuration store, the name shall be “OutputChannel”.

3 General Requirements

This section describes the general requirements a specific driver shall meet in order to be compliant with this specification. In addition, it provides general requirements that specific drivers shall meet in order to comply with a capability group, attribute, or function.

3.1 *Minimum Class Compliance*

To be compliant with the IviDCPwr Class Specification, a specific driver shall implement the inherent capabilities that *IVI-3.2: Inherent IVI Capabilities Specification* defines, and the IviDCPwrBase capability group.

3.1.1 Disable

Refer to *IVI-3.2: Inherent Capabilities Specification* for the prototype of this function.

The Disable function shall cause the DC Power Supply to apply the minimum amount of power possible at the output terminals. Setting the voltage to a value close to zero, setting the current limit to value close to zero, or physically disconnecting the power supply from the output terminals meets this requirement. Other techniques are also allowed.

3.2 *Capability Group Compliance*

IVI-3.1: Driver Architecture Specification defines the general rules for a specific driver to be compliant with a capability group.

4 IviDCPwrBase Capability Group

4.1 Overview

The IviDCPwrBase capability group supports the most basic DC power supply capabilities. The user can enable or disable outputs, specify the DC voltage to generate, specify output limits, and control the behavior of the power supply when the output is greater than or equal to one of the limits.

This specification uses the following terms to describe the power supply's output: Voltage Level, OVP Limit, Current Limit, Current Limit Behavior, Constant Voltage Mode, Constant Current Mode, and Unregulated Mode.

Voltage Level – The DC voltage the power supply attempts to generate. The user configures the voltage level with the Voltage Level attribute.

OVP – OVP is an acronym for Over-Voltage Protection.

OVP Limit and OVP Enabled – If the OVP limit is enabled, the power supply disables the output when the output voltage is greater than or equal to the OVP limit. The user configures the OVP limit with the OVP Limit attribute, and enables or disables the OVP limit with the OVP Enabled attribute.

Current Limit and Current Limit Behavior – The current limit behavior determines the behavior of the instrument when the output current is greater than or equal to the current limit. When the current limit behavior is *trip*, the power supply disables the output when the output current is greater than or equal to the current limit. When the current limit behavior is *regulate*, the power supply restricts the output voltage such that the output current is not greater than the current limit. The user configures the current limit and current limit behavior with the Current Limit and Current Limit Behavior attributes.

Constant Voltage Mode – The power supply is said to be in the constant voltage mode when the power supply's output signal reaches the voltage level before it reaches the current limit. In the constant voltage mode, the power supply's output voltage remains constant at the voltage level and its output current can vary.

Constant Current Mode – The power supply is said to be in the constant current mode when the power supply's output signal reaches the current limit before it reaches the voltage level, and the current limit behavior is set to regulate. In the constant current mode, the power supply's output current remains constant at the current limit and its output voltage varies.

Unregulated Mode – The power supply is said to be in the unregulated mode when the power supply's output signal reaches neither the voltage level or the current limit. In the unregulated mode, the power supply's output current and output voltage vary.

The signal that the power supply produces depends on the values of the voltage level, OVP limit, and current limit that the user supplies, and the impedance of the load to which the power supply is attached. Therefore, the power supply might not produce the exact voltage or current that the user configures. The following diagram shows the possible output scenarios.

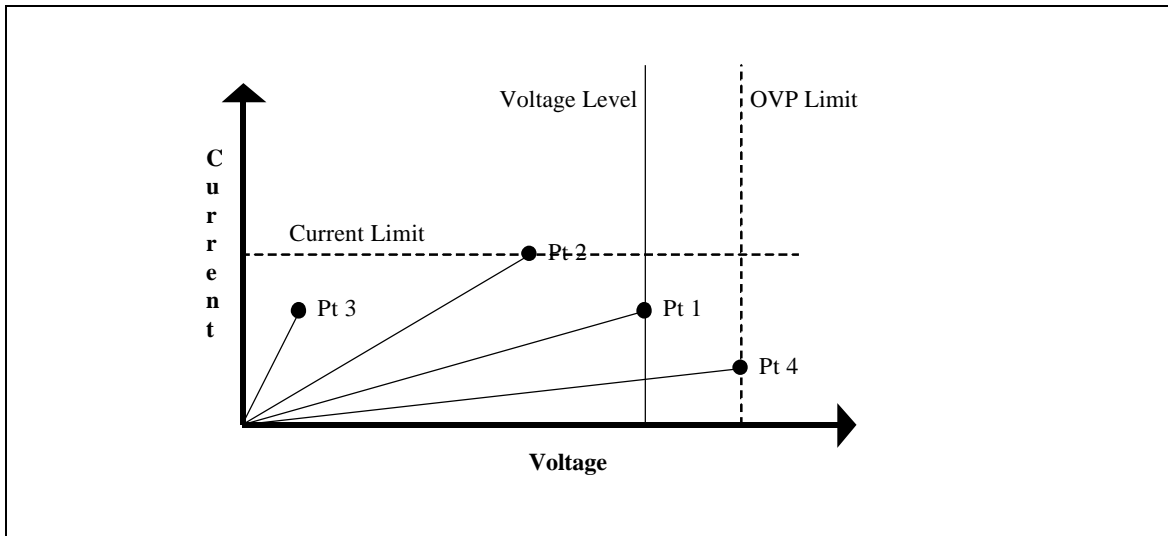


Figure 4-1. Power Supply Output Diagram

At Point 1, the power supply's output has reached the voltage level before it reached the current limit. This is an example of the power supply operating in the constant voltage mode. Note that any point on the vertical line defined by the voltage level would also cause the power supply to operate in the constant voltage mode.

At Point 2, the power supply's output has reached the current limit before it reached the voltage level. If the current limit behavior is set to regulate, this is an example of the power supply operating in the constant current mode. If the current limit behavior is set to trip, the power supply disables the output. Note that any point on the horizontal line defined by the current limit would also cause the power supply to operate in the constant current mode when the current limit behavior is set to regulate.

At point 3, the power supply's output has reached neither the voltage level or the current limit. This is an example of the power supply operating in the unregulated mode. Note that any point within the rectangle defined by the voltage level and current limit would also cause the power supply to operate in the unregulated mode.

At point 4, the power supply's output has reached the OVP limit. If OVP is enabled, the power supply disables the output. Note that any point on the vertical line defined by the OVP limit would also cause the power supply to disable the output when OVP is enabled.

4.2 *IviDCPwrBase* Attributes

The *IviDCPwrBase* capability group defines the following attributes:

- ? Current Limit
- ? Current Limit Behavior
- ? Output Enabled
- ? OVP Enabled
- ? OVP Limit
- ? Voltage Level
- ? OutputChannel Count
- ? OutputChannel Item (COM only)
- ? OutputChannel Name (COM only)

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in section 8, *IviDCPwr* Attribute ID Definitions.

4.2.1 Current Limit

Data Type	Access	Applies to	Coercion	High Level Functions
ViReal64	R/W	OutputChannel	None	Configure Current Limit

COM Property Name

`Outputs.Item().CurrentLimit`

COM Enumeration Name

N/A

C Constant Name

`IVIDCPWR_ATTR_CURRENT_LIMIT`

Description

Specifies the output current limit. The units are Amps.

The value of the Current Limit Behavior attribute determines the behavior of the power supply when the output current is equal to or greater than the value of this attribute.

4.2.2 Current Limit Behavior

Data Type	Access	Applies to	Coercion	High Level Functions
ViInt32	R/W	OutputChannel	None	Configure Current Limit

COM Property Name

`Outputs.Item().CurrentLimitBehavior`

COM Enumeration Name

`IviDcPwrCurrentLimitEnum`

C Constant Name

`IVIDCPWR_ATTR_CURRENT_LIMIT_BEHAVIOR`

Description

Specifies the behavior of the power supply when the output current is equal to or greater than the value of the Current Limit attribute.

Defined Values

<i>Name</i>	<i>Description</i>	
	<i>Language</i>	<i>Identifier</i>
Current Trip	The power supply disables the output when the output current is equal to or greater than the value of the Current Limit attribute.	
	C	IVIDCPWR_VAL_CURRENT_TRIP
	COM	IviDcPwrCurrentLimitTrip
Current Regulate	The power supply restricts the output voltage such that the output current is not greater than the value of the Current Limit attribute.	
	C	IVIDCPWR_VAL_CURRENT_REGULATE
	COM	IviDcPwrCurrentLimitRegulate

Compliance Notes

1. If an IVI-C specific driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDCPWR_VAL_CURRENT_LIMIT_BEHAVIOR_SPECIFIC_EXT_BASE.
2. If an IVI-C class driver defines additional values for this attribute, the actual values shall be greater than or equal to IVIDCPWR_VAL_CURRENT_LIMIT_BEHAVIOR_CLASS_EXT_BASE and less than IVIDCPWR_VAL_CURRENT_LIMIT_BEHAVIOR_SPECIFIC_EXT_BASE.
3. When an IVI-COM specific driver implements this attribute with additional elements in its instrument specific interfaces, the actual values of the additional elements shall be greater than or equal to Current Limit Behavior Specific Ext Base.

4.2.3 Output Enabled

Data Type	Access	Applies to	Coercion	High Level Functions
ViBoolean	R/W	OutputChannel	None	Configure Output Enabled

COM Property Name

`Outputs.Item().Enabled`

COM Enumeration Name

N/A

C Constant Name

`IVIDCPWR_ATTR_OUTPUT_ENABLED`

Description

Specifies whether the signal the power supply produces appears at the output connector.

Defined Values

<i>Name</i>	<i>Description</i>	
	<i>Language</i>	<i>Identifier</i>
True	The signal the power supply produces appears at the output connector.	
	C	VI_TRUE
	COM	True
False	The signal the power supply produces does not appear at the output connector.	
	C	VI_FALSE
	COM	False

Compliance Notes

Instrument drivers shall support the value True on all outputs.

4.2.4 OVP Enabled

Data Type	Access	Applies to	Coercion	High Level Functions
ViBoolean	R/W	OutputChannel	None	Configure OVP

COM Property Name

`Outputs.Item().OVPEnabled`

COM Enumeration Name

N/A

C Constant Name

`IVIDCPWR_ATTR_OVP_ENABLED`

Description

Specifies whether the power supply provides over-voltage protection. If this attribute is set to True, the power supply disables the output when the output voltage is greater than or equal to the value of the OVP Limit attribute.

Defined Values

Name	Description	
	Language	Identifier
True	The power supply disables the output when the output voltage is greater than or equal to the value of the OVP Limit attribute.	
	C	VI_TRUE
	COM	True
False	The power supply does not disable the output when the output voltage is greater than or equal to the value of the OVP Limit attribute.	
	C	VI_FALSE
	COM	False

4.2.5 OVP Limit

Data Type	Access	Applies to	Coercion	High Level Functions
ViReal64	R/W	OutputChannel	None	Configure OVP

COM Property Name

`Outputs.Item().OVPLimit`

COM Enumeration Name

N/A

C Constant Name

`IVIDCPWR_ATTR_OVP_LIMIT`

Description

Specifies the voltage the power supply allows. The units are Volts.

If the OVP Enabled attribute is set to True, the power supply disables the output when the output voltage is greater than or equal to the value of this attribute.

If the OVP Enabled is set to False, this attribute does not affect the behavior of the instrument.

4.2.6 Voltage Level

Data Type	Access	Applies to	Coercion	High Level Functions
ViReal64	R/W	OutputChannel	None	Configure Voltage Level

COM Property Name

`Outputs.Item().VoltageLevel`

COM Enumeration Name

N/A

C Constant Name

`IVIDCPWR_ATTR_VOLTAGE_LEVEL`

Description

Specifies the voltage level the DC power supply attempts to generate. The units are Volts.

4.2.7 OutputChannel Count

Data Type	Access	Applies to	Coercion	High Level Functions
viInt32	RO	N/A	None	None

COM Property Name

`Outputs.Count`

COM Enumeration Name

N/A

C Constant Name

`IVIDCPWR_ATTR_CHANNEL_COUNT`

Description

Returns the number of available outputs. The C Constant Name uses CHANNEL while the COM Property Name is tied to the Outputs collection.

4.2.8 OutputChannel Item (COM only)

Data Type	Access	Applies to	Coercion	High Level Functions
IIviDCPwrOutput*	RO	N/A	None	None

COM Property Name

`Outputs.Item([in] BSTR OutputName)`

COM Enumeration Name

N/A

C Constant Name

N/A

Description

The OutputChannel Item attribute requires a string parameter which is the name of one of the outputs. It returns an interface pointer which can be used to control the attributes and other functionality of that output.

Valid output names include those defined by the driver itself and names defined as aliases in the configuration store.

4.2.9 OutputChannel Name (COM only)

Data Type	Access	Applies to	Coercion	High Level Functions
ViString	RO	N/A	None	

COM Property Name

Outputs.Name ([in] LONG Index)

COM Enumeration Name

N/A

C Constant Name

N/A

Description

This attribute returns the physical name identifier defined by specific driver for the OutputChannel that corresponds to the one-based index that the user specifies. If the value that the user passes for the `Index` parameter is less than one or greater than the value of the OutputChannel Count, the attribute returns an empty string for the value and returns an error.

4.3 IviDCPwrBase Functions

The IviDCPwrBase capability group defines the following functions:

- ? Configure Current Limit
- ? Configure Output Enabled (IVI-C only)
- ? Configure Output Range
- ? Configure OVP
- ? Configure Voltage Level (IVI-C only)
- ? Get OutputChannel Name (IVI-C only)
- ? Query Current Limit Max
- ? Query Voltage Level Max
- ? Query Output State
- ? Reset Output Protection

This section describes the behavior and requirements of each function.

4.3.1 Configure Current Limit

Description

This function configures the current limit. It specifies the output current limit value and the behavior of the power supply when the output current is greater than or equal to that value.

COM Method Prototype

```
HRESULT Outputs.Item().ConfigureCurrentLimit(  
    [in] IviDCPwrCurrentLimitBehaviorEnum Behavior,  
    [in] DOUBLE Limit);
```

C Prototype

```
ViStatus IviDCPwr_ConfigureCurrentLimit (ViSession Vi,  
    ViConstString ChannelName,  
    ViInt32 Behavior,  
    ViReal64 Limit);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output on which to configure the current limit.	ViConstString
Behavior	Specifies the behavior of the power supply when the output current is greater than or equal to the value of the <code>Limit</code> parameter. The driver uses this value to set the Current Limit Behavior attribute. See the attribute description for more details.	ViInt32
Limit	Specifies the power supply's output current limit. The driver uses this value to set the Current Limit attribute. See the attribute description for more details.	ViReal64

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

4.3.2 Configure Output Enabled (IVI-C only)

Description

Configures whether the signal that the power supply produces appears at the output connector.

COM Method Prototype

N/A

(use the `Outputs.Item().Enabled` property)

C Prototype

```
ViStatus IviDCPwr_ConfigureOutputEnabled (ViSession Vi,  
                                           ViConstString ChannelName,  
                                           ViBoolean Enabled);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output to enable or disable.	ViConstString
Enabled	Specifies whether the signal the power supply produces appears at the output connector. The driver uses this value to set the Output Enabled attribute. See the attribute description for more details.	ViBoolean

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

4.3.3 Configure Output Range

Description

Configures the power supply's output range on an output. One parameter specifies whether to configure the voltage or current range, and the other parameter is the value to which to set the range.

Setting a voltage range can invalidate a previously configured current range. Setting a current range can invalidate a previously configured voltage range.

COM Method Prototype

```
HRESULT Outputs.Item().ConfigureRange([in] IviDCPwrRangeTypeEnum RangeType,
[in] DOUBLE Range);
```

C Prototype

```
ViStatus IviDCPwr_ConfigureOutputRange (ViSession Vi,
ViConstString ChannelName,
ViInt32 RangeType,
ViReal64 Range);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output on which to configure the output range.	ViConstString
RangeType	The kind of range to be configured.	ViInt32
Range	The range in which the power supply operates. The instrument driver coerces this value to the closest value the instrument supports that is greater than or equal to the value specified.	ViReal64

Defined Values for RangeType Parameter

Name	Description	
	Language	Identifier
Voltage	Voltage range is set by the Range parameter.	
	C	IVIDCPWR_VAL_RANGE_VOLTAGE
	COM	IviDCPwrRangeVoltage
Current	Current range is set by the Range parameter.	
	C	IVIDCPWR_VAL_RANGE_CURRENT
	COM	IviDCPwrRangeCurrent

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

Compliance Notes

1. If an IVI-C class driver defines additional values for the `RangeType` parameter, the actual values shall be greater than or equal to `IVIDCPWR_VAL_RANGE_TYPE_CLASS_EXT_BASE` and less than `IVIDCPWR_VAL_RANGE_TYPE_SPECIFIC_EXT_BASE`.
2. If an IVI-C specific driver defines additional values for the `RangeType` parameter, the actual values shall be greater than or equal to `IVIDCPWR_VAL_RANGE_TYPE_SPECIFIC_EXT_BASE`.

4.3.4 Configure OVP

Description

Configures the over-voltage protection. It specifies the over-voltage limit and the behavior of the power supply when the output voltage is greater than or equal to that value.

When the `Enabled` parameter is `False`, the `Limit` parameter does not affect the instrument's behavior, and the driver does not set the `OVP Limit` attribute.

COM Method Prototype

```
HRESULT Outputs.Item().ConfigureOVP([in] VARIANT_BOOL Enabled,  
                                     [in] DOUBLE Limit);
```

C Prototype

```
ViStatus IviDCPwr_ConfigureOVP (ViSession Vi,  
                                ViConstString ChannelName,  
                                ViBoolean Enabled,  
                                ViReal64 Limit);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output on which to configure over-voltage protection.	ViConstString
Enabled	Specifies the behavior of the power supply when the output voltage is greater than or equal to the value of the <code>Limit</code> parameter. The driver uses this value to set the <code>OVP Enabled</code> attribute. See the attribute description for more details.	ViBoolean
Limit	Specifies the power supply's over-voltage protection limit. The driver uses this value to set the <code>OVP Limit</code> attribute. See the attribute description for more details.	ViReal64

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

4.3.5 Configure Voltage Level (IVI-C only)

Description

Configures the voltage level the power supply attempts to generate.

COM Method Prototype

N/A

(use the `Outputs.Item().VoltageLevel` property)

C Prototype

```
ViStatus IviDCPwr_ConfigureVoltageLevel (ViSession Vi,  
                                         ViConstString ChannelName,  
                                         ViReal64 Level);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output on which to configure the voltage level.	ViConstString
Level	Specifies the voltage level the power supply attempts to generate. The driver uses this value to set the Voltage Level attribute. See the attribute description for more details.	ViReal64

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

4.3.6 Get OutputChannel Name (IVI-C only)

Description

This function returns the specific driver defined output name that corresponds to the one-based index that the user specifies. If the value that the user passes for the `Index` parameter is less than one or greater than the value of the `OutputChannel Count` attribute, the function returns an empty string in the `Name` parameter and returns an error.

The C prototype uses `Channel` in the function name rather than the name of the repeated capability.

COM Method Prototype

N/A

(use the `Outputs.Name` property)

C Prototype

```
ViStatus IviDCPwr_GetChannelName (ViSession Vi,  
                                  ViInt32 Index,  
                                  ViInt32 NameBufferSize,  
                                  ViChar Name[]);
```

Parameters

Inputs	Description	Base Type
<code>Vi</code>	Instrument handle	<code>ViSession</code>
<code>Index</code>	A one-based index that defines which name to return.	<code>ViInt32</code>
<code>NameBufferSize</code>	The number of bytes in the <code>ViChar</code> array that the user specifies for the <code>Name</code> parameter.	<code>ViInt32</code>

Outputs	Description	Base Type
<code>Name</code>	A user-allocated (for IVI-C) or driver-allocated (for IVI-COM) buffer into which the driver stores the output name. The caller may pass <code>VI_NULL</code> for this parameter if the <code>NameBufferSize</code> parameter is 0.	<code>ViChar[]</code>

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

4.3.7 Query Current Limit Max

Description

This function returns the maximum programmable current limit that the power supply accepts for a particular voltage level on an output.

COM Method Prototype

```
HRESULT Outputs.Item().QueryCurrentLimitMax(  
    [in] DOUBLE VoltageLevel,  
    [out, retval] DOUBLE *MaxCurrentLimit);
```

C Prototype

```
ViStatus IviDCPwr_QueryMaxCurrentLimit (ViSession Vi,  
    ViConstString ChannelName,  
    ViReal64 VoltageLevel,  
    ViReal64* MaxCurrentLimit);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output on which to query the maximum current limit.	ViConstString
VoltageLevel	The voltage level for which to determine the maximum programmable current limit.	ViReal64

Outputs	Description	Base Type
MaxCurrentLimit	Returns the maximum programmable current limit for the specified voltage level.	ViReal64*

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

4.3.8 Query Voltage Level Max

Description

This function returns the maximum programmable voltage level that the power supply accepts for a particular current limit on an output.

COM Method Prototype

```
HRESULT Outputs.Item().QueryVoltageLevelMax(  
    [in] DOUBLE CurrentLimit,  
    [out, retval] DOUBLE *MaxVoltageLevel);
```

C Prototype

```
ViStatus IviDCPwr_QueryMaxVoltageLevel (ViSession Vi,  
    ViConstString ChannelName,  
    ViReal64 CurrentLimit,  
    ViReal64* MaxVoltageLevel);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output on which to query the maximum voltage level.	ViConstString
CurrentLimit	The current limit for which to determine the maximum programmable voltage level.	ViReal64

Outputs	Description	Base Type
MaxVoltageLevel	Returns the maximum programmable voltage level for the specified current limit.	ViReal64*

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

4.3.9 Query Output State

Description

This function returns whether the power supply is in a particular output state.

A constant voltage condition occurs when the output voltage is equal to the value of the Voltage Level attribute and the current is less than or equal to the value of the Current Limit attribute.

A constant current condition occurs when the output current is equal to the value of the Current Limit attribute and the Current Limit Behavior attribute is set to the Current Regulate defined value.

An unregulated condition occurs when the output voltage is less than the value of the Voltage Level attribute and the current is less than the value of the Current Limit attribute.

An over-voltage condition occurs when the output voltage is equal to or greater than the value of the OVP Limit attribute and the OVP Enabled attribute is set to True.

An over-current condition occurs when the output current is equal to or greater than the value of the Current Limit attribute and the Current Limit Behavior attribute is set to the Current Trip defined value.

When either an over-voltage condition or an over-current condition occurs, the power supply's output protection disables the output. If the power supply is in an over-voltage or over-current state, it does not produce power until the output protection is reset. The Reset Output Protection function resets the output protection. Once the output protection is reset, the power supply resumes generating a power signal.

COM Method Prototype

```
HRESULT Outputs.Item().QueryState([in] IviDCPwrOutputStateEnum OutputState,  
[out, retval] VARIANT_BOOL *InState);
```

C Prototype

```
ViStatus IviDCPwr_QueryOutputState (ViSession Vi,  
ViConstString ChannelName,  
ViInt32 OutputState,  
ViBoolean* InState);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The output on which to query the output state.	ViConstString
OutputState	The output state for which to query.	ViInt32

Outputs	Description	Base Type
InState	Returns True if the power supply is currently in the state specified with the OutputState parameter, and False if it is not.	ViBoolean

Defined Values for OutputState Parameter

<i>Name</i>	<i>Description</i>	
	<i>Language</i>	<i>Identifier</i>
Constant Voltage	Return value indicates whether a constant voltage condition exists.	
	C	IVIDCPWR_VAL_OUTPUT_CONSTANT_VOLTAGE
	COM	IviDCPwrOutputConstantVoltage
Constant Current	Return value indicates whether a constant current condition exists.	
	C	IVIDCPWR_VAL_OUTPUT_CONSTANT_CURRENT
	COM	IviDCPwrOutputConstantCurrent
Over Voltage	Return value indicates whether an over-voltage condition exists.	
	C	IVIDCPWR_VAL_OUTPUT_OVER_VOLTAGE
	COM	IviDCPwrOutputOverVoltage
Over Current	Return value indicates whether an over-current condition exists.	
	C	IVIDCPWR_VAL_OUTPUT_OVER_CURRENT
	COM	IviDCPwrOutputOverCurrent
Unregulated	Return value indicates whether an unregulated condition exists.	
	C	IVIDCPWR_VAL_OUTPUT_UNREGULATED
	COM	IviDCPwrOutputUnregulated

Return Values

The IVI-3.2: *Inherent Capabilities Specification* defines general status codes that this function can return.

Compliance Notes

1. If an IVI-C class driver defines additional values for the OutputState parameter, the actual values shall be greater than or equal to IVIDCPWR_VAL_OUTPUT_STATE_CLASS_EXT_BASE and less than IVIDCPWR_VAL_OUTPUT_STATE_SPECIFIC_EXT_BASE.
2. If an IVI-C specific driver defines additional values for the OutputState parameter, the actual values shall be greater than or equal to IVIDCPWR_VAL_OUTPUT_STATE_SPECIFIC_EXT_BASE.

4.3.10 Reset Output Protection

Description

This function resets the power supply output protection after an over-voltage or over-current condition occurs.

An over-voltage condition occurs when the output voltage is equal to or greater than the value of the OVP Limit attribute and the OVP Enabled attribute is set to True.

An over-current condition occurs when the output current is equal to or greater than the value of the Current Limit attribute and the Current Limit Behavior attribute is set to Current Trip.

When either an over-voltage condition or an over-current condition occurs, the output protection of the power supply disables the output. Once the output protection is reset, the power supply resumes generating a power signal.

Use the Query Output State function to determine if the power supply is in an over-voltage or over-current state.

COM Method Prototype

```
HRESULT Outputs.Item().ResetOutputProtection();
```

C Prototype

```
ViStatus IviDCPwr_ResetOutputProtection (ViSession Vi,  
                                         ViConstString ChannelName);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The output on which to reset output protection	ViConstString

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

4.4 IviDCPwrBase Behavior Model

After the user calls the Initialize or Reset functions, the power supply produces a power signal based on its current configuration.

All changes to the power supply's IviDCPwrBase functions and attributes take place immediately.

5 IviDCPwrTrigger Extension Group

5.1 *IviDCPwrTrigger Overview*

The IviDCPwrTrigger extension group defines extensions for DC power supplies capable of changing the output signal based on a trigger event.

5.2 *IviDCPwrTrigger Attributes*

The IviDCPwrTrigger capability group defines the following attributes:

- ? Trigger Source
- ? Triggered Current Limit
- ? Triggered Voltage Level

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in section 8, *IviDCPwr Attribute ID Definitions*.

5.2.1 Trigger Source

Data Type	Access	Applies to	Coercion	High Level Functions
ViInt32	R/W	OutputChannel	None	Configure Trigger Source

COM Property Name

```
Outputs.Item().TriggerSource
```

COM Enumeration Name

```
IviDCPwrTriggerSourceEnum
```

C Constant Name

```
IVIDCPWR_ATTR_TRIGGER_SOURCE
```

Description

Specifies the trigger source. After an Initiate call, the power supply waits for a trigger event from the source specified with this attribute. After a trigger event occurs, the power supply changes the voltage level to the value of the Triggered Voltage Level attribute and the current limit to the value of the Triggered Current Limit attribute.

Defined Values

Name	Description	
	Language	Identifier
Immediate	The power supply does not wait for a trigger before changing the output signal.	
	C	IVIDCPWR_VAL_TRIG_IMMEDIATE
	COM	IviDCPwrTriggerSourceImmediate
External	The power supply waits for an external trigger before changing the output signal.	
	C	IVIDCPWR_VAL_TRIG_EXTERNAL
	COM	IviDCPwrTriggerSourceExternal
Software Trigger	The power supply waits for the Send Software Trigger function to execute before changing the output signal.	
	C	IVIDCPWR_VAL_SOFTWARE_TRIG
	COM	IviDCPwrTriggerSourceSwTrigFunc
TTL0	The power supply waits for a trigger on the TTL 0 line before changing the output signal.	
	C	IVIDCPWR_VAL_TRIG_TTL0
	COM	IviDCPwrTriggerSourceTTL0
TTL1	The power supply waits for a trigger on the TTL 1 line before changing the output signal.	
	C	IVIDCPWR_VAL_TRIG_TTL1
	COM	IviDCPwrTriggerSourceTTL1

TTL2	The power supply waits for a trigger on the TTL 2 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_TTL2
	COM	IviDCPwrTriggerSourceTTL2
TTL3	The power supply waits for a trigger on the TTL 3 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_TTL3
	COM	IviDCPwrTriggerSourceTTL3
TTL4	The power supply waits for a trigger on the TTL 4 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_TTL4
	COM	IviDCPwrTriggerSourceTTL4
TTL5	The power supply waits for a trigger on the TTL 5 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_TTL5
	COM	IviDCPwrTriggerSourceTTL5
TTL6	The power supply waits for a trigger on the TTL 6 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_TTL6
	COM	IviDCPwrTriggerSourceTTL6
TTL7	The power supply waits for a trigger on the TTL 7 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_TTL7
	COM	IviDCPwrTriggerSourceTTL7
ECL0	The power supply waits for a trigger on the ECL 0 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_ECL0
	COM	IviDCPwrTriggerSourceECL0
ECL1	The power supply waits for a trigger on the ECL 1 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_ECL1
	COM	IviDCPwrTriggerSourceECL1
PXI Star	The power supply waits for a trigger on the PXI Star line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_PXI_STAR
	COM	IviDCPwrTriggerSourcePXIStar
RTSI0	The power supply waits for a trigger on the RTSI 0 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_RTISI_0
	COM	IviDCPwrTriggerSourceRTSI0
RTSI1	The power supply waits for a trigger on the RTSI 1 line before changing the output signal.	
	C	IviDCPwr_VAL_TRIG_RTISI_1
	COM	IviDCPwrTriggerSourceRTSI1
RTSI2	The power supply waits for a trigger on the RTSI 2 line before changing the output signal.	

		C	IvIDCPWR_VAL_TRIG_RTISI_2
		COM	IviDCPwrTriggerSourceRTISI2
RTSI3	The power supply waits for a trigger on the RTSI 3 line before changing the output signal.		
		C	IvIDCPWR_VAL_TRIG_RTISI_3
		COM	IviDCPwrTriggerSourceRTISI3
RTSI4	The power supply waits for a trigger on the RTSI 4 line before changing the output signal.		
		C	IvIDCPWR_VAL_TRIG_RTISI_4
		COM	IviDCPwrTriggerSourceRTISI4
RTSI5	The power supply waits for a trigger on the RTSI 5 line before changing the output signal.		
		C	IvIDCPWR_VAL_TRIG_RTISI_5
		COM	IviDCPwrTriggerSourceRTISI5
RTSI6	The power supply waits for a trigger on the RTSI 6 line before changing the output signal.		
		C	IvIDCPWR_VAL_TRIG_RTISI_6
		COM	IviDCPwrTriggerSourceRTISI6

Compliance Notes

1. If an IVI-C class driver defines additional values for this attribute, the actual values shall be greater than or equal to `IvIDCPWR_VAL_TRIG_SRC_CLASS_EXT_BASE` and less than `IvIDCPWR_VAL_TRIG_SRC_SPECIFIC_EXT_BASE`.
2. If an IVI-C specific driver defines additional values for this attribute, the actual values must be greater than or equal to `IvIDCPWR_VAL_TRIG_SRC_SPECIFIC_EXT_BASE`.
3. If a specific driver implements any of the defined values in the following table, it must also implement the corresponding capability group:

Value	Required Capability Group
Software Trigger	IviDCPwrSoftwareTrigger

4. When an IVI-COM specific driver implements this attribute with additional elements in its instrument specific interfaces, the actual values of the additional elements shall be greater than or equal to `Trigger Source Specific Ext Base`.

5.2.2 Triggered Current Limit

Data Type	Access	Applies to	Coercion	High Level Functions
ViReal64	R/W	OutputChannel	None	Configure Triggered Current Limit

COM Property Name

`Outputs.Item().TriggeredCurrentLimit`

COM Enumeration Name

N/A

C Constant Name

`IVIDCPWR_ATTR_TRIGGERED_CURRENT_LIMIT`

Description

Specifies the value to which the power supply sets the current limit after a trigger event occurs. The units are Amps.

After an Initiate call, the power supply waits for a trigger event from the source specified with the Trigger Source attribute. After a trigger event occurs, the power supply sets the current limit to the value of this attribute.

After a trigger occurs, the value of the Current Limit attribute reflects the new value to which the current limit has been set.

5.2.3 Triggered Voltage Level

Data Type	Access	Applies to	Coercion	High Level Functions
ViReal64	R/W	OutputChannel	None	Configure Triggered Voltage Level

COM Property Name

`Outputs.Item().TriggeredVoltageLevel`

COM Enumeration Name

N/A

C Constant Name

`IVIDCPWR_ATTR_TRIGGERED_VOLTAGE_LEVEL`

Description

Specifies the value to which the power supply sets the voltage level after a trigger event occurs. The units are Volts.

After an `Initiate` call, the power supply waits for a trigger event from the source specified with the `Trigger Source` attribute. After a trigger event occurs, the power supply sets the voltage level to the value of this attribute.

After a trigger occurs, the value of the `Voltage Level` attribute reflects the new value to which the voltage level has been set.

5.3 IviDCPwrTrigger Functions

The IviDCPwrTrigger extension defines the following functions:

- ? Abort
- ? Configure Triggered Current Limit (IVI-C only)
- ? Configure Triggered Voltage Level (IVI-C only)
- ? Configure Trigger Source (IVI-C only)
- ? Initiate

This section describes the behavior and requirements of the function.

5.3.1 Abort

Description

If the power supply is currently waiting for a trigger to change the output signal, this function returns the power supply to the ignore triggers state.

If the power supply is not waiting for a trigger, this function does nothing and returns Success.

COM Method Prototype

```
HRESULT Trigger.Abort();
```

C Prototype

```
ViStatus IviDCPwr_Abort (ViSession Vi);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

5.3.2 Configure Triggered Current Limit (IVI-C only)

Description

Configures the current limit the power supply attempts to generate after it receives a trigger.

COM Method Prototype

N/A

(use the `Outputs.Item().TriggeredCurrentLimit` property)

C Prototype

```
ViStatus IviDCPwr_ConfigureTriggeredCurrentLimit (ViSession Vi,  
                                                ViConstString ChannelName,  
                                                ViReal64 Limit);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output on which to configure the triggered current limit.	ViConstString
Limit	Specifies the current limit the power supply attempts to generate after it receives a trigger. The driver uses this value to set the Triggered Current Limit attribute. See the attribute description for more details.	ViReal64

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

5.3.3 Configure Triggered Voltage Level (IVI-C only)

Description

Configures the voltage level the power supply attempts to generate after it receives a trigger.

COM Method Prototype

N/A

(use the `Outputs.Item().TriggeredVoltageLevel` property)

C Prototype

```
ViStatus IviDCPwr_ConfigureTriggeredVoltageLevel (ViSession Vi,  
                                                ViConstString ChannelName,  
                                                ViReal64 Level);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output on which to configure the triggered voltage level.	ViConstString
Level	Specifies the voltage level the power supply attempts to generate after it receives a trigger. The driver uses this value to set the Triggered Voltage Level attribute. See the attribute description for more details.	ViReal64

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

5.3.4 Configure Trigger Source (IVI-C only)

Description

Configures the trigger source the power supply waits for before changing the output signal.

COM Method Prototype

N/A

(use the `Trigger.Source` property)

C Prototype

```
ViStatus IviDCPwr_ConfigureTriggerSource (ViSession Vi,  
                                          ViConstString ChannelName,  
                                          ViInt32 Source);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The name of the output on which to configure the trigger source.	ViConstString
Source	Specifies the trigger source the power supply waits for before changing the output signal. The driver uses this value to set the Trigger Source attribute. See the attribute description for more details.	ViInt32

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

5.3.5 Initiate

Description

If the power supply is not currently waiting for a trigger, this function causes the power supply to wait for a trigger.

If the power supply is already waiting for a trigger, this function does nothing and returns Success.

COM Method Prototype

```
HRESULT Trigger.Initiate();
```

C Prototype

```
ViStatus IviDCPwr_Initiate (ViSession Vi);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

5.4 IviDCPwrTrigger Behavior Model

The following behavior model shows the relationship between the IviDCPwrTrigger capability group and power supply behavior.

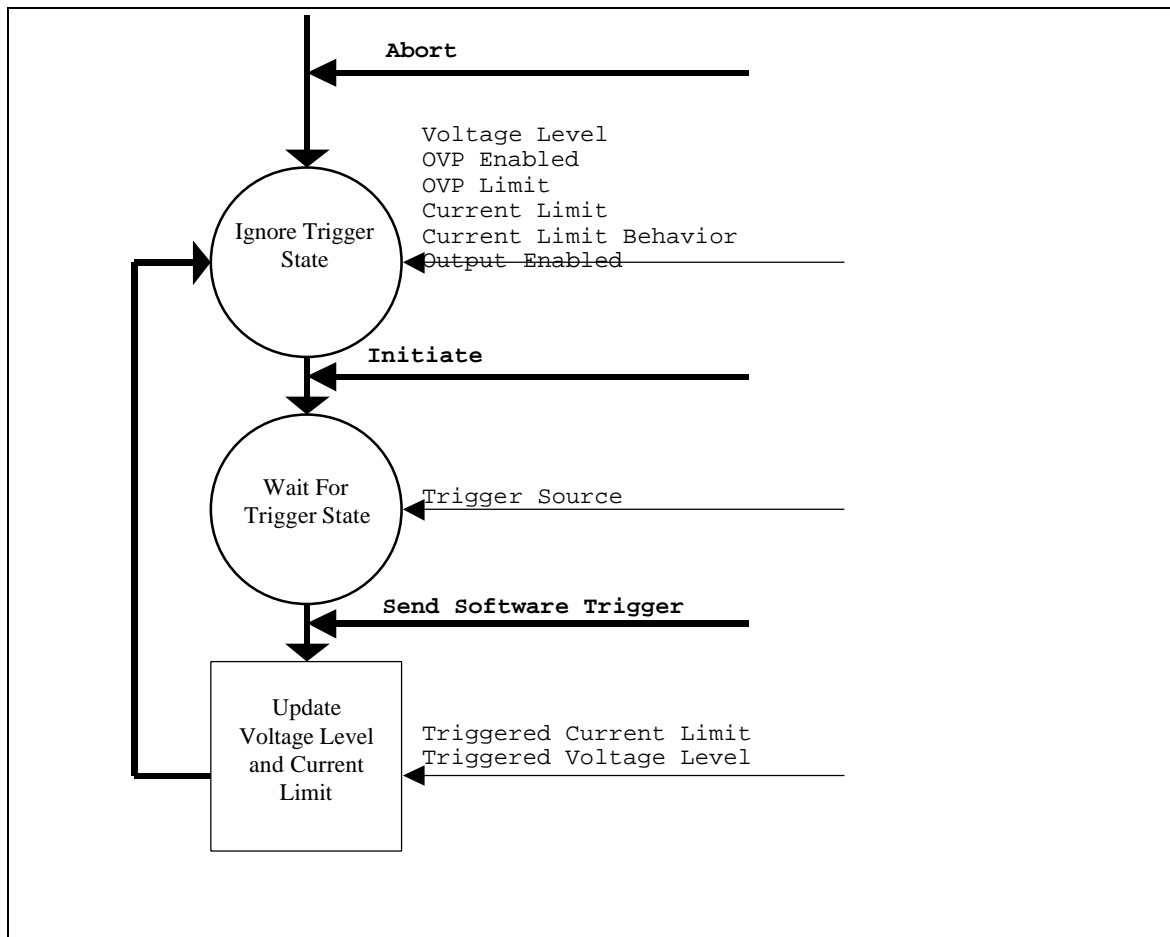


Figure 5-2. IviDCPwrTrigger Behavior Model

After the user calls the Initialize or Reset functions, the power supply enters the *ignore trigger* state.

In the *ignore trigger* state, the attributes of the IviDCPwrBase capability group determine the signal that the power supply produces. IviDCPwrTrigger attributes can be set, but do not affect the output signal.

Calling Initiate moves the power supply to the *wait for trigger* state.

In the *wait for trigger* state, the power supply waits for a trigger on the trigger source specified by the Trigger Source attribute. After the power supply receives a trigger, it sets the voltage level to the value of the Triggered Voltage Level attribute, and the current limit to the value of the Triggered Current Limit attribute. It then returns to the *ignore trigger* state.

After the changes in output occur, the Voltage Level and Current Limit attributes reflect the power supply's new configuration..

Calling the Abort function moves the power supply from its current state to the *ignore trigger* state. If the power supply has not yet responded to a trigger, no change occurs to the voltage level or current limit.

6 IviDCPwrSoftwareTrigger Extension Group

6.1 *IviDCPwrSoftwareTrigger Overview*

The IviDCPwrSoftwareTrigger extension group defines extensions for DC power supplies capable of changing the output signal based on a software trigger event.

6.2 *IviDCPwrTrigger Functions*

The IviDCPwrTrigger extension defines the following functions:

? Send Software Trigger

This section describes the behavior and requirements of this function.

6.2.1 Send Software Trigger

Refer to *IVI-3.3: Standard Cross Class Capabilities Specification, Section 2 Software Triggering Capability* for the prototype and complete description of this function.

6.3 *IviDCPwrSoftwareTrigger Behavior Model*

The IviDCPwrSoftwareTrigger Extension Group follows the behavior model of the IviDCPwrTrigger capability group. The only modification to the behavior model from the IviDCPwrTrigger capability group is the ability to send software triggers.

6.4 *IviDCPwrSoftwareTrigger Compliance Notes*

1. If an instrument driver implements the IviDCPwrSoftwareTrigger Extension Group, it shall implement the IviDCPwrTrigger Extension Group.
2. If an instrument driver implements the IviDCPwrSoftwareTrigger Extension Group, it shall implement the Software Trigger defined value for the Trigger Source attribute.

7 IviDCPwrMeasurement Extension Group

7.1 *IviDCPwrMeasurement Overview*

The IviDCPwrMeasurement extension group defines extensions for DC power supplies capable of calculating various measurements such as voltage and current from the output signal.

7.2 *IviDCPwrMeasurement Functions*

The IviDCPwrMeasurement extension defines the following function:

? Measure

This section describes the behavior and requirements of the function.

7.2.1 Measure

Description

Takes a measurement on the output signal and returns the measured value.

COM Method Prototype

```
HRESULT Outputs.Item().Measure(
    [in] IviDCPwrMeasurementTypeEnum MeasurementType,
    [out, retval] DOUBLE *Measurement);
```

C Prototype

```
ViStatus IviDCPwr_Measure (ViSession Vi,
    ViConstString ChannelName,
    ViInt32 MeasurementType,
    ViReal64* Measurement);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelName	The output on which to take a measurement	ViConstString
MeasurementType	The type of measurement to take.	ViInt32

Outputs	Description	Base Type
Measurement	The measured value.	ViReal64

Defined Values for MeasurementType Parameter

Name	Description	
	Language	Identifier
Voltage	Voltage is measured.	
	C	IVIDCPWR_VAL_MEASURE_VOLTAGE
	COM	IviDCPwrMeasurementVoltage
Current	Current is measured.	
	C	IVIDCPWR_VAL_MEASURE_CURRENT
	COM	IviDCPwrMeasurementCurrent

Return Values

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return.

Compliance Notes

1. If an IVI-C class driver defines additional values for the `measurementType` parameter, the actual values shall be greater than or equal to `IVIDCPWR_VAL_MEASURE_CLASS_EXT_BASE` and less than `IVIDCPWR_VAL_MEASURE_SPECIFIC_EXT_BASE`.

2. If an IVI-C specific driver defines additional values for the `rangeType` parameter, the actual values shall be greater than or equal to `IVIDCPWR_VAL_MEASURE_SPECIFIC_EXT_BASE`.

7.3 *IviDCPwrMeasurement Behavior Model*

The IviDCPwrMeasurement Extension Group follows the behavior model of the IviDCPwrBase capability group. The only modification to the behavior model from the IviDCPwrBase capability group is the ability to take measurement on the output signal.

8 IVIDCPwr Attribute ID Definitions

The following table defines the ID value for all IviDCPwr class attributes.

Table 8-1. IviDCPwr Attributes ID Values

Attribute Name	ID Definition
IVIDCPWR_ATTR_VOLTAGE_LEVEL	IVI_CLASS_ATTR_BASE + 1
IVIDCPWR_ATTR_OVP_ENABLED	IVI_CLASS_ATTR_BASE + 2
IVIDCPWR_ATTR_OVP_LIMIT	IVI_CLASS_ATTR_BASE + 3
IVIDCPWR_ATTR_CURRENT_LIMIT_BEHAVIOR	IVI_CLASS_ATTR_BASE + 4
IVIDCPWR_ATTR_CURRENT_LIMIT	IVI_CLASS_ATTR_BASE + 5
IVIDCPWR_ATTR_OUTPUT_ENABLED	IVI_CLASS_ATTR_BASE + 6
IVIDCPWR_ATTR_CHANNEL_COUNT	IVI_CLASS_ATTR_BASE + 203
IVIDCPWR_ATTR_TRIGGER_SOURCE	IVI_CLASS_ATTR_BASE + 101
IVIDCPWR_ATTR_TRIGGERED_CURRENT_LIMIT	IVI_CLASS_ATTR_BASE + 102
IVIDCPWR_ATTR_TRIGGERED_VOLTAGE_LEVEL	IVI_CLASS_ATTR_BASE + 103

In revision 1.0 of IVI 7: IviDCPwr Class Specification, the base attribute ID value was listed as: `IVI_CLASS_PUBLIC_ATTR_BASE`. The constant ID definition was renamed to `IVI_CLASS_ATTR_BASE`. The value of `IVI_CLASS_ATTR_BASE` is defined in IVI-3.1: Driver Architecture Specification.

9 IVIDCPwr Attribute Value Definitions

This section specifies the actual value for each defined attribute value.

Current Limit Behavior

<i>Value Name</i>	<i>Language</i>	<i>Identifier</i>	<i>Actual Value</i>
Current Regulate	C	IVIDCPWR_VAL_CURRENT_REGULATE	0
	COM	IviDCPwrCurrentLimitRegulate	0
Current Trip	C	IVIDCPWR_VAL_CURRENT_TRIP	1
	COM	IviDCPwrCurrentLimitTrip	1
Current Limit Behavior Class Ext Base	C	IVIDCPWR_VAL_CURRENT_LIMIT_BEHAVIOR_CLASS_EXT_BASE	500
	COM		
Current Limit Behavior Specific Ext Base	C	IVIDCPWR_VAL_CURRENT_LIMIT_BEHAVIOR_SPECIFIC_EXT_BASE	1000
	COM		1000

Trigger Source

<i>Value Name</i>	<i>Language</i>	<i>Identifier</i>	<i>Actual Value</i>
Immediate	C	IVIDCPWR_VAL_TRIG_IMMEDIATE	0
	COM	IviDCPwrTriggerSourceImmediate	0
External	C	IVIDCPWR_VAL_TRIG_EXTERNAL	1
	COM	IviDCPwrTriggerSourceExternal	1
Software Trigger	C	IVIDCPWR_VAL_SOFTWARE_TRIG	2
	COM	IviDCPwrTriggerSourceSwTrigFunc	2
TTL0	C	IVIDCPWR_VAL_TRIG_TTL0	3
	COM	IviDCPwrTriggerSourceTTL0	3
TTL1	C	IVIDCPWR_VAL_TRIG_TTL1	4
	COM	IviDCPwrTriggerSourceTTL1	4
TTL2	C	IVIDCPWR_VAL_TRIG_TTL2	5
	COM	IviDCPwrTriggerSourceTTL2	5
TTL3	C	IVIDCPWR_VAL_TRIG_TTL3	6
	COM	IviDCPwrTriggerSourceTTL3	6
TTL4	C	IVIDCPWR_VAL_TRIG_TTL4	7
	COM	IviDCPwrTriggerSourceTTL4	7
TTL5	C	IVIDCPWR_VAL_TRIG_TTL5	8
	COM	IviDCPwrTriggerSourceTTL5	8
TTL6	C	IVIDCPWR_VAL_TRIG_TTL6	9
	COM	IviDCPwrTriggerSourceTTL6	9
TTL7	C	IVIDCPWR_VAL_TRIG_TTL7	10
	COM	IviDCPwrTriggerSourceTTL7	10
ECL0	C	IVIDCPWR_VAL_TRIG_ECL0	11
	COM	IviDCPwrTriggerSourceECL0	11
ECL1	C	IVIDCPWR_VAL_TRIG_ECL1	12
	COM	IviDCPwrTriggerSourceECL1	12
PXI Star	C	IVIDCPWR_VAL_TRIG_PXI_STAR	13
	COM	IviDCPwrTriggerSourcePXIStar	13
RTSI0	C	IVIDCPWR_VAL_TRIG_RTSI_0	14
	COM	IviDCPwrTriggerSourceRTSI0	14
RTSI1	C	IVIDCPWR_VAL_TRIG_RTSI_1	15
	COM	IviDCPwrTriggerSourceRTSI1	15
RTSI2	C	IVIDCPWR_VAL_TRIG_RTSI_2	16
	COM	IviDCPwrTriggerSourceRTSI2	16
RTSI3	C	IVIDCPWR_VAL_TRIG_RTSI_3	17
	COM	IviDCPwrTriggerSourceRTSI3	17
RTSI4	C	IVIDCPWR_VAL_TRIG_RTSI_4	18
	COM	IviDCPwrTriggerSourceRTSI4	18
RTSI5	C	IVIDCPWR_VAL_TRIG_RTSI_5	19
	COM	IviDCPwrTriggerSourceRTSI5	19

RTSI6	C	IVIDCPWR_VAL_TRIG_RTSI_6	20
	COM	IviDCPwrTriggerSourceRTSI6	20
Trigger Source Class Ext Base	C	IVIDCPWR_VAL_TRIG_SRC_CLASS_EXT_BASE	500
	COM		
Trigger Source Specific Ext Base	C	IVIDCPWR_VAL_TRIG_SRC_SPECIFIC_EXT_BASE	1000
	COM		1000

10 IviDCPwr Function Parameter Value Definitions

This section specifies the actual values for each function parameter that defines values.

Configure Output Range

Parameter: RangeType

<i>Value Name</i>	<i>Language</i>	<i>Identifier</i>	<i>Actual Value</i>
Current	C	IVIDCPWR_VAL_RANGE_CURRENT	0
	COM	IviDCPwrRangeCurrent	0
Voltage	C	IVIDCPWR_VAL_RANGE_VOLTAGE	1
	COM	IviDCPwrRangeVoltage	1
Configure Output Range Class Ext Base	C	IVIDCPWR_VAL_RANGE_TYPE_CLASS_EXT_BASE	500
	COM		
Configure Output Range Specific Ext Base	C	IVIDCPWR_VAL_RANGE_TYPE_SPECIFIC_EXT_BASE	1000
	COM		

Query Output State

Parameter: OutputState

COM Enumeration Name: IviDCPwrOutputStateEnum

<i>Value Name</i>	<i>Language</i>	<i>Identifier</i>	<i>Actual Value</i>
Constant Voltage	C	IVIDCPWR_VAL_OUTPUT_CONSTANT_VOLTAGE	0
	COM	IviDCPwrOutputConstantVoltage	0
Constant Current	C	IVIDCPWR_VAL_OUTPUT_CONSTANT_CURRENT	1
	COM	IviDCPwrOutputConstantCurrent	1
Over Voltage	C	IVIDCPWR_VAL_OUTPUT_OVER_VOLTAGE	2
	COM	IviDCPwrOutputOverVoltage	2
Over Current	C	IVIDCPWR_VAL_OUTPUT_OVER_CURRENT	3
	COM	IviDCPwrOutputOverCurrent	3
Unregulated	C	IVIDCPWR_VAL_OUTPUT_UNREGULATED	4
	COM	IviDCPwrOutputUnregulated	4
Query Output State Class Ext Base	C	IVIDCPWR_VAL_OUTPUT_STATE_CLASS_EXT_BASE	500
	COM		
Query Output State Specific Ext Base	C	IVIDCPWR_VAL_OUTPUT_STATE_SPECIFIC_EXT_BASE	1000
	COM		

Measure

Parameter: MeasurementType

<i>Value Name</i>	<i>Language</i>	<i>Identifier</i>	<i>Actual Value</i>
Current	C	IVIDCPWR_VAL_MEASURE_CURRENT	0
	COM	IviDCPwrMeasurementCurrent	0
Voltage	C	IVIDCPWR_VAL_MEASURE_VOLTAGE	1
	COM	IviDCPwrMeasurementVoltage	1
Measure Class Ext Base	C	IVIDCPWR_VAL_MEASURE_CLASS_EXT_BASE	500
	COM		
Measure Specific Ext Base	C	IVIDCPWR_VAL_MEASURE_SPECIFIC_EXT_BAS E	1000
	COM		

11 Error and Completion Code Value Definitions

The IviDCPwr class specification does not define any additional status codes.

12 IviDCPwr Hierarchies

12.1 IviDCPwr COM Hierarchy

The full IviDCPwr COM Hierarchy includes the Inherent Capabilities Hierarchy as defined in Section 4.1, *COM Inherent Capabilities of IVI-3.2: Inherent Capabilities Specification*. To avoid redundancy, the Inherent Capabilities are omitted here.

Table 12-1. IviDCPwr COM Hierarchy

COM Interface Hierarchy	Generic Name	Type
Outputs		
Count	OutputChannel Count	P
Name	OutputChannel Name	P
Item		
Enabled	Output Enabled	P
VoltageLevel	Voltage Level	P
TriggerSource	Trigger Source	P
TriggeredVoltageLevel	Triggered Voltage Level	P
TriggeredCurrentLimit	Triggered Current Limit	P
ConfigureCurrentLimit	Configure Current Limit	M
CurrentLimit	Current Limit	P
CurrentLimitBehavior	Current Limit Behavior	P
ConfigureOVP	Configure OVP	M
OVPLimit	OVP Limit	P
OVPEnabled	OVP Enabled	P
ConfigureRange	Configure Output Range	M
Measure	Measure	M
ResetOutputProtection	Reset Output Protection	M
QueryCurrentLimitMax	Query Current Limit Max	M
QueryVoltageLevelMax	Query Voltage Level Max	M
QueryState	Query Output State	M
Trigger		
Abort	Abort	M
Initiate	Initiate	M
SendSoftwareTrigger	Send Software Trigger	M

12.1.1 IviDCPwr COM Interfaces

In addition to implementing IVI inherent capabilities interfaces, IviDCPwr interfaces contain interface reference properties for accessing the following IviDCPwr interfaces:

- ? IviDCPwrOutputs
- ? IviDCPwrTrigger

The IviDCPwrOutputs interface contains methods and properties for accessing a collection of objects that implement the IviDCPwrOutput interface.

Table 12-2. **IviDCPwr Interface GUIDs** lists the interfaces that this specification defines and their GUIDs.

Table 12-2. IviDCPwr Interface GUIDs

Interface	GUID
IiIviDCPwr	{ 47ed51b6-a398-11d4-ba58-000064657374 }
IiIviDCPwrOutputs	{ 47ed51b7-a398-11d4-ba58-000064657374 }
IiIviDCPwrOutput	{ 47ed51b8-a398-11d4-ba58-000064657374 }
IiIviDCPwrTrigger	{ 47ed51b9-a398-11d4-ba58-000064657374 }

12.1.2 IviDCPwr COM Interface Reference Properties

Interface reference properties are used to navigate the IviDCPwr COM hierarchy. This section describes the interface reference properties that the IiIviDCPwr interface defines.

12.1.2.1 Outputs

Data Type	Access
IiIviDCPwrOutputs*	RO

COM Property Name

Outputs

Description

Returns a pointer to the IiIviDCPwrOutputs interface.

12.1.2.2 Trigger

Data Type	Access
IiIviDCPwrTrigger*	RO

COM Property Name

Trigger

Description

Returns a pointer to the IiIviDCPwrTrigger interface.

12.1.3 IviDCPwr COM Category

The IviDCPwr class COM Category shall be “IviDCPwr”, and the Category ID (CATID) shall be {47ed5153-a398-11d4-ba58-000064657374}.

12.2 IviDCPwr C Function Hierarchy

The IviDCPwr class function hierarchy is shown in the following table. The full IviDCPwr C Function Hierarchy includes the Inherent Capabilities Hierarchy as defined in Section 4.2, *C Inherent Capabilities* of IVI-3.2: *Inherent Capabilities Specification*. To avoid redundancy, the Inherent Capabilities are omitted here.

Table 12-3 IviDCPwr C Function Hierarchy

Name or Class	Function Name	
<i>Configuration</i> Configure Voltage Level Configure OVP Configure Current Limit Configure Output Range Configure Output Enabled Query Current Limit Max Query Voltage Level Max	IviDCPwr_ConfigureVoltageLevel IviDCPwr_ConfigureOVP IviDCPwr_ConfigureCurrentLimit IviDCPwr_ConfigureOutputRange IviDCPwr_ConfigureOutputEnabled IviDCPwr_QueryMaxCurrentLimit IviDCPwr_QueryMaxVoltageLevel	
<i>Triggering</i> Configure Trigger Source Configure Triggered	IviDCPwr_ConfigureTriggerSource IviDCPwr_ConfigureTriggeredVoltageLevel	
Voltage Level Configure Triggered	IviDCPwr_ConfigureTriggeredCurrentLimit	
Current Limit		
<i>Action</i> Initiate Abort Send SoftwareTrigger Query Output State Reset Output Protection Measure Get OutputChannel Name	IviDCPwr_Initiate IviDCPwr_Abort IviDCPwr_SendSoftwareTrigger IviDCPwr_QueryOutputState IviDCPwr_ResetOutputProtection IviDCPwr_Measure IviDCPwr_GetChannelName	

12.3 IviDCPwr C Attribute Hierarchy

The IviDCPwr class attribute hierarchy is shown in the following table. The full IviDCPwr C Attribute Hierarchy includes the Inherent Capabilities Hierarchy as defined in Section 4.2, *C Inherent Capabilities* of IVI-3.2: *Inherent Capabilities Specification*. To avoid redundancy, the Inherent Capabilities are omitted here.

Table 12-4. IviDCPwr C Attributes Hierarchy

Category or Generic Attribute Name	C Defined Constant
<i>Output</i>	
Current Limit	IVIDCPWR_ATTR_CURRENT_LIMIT
Current Limit Behavior	IVIDCPWR_ATTR_CURRENT_LIMIT_BEHAVIOR
OutputChannel Count	IVIDCPWR_ATTR_CHANNEL_COUNT
Output Enabled	IVIDCPWR_ATTR_OUTPUT_ENABLED
OVP Enabled	IVIDCPWR_ATTR_OVP_ENABLED
OVP Limit	IVIDCPWR_ATTR_OVP_LIMIT
Voltage Level	IVIDCPWR_ATTR_VOLTAGE_LEVEL
<i>Trigger</i>	
Trigger Source	IVIDCPWR_ATTR_TRIGGER_SOURCE
Triggered Current Limit	IVIDCPWR_ATTR_TRIGGERED_CURRENT_LIMIT
Triggered Voltage Level	IVIDCPWR_ATTR_TRIGGERED_VOLTAGE_LEVEL

Appendix A Specific Driver Development Guidelines

A.1 Introduction

This section describes situations driver developers should be aware of when developing a specific instrument driver that complies with the IviDCPwr class.

A.2 Disabling Unused Extensions

Specific drivers are required to disable extension capability groups that an application program does not explicitly use. The specific driver can do so by setting the attributes of an extension capability group to the values that this section recommends. A specific driver can set these values for all extension capability groups when the Initialize or functions execute. This assumes that the extension capability groups remain disabled until the application program explicitly uses them. For the large majority of instruments, this assumption is true.

Under certain conditions, a specific driver might have to implement a more complex approach. For some instruments, configuring a capability group might affect instrument settings that correspond to an unused extension capability group. If these instrument settings affect the behavior of the instrument, then this might result in an interchangeability problem. If this can occur, the specific driver must take appropriate action so that the instrument settings that correspond to the unused extension capability group do not affect the behavior of the instrument when the application program performs an operation that might be affected by those settings.

The remainder of this section recommends attribute values that effectively disable each extension capability group.

Disabling the IviDCPwrTrigger Extension Group

The IviDCPwrTrigger extension group affects the instrument behavior only when the user calls the Initiate function. Therefore, this specification does not recommend attribute values that disable the IviDCPwrTrigger extension group.

Disabling the IviDCPwrSoftwareTrigger Extension Group

The IviDCPwrSoftwareTrigger extension group affects the instrument behavior only when the user calls the Send Software Trigger function. Therefore, this specification does not recommend attribute values that disable the IviDCPwrSoftwareTrigger extension group.

Disabling the IviDCPwrMeasurement Extension Group

The IviDCPwrMeasurement extension group affects the instrument behavior only when the user calls the Measure function. Therefore, this specification does not recommend attribute values that disable the IviDCPwrMeasurement extension group.

A.3 Special Considerations for Configure Output Range

Some DC power supplies do not allow the user to explicitly specify an output's range. Instead, they automatically change the range based on the values the user requests for the voltage level, OVP limit, and current limit. For instruments that automatically change the range, the `IviDCPwr_ConfigureOutputRange` function should perform range checking to verify that its input parameters are valid, but should not perform any communication with the instrument or set any attributes.

A.4 Special Considerations for Initiate

After an `Initiate` call, the values of the `Voltage Level` and `Current Limit` attributes might change. If a driver caches these attribute values, the driver needs to invalidate them when `Initiate` is called.

A.5 Special Considerations for Current Limit and Current Limit Behavior

Not all DC power supplies implement the current limit and current limit behavior settings in the same manner that IviDCPwr specification defines. Instead, some power supplies have a current level setting that is always active and an over-current limit setting that can be enabled or disabled. In order to develop a driver for these power supplies that complies with the IviDCPwr specification, the driver developer can implement the driver as described in this section.

It can be difficult to implement an IVI driver when the physical instrument settings differ from the attributes that the class specification defines. For these cases, a preferred approach can be to implement hidden attributes in the specific driver that control the physical instrument settings. The public attributes that the class specification defines manipulate the hidden attributes that correspond to the instrument settings. For this example the specific driver might implement the following hidden attributes

Instr Current Level – A hidden attribute that sets the instrument’s current level.

Instr OCP Limit – A hidden attribute that sets the instrument’s over-current protection limit.

Instr OCP Enable – A hidden attribute that enables and disables the instrument’s over-current protection.

Using these hidden attributes, the driver developer implements the specific driver as follows:

- ? When the user sets the Current Limit attribute, the specific driver sets both the Instr Current Level attribute and the Instr OCP Limit to the value that the user requests.
- ? When the user sets the Current Limit Behavior attribute to Current Trip, the driver sets the Instr OCP Enable to True. When the user sets the Current Limit Behavior attribute to Current Regulate, the driver sets the Instr OCP Enable to False.

For one particular DC power supply, the over-current protection is disabled by setting the over-current limit to a value that is 120% of the maximum current that the power supply can produce. In this case the specific driver implementation changes as follows:

- ? The specific driver implements hidden attributes only for the current level and the over-current limit.
- ? When the user sets the Current Limit attribute the specific driver always sets the Instr Current Level attribute to the value that the user requests. If the Current Limit Behavior attribute is set to Trip, the specific driver also sets the Instr OCP Limit to the value that the user requests. Otherwise, the specific driver does not set the value of the Instr OCP Limit attribute.
- ? When the user sets the Current Limit Behavior attribute to Current Regulate, the driver sets the Instr OCP Limit to 120% of the maximum current that the power supply can produce. When the user sets the Current Limit Behavior attribute to Current Trip, the driver sets the Instr OCP Limit to the same value as the Current Level attribute.

Appendix B. Interchangeability Checking Rules

B.1 Introduction

IVI drivers have a feature called interchangeability checking. Interchangeability checking returns a warning when it encounters a situation where the application program might not produce the same behavior when the user attempts to use a different instrument.

B.2 When to Perform Interchangeability Checking

Interchangeability checking occurs when all of the following conditions are met:

- ? The Interchange Check attribute is set to True
- ? The user calls one of the following functions.
 - ? Measure
 - ? Initiate

B.3 Interchangeability Checking Rules

Interchangeability checking is performed on a capability group basis. When enabled, interchangeability checking is always performed on the base capability group. In addition, interchangeability checking is performed on extension capability groups for which the user has ever set any of the attributes of the group. If the user has never set any attributes of an extension capability group, interchangeability checking is not performed on that group.

In general interchangeability warnings are generated if the following conditions are encountered:

- ? An attribute that affects the behavior of the instrument is not in a state that the user specifies.
- ? The user sets a class driver defined attribute to an instrument-specific value.
- ? The user configures the value of an attribute that the class defines as read-only. In a few cases the class drivers define read-only attributes that specific drivers might implement as read/write.

The remainder of this section defines additional rules and exceptions for each capability group.

IviDCPwrBase Capability Group

No additional interchangeability rules or exceptions are defined for the IviDCPwrBase capability group.

IviDCPwrTrigger Capability Group

The driver needs to perform interchangeability checking on this group only when the function performing interchangeability checking is Initiate.

IviDCPwrSoftwareTrigger Capability Group

No additional interchangeability rules or exceptions are defined for the IviDCPwrSoftwareTrigger capability group.

IviDCPwrMeasurement Capability Group

No additional interchangeability rules or exceptions are defined for the IviDCPwrMeasurement capability group.

Appendix C. ANSI C Include File

The C source code below provides an example of how a class driver C interface might be defined. It provides definitions only for attributes, functions, values, and status codes that this specification defines. It does not represent a complete interface for an IviFgen compliant driver. To aid in the creation of an IviFgen compliant specific driver, replace IVIFGEN with the actual driver prefix using uppercase characters and replace IviFgen with consistent case sensitivity.

```
/*
 *
 * I V I - D C P W R
 *
 * Title:      IviDCPwr include file
 * Purpose:    IviDCPwr Class declarations for Inherent Capabilities and
 *             IviDCPwr Base and Extended Capabilities.
 */
*****/

#ifndef IVIDCPWR_HEADER
#define IVIDCPWR_HEADER

#include <ivic.h>

#if defined(__cplusplus) || defined(_cplusplus)
extern "C" {
#endif

/*
 *----- IviDCPwr Class Attribute Defines -----*
 */
*****/

/*- IviDCPwrBase Attributes -*/
#define IVIDCPWR_ATTR_VOLTAGE_LEVEL          (IVI_CLASS_ATTR_BASE + 1)
#define IVIDCPWR_ATTR_OVP_ENABLED           (IVI_CLASS_ATTR_BASE + 2)
#define IVIDCPWR_ATTR_OVP_LIMIT             (IVI_CLASS_ATTR_BASE + 3)
#define IVIDCPWR_ATTR_CURRENT_LIMIT_BEHAVIOR (IVI_CLASS_ATTR_BASE + 4)
#define IVIDCPWR_ATTR_CURRENT_LIMIT         (IVI_CLASS_ATTR_BASE + 5)
#define IVIDCPWR_ATTR_OUTPUT_ENABLED        (IVI_CLASS_ATTR_BASE + 6)
#define IVIDCPWR_ATTR_CHANNEL_COUNT         (IVI_CLASS_ATTR_BASE + 7)

/*- IviDCPwr Extended Attributes -*/
/*- Trigger Attributes -*/
#define IVIDCPWR_ATTR_TRIGGER_SOURCE         (IVI_CLASS_ATTR_BASE + 101)
#define IVIDCPWR_ATTR_TRIGGERED_CURRENT_LIMIT (IVI_CLASS_ATTR_BASE + 102)
#define IVIDCPWR_ATTR_TRIGGERED_VOLTAGE_LEVEL (IVI_CLASS_ATTR_BASE + 103)

/*
 *-----IviDCPwr Parameter Value Defines-----*
 */
*****/

/*- Defined values for rangeType parameter of function -*/
/*- IviDCPwr_ConfigureOutputRange -*/
#define IVIDCPWR_VAL_RANGE_CURRENT          (0L)
#define IVIDCPWR_VAL_RANGE_VOLTAGE         (1L)

#define IVIDCPWR_VAL_RANGE_TYPE_CLASS_EXT_BASE (500L)
#define IVIDCPWR_VAL_RANGE_TYPE_SPECIFIC_EXT_BASE (1000L)

/*- Defined values for outputState parameter of function -*/
/*- IviDCPwr_QueryOutputState -*/
#define IVIDCPWR_VAL_OUTPUT_CONSTANT_VOLTAGE (0L)
#define IVIDCPWR_VAL_OUTPUT_CONSTANT_CURRENT (1L)
#define IVIDCPWR_VAL_OUTPUT_OVER_VOLTAGE     (2L)
#define IVIDCPWR_VAL_OUTPUT_OVER_CURRENT    (3L)
#define IVIDCPWR_VAL_OUTPUT_UNREGULATED     (4L)

#define IVIDCPWR_VAL_OUTPUT_STATE_CLASS_EXT_BASE (500L)
#define IVIDCPWR_VAL_OUTPUT_STATE_SPECIFIC_EXT_BASE (1000L)
```

```

    /*- Defined values for measurementType parameter of function -*/
    /*- IviDCPwr_Measure -*/
#define IVIDCPWR_VAL_MEASURE_CURRENT          (0L)
#define IVIDCPWR_VAL_MEASURE_VOLTAGE        (1L)

#define IVIDCPWR_VAL_MEASURE_CLASS_EXT_BASE  (500L)
#define IVIDCPWR_VAL_MEASURE_SPECIFIC_EXT_BASE (1000L)

/*****
 *----- IviDCPwr Class Attribute Value Defines -----*
 *****/

    /*- Defined values for attribute IVIDCPWR_ATTR_CURRENT_LIMIT_BEHAVIOR -*/
#define IVIDCPWR_VAL_CURRENT_REGULATE        (0)
#define IVIDCPWR_VAL_CURRENT_TRIP           (1)

#define IVIDCPWR_VAL_CURRENT_LIMIT_BEHAVIOR_CLASS_EXT_BASE (500)
#define IVIDCPWR_VAL_CURRENT_LIMIT_BEHAVIOR_SPECIFIC_EXT_BASE (1000)

    /*- Defined values for attribute IVIDCPWR_ATTR_TRIGGER_SOURCE -*/
#define IVIDCPWR_VAL_TRIG_IMMEDIATE         (0)
#define IVIDCPWR_VAL_TRIG_EXTERNAL         (1)
#define IVIDCPWR_VAL_SOFTWARE_TRIG        (2)
#define IVIDCPWR_VAL_TRIG_TTL0            (3)
#define IVIDCPWR_VAL_TRIG_TTL1            (4)
#define IVIDCPWR_VAL_TRIG_TTL2            (5)
#define IVIDCPWR_VAL_TRIG_TTL3            (6)
#define IVIDCPWR_VAL_TRIG_TTL4            (7)
#define IVIDCPWR_VAL_TRIG_TTL5            (8)
#define IVIDCPWR_VAL_TRIG_TTL6            (9)
#define IVIDCPWR_VAL_TRIG_TTL7            (10)
#define IVIDCPWR_VAL_TRIG_ECL0            (11)
#define IVIDCPWR_VAL_TRIG_ECL1            (12)
#define IVIDCPWR_VAL_TRIG_PXI_STAR        (13)
#define IVIDCPWR_VAL_TRIG_RTSI_0          (14)
#define IVIDCPWR_VAL_TRIG_RTSI_1          (15)
#define IVIDCPWR_VAL_TRIG_RTSI_2          (16)
#define IVIDCPWR_VAL_TRIG_RTSI_3          (17)
#define IVIDCPWR_VAL_TRIG_RTSI_4          (18)
#define IVIDCPWR_VAL_TRIG_RTSI_5          (19)
#define IVIDCPWR_VAL_TRIG_RTSI_6          (20)

#define IVIDCPWR_VAL_TRIG_SRC_CLASS_EXT_BASE (500)
#define IVIDCPWR_VAL_TRIG_SRC_SPECIFIC_EXT_BASE (1000)

/*****
 *----- IviDCPwr Class Instrument Driver Function Declarations -----*
 *****/

ViStatus _VI_FUNC IviDCPwr_GetChannelName (ViSession Vi,
                                           ViInt32 Index,
                                           ViInt32 NameBufferSize,
                                           ViChar Name[]);

// The following functions should only be implemented with C Class Drivers
//
ViStatus _VI_FUNC IviDCPwr_GetSpecificDriverCHandle (ViSession Vi,
                                                    ViSession *SpecificDriverCHandle);
ViStatus _VI_FUNC IviDCPwr_GetSpecificDriverIUnknownPtr (ViSession Vi,
                                                         IUnknown **SpecificDriverIUnknownPtr);

// The following functions should only be implemented with C Wrappers
// over IVI-COM Specific Drivers.
//
ViStatus _VI_FUNC IviDCPwr_GetNativeIUnknownPtr (ViSession Vi,
                                                 IUnknown **NativeIUnknownPtr);
ViStatus _VI_FUNC IviDCPwr_AttachToExistingCOMSession (IUnknown *ExistingIUnknownPtr,
                                                       ViSession *Vi);

```

```

    /*- IviDCPwrBase Capability Group Functions -*/
ViStatus _VI_FUNC IviDCPwr_ConfigureOutputEnabled (ViSession Vi,
                                                    ViConstString ChannelName,
                                                    ViBoolean Enabled);

ViStatus _VI_FUNC IviDCPwr_ConfigureOutputRange (ViSession Vi,
                                                  ViConstString ChannelName,
                                                  ViInt32 RangeType,
                                                  ViReal64 Range);

ViStatus _VI_FUNC IviDCPwr_ConfigureCurrentLimit (ViSession Vi,
                                                  ViConstString ChannelName,
                                                  ViInt32 Behavior,
                                                  ViReal64 Limit);

ViStatus _VI_FUNC IviDCPwr_ConfigureOVP (ViSession Vi,
                                          ViConstString ChannelName,
                                          ViBoolean Enabled,
                                          ViReal64 Limit);

ViStatus _VI_FUNC IviDCPwr_ConfigureVoltageLevel (ViSession Vi,
                                                  ViConstString ChannelName,
                                                  ViReal64 Level);

ViStatus _VI_FUNC IviDCPwr_QueryOutputState (ViSession Vi,
                                              ViConstString ChannelName,
                                              ViInt32 OutputState,
                                              ViBoolean* InState);

ViStatus _VI_FUNC IviDCPwr_QueryMaxCurrentLimit (ViSession Vi,
                                                  ViConstString ChannelName,
                                                  ViReal64 VoltageLevel,
                                                  ViReal64* MaxCurrentLimit);

ViStatus _VI_FUNC IviDCPwr_QueryMaxVoltageLevel (ViSession Vi,
                                                  ViConstString ChannelName,
                                                  ViReal64 CurrentLimit,
                                                  ViReal64* MaxVoltageLevel);

ViStatus _VI_FUNC IviDCPwr_ResetOutputProtection (ViSession Vi,
                                                  ViConstString ChannelName);

    /*- IviDcPwrTrigger Functions -*/
ViStatus _VI_FUNC IviDCPwr_ConfigureTriggerSource (ViSession Vi,
                                                  ViConstString ChannelName,
                                                  ViInt32 Source);

ViStatus _VI_FUNC IviDCPwr_ConfigureTriggeredVoltageLevel (ViSession Vi,
                                                           ViConstString ChannelName,
                                                           ViReal64 Level);

ViStatus _VI_FUNC IviDCPwr_ConfigureTriggeredCurrentLimit (ViSession Vi,
                                                           ViConstString ChannelName,
                                                           ViReal64 Limit);

ViStatus _VI_FUNC IviDCPwr_Abort (ViSession Vi);

ViStatus _VI_FUNC IviDCPwr_Initiate (ViSession Vi);

    /*- IviDCPwrSoftwareTrigger Functions -*/
ViStatus _VI_FUNC IviDCPwr_SendSoftwareTrigger (ViSession Vi);

    /*- IviDCPwrMeasure Functions -*/
ViStatus _VI_FUNC IviDCPwr_Measure (ViSession Vi,
                                    ViConstString ChannelName,
                                    ViInt32 MeasurementType,
                                    ViReal64* Measurement);

ViStatus _VI_FUNC IviDCPwr_ConfigureOutputEnabled (ViSession Vi,

```

```

        ViConstString ChannelName,
        ViBoolean Enabled);

ViStatus _VI_FUNC IviDCPwr_ConfigureOutputRange (ViSession Vi,
        ViConstString ChannelName,
        ViInt32 RangeType,
        ViReal64 Range);

ViStatus _VI_FUNC IviDCPwr_ConfigureCurrentLimit (ViSession Vi,
        ViConstString ChannelName,
        ViInt32 Behavior,
        ViReal64 Limit);

ViStatus _VI_FUNC IviDCPwr_ConfigureOVP (ViSession Vi,
        ViConstString ChannelName,
        ViBoolean Enabled,
        ViReal64 Limit);

ViStatus _VI_FUNC IviDCPwr_ConfigureVoltageLevel (ViSession Vi,
        ViConstString ChannelName,
        ViReal64 Level);

ViStatus _VI_FUNC IviDCPwr_QueryOutputState (ViSession Vi,
        ViConstString ChannelName,
        ViInt32 OutputState,
        ViBoolean* InState);

ViStatus _VI_FUNC IviDCPwr_QueryMaxCurrentLimit (ViSession Vi,
        ViConstString ChannelName,
        ViReal64 VoltageLevel,
        ViReal64* MaxCurrentLimit);

ViStatus _VI_FUNC IviDCPwr_QueryMaxVoltageLevel (ViSession Vi,
        ViConstString ChannelName,
        ViReal64 CurrentLimit,
        ViReal64* MaxVoltageLevel);

ViStatus _VI_FUNC IviDCPwr_ResetOutputProtection (ViSession Vi,
        ViConstString ChannelName);

    /*- IviDcPwrTrigger Functions -*/
ViStatus _VI_FUNC IviDCPwr_ConfigureTriggerSource (ViSession Vi,
        ViConstString ChannelName,
        ViInt32 Source);

ViStatus _VI_FUNC IviDCPwr_ConfigureTriggeredVoltageLevel (ViSession Vi,
        ViConstString ChannelName,
        ViReal64 Level);

ViStatus _VI_FUNC IviDCPwr_ConfigureTriggeredCurrentLimit (ViSession Vi,
        ViConstString ChannelName,
        ViReal64 Limit);

ViStatus _VI_FUNC IviDCPwr_Abort (ViSession Vi);

ViStatus _VI_FUNC IviDCPwr_Initiate (ViSession Vi);

    /*- IviDCPwrSoftwareTrigger Functions -*/
ViStatus _VI_FUNC IviDCPwr_SendSoftwareTrigger (ViSession vi);

    /*- IviDCPwrMeasure Functions -*/
ViStatus _VI_FUNC IviDCPwr_Measure (ViSession Vi,
        ViConstString ChannelName,
        ViInt32 MeasurementType,
        ViReal64* Measurement);
/*****
 *----- IviDCPwr Class Error And Completion Codes -----*
 *****/
#define IVIDCPWR_ERROR_TRIGGER_NOT_SOFTWARE    (IVI_SHARED_COMPONENT_ERROR_BASE + 1)
/*****

```

```
*----- End Include File -----*
*****/
#if defined(__cplusplus) || defined(__cplusplus__)
}
#endif
#endif /* IVIDCPWR_HEADER */
```

Appendix D. COM IDL File

To ease the development of a compliant IVI-COM driver for the IviDCPwr class, the IVI Foundation publishes IDL (Interface Description Language) files that consolidate all the method and property definitions listed in this specification. Notice that the interface IviDCPwr derives from IiviDriver, It is described in IVI-3.2, *Inherent Capabilities Specification*.

These files along with these definitions compiled into type libraries are available from the IVI Foundation web site at <http://www.ivifoundation.org/>.

D.1 IviDCPwrTypeLib.idl

```
#if !defined(IVI_DCPWR_TYPELIB_IDL_INCLUDED_)
#define IVI_DCPWR_TYPELIB_IDL_INCLUDED_
/*****
 *
 * (C) COPYRIGHT INTERCHANGEABLE VIRTUAL INSTRUMENTS FOUNDATION, 2001,2002
 * All rights reserved.
 *
 *
 * FILENAME      : IviDCPwrTypeLib.idl
 *
 * STATUS        : UN-PUBLISHED.
 * COMPILER      : MSVC++ 6.0, sp4 MIDL
 * CONTENT       : IVI DC Power Supply Instrument Class Standard
 *               : type library definition
 *
 *****/

import "oidl.idl";
import "ocidl.idl";

[
    uuid(47ed5121-a398-11d4-ba58-000064657374),
    version(0.5),
    helpstring("IviDCPwr 0.5 (draft) Type Library"),
    helpfile("IviDCPwr.chm")
]
library IviDCPwrLib
{
    importlib("stdole32.tlb");
    importlib("stdole2.tlb");

    #include "IviDCPwr.idl"
};

#endif // !defined(IVI_DCPWR_TYPELIB_IDL_INCLUDED_)
```

D.2 IVIDCPwr.idl

```
#if !defined(IVI_DCPWR_IDL_INCLUDED_)
#define IVI_DCPWR_IDL_INCLUDED_
/*****
 *
 * (C) COPYRIGHT INTERCHANGEABLE VIRTUAL INSTRUMENTS FOUNDATION, 2001,2002
 * All rights reserved.
 *
 *
 * FILENAME      : IviDCPwr.idl
 *
 *****/
```

```

* STATUS      : UN-PUBLISHED.
* COMPILER    : MSVC++ 6.0, sp4 MIDL
* CONTENT     : IVI DC Power Supply Instrument Class Standard IDL
*
*****
/

#include <winerror.h>
import "oidl.idl";
import "ocidl.idl";

#ifdef IVI_USE_IMPORT
import "..\IviDriverTypeLib\IviDriver.idl";
#else
importlib ("..\TypeLibraries\IviDriverTypeLib.dll");
#endif

//-----
--
// Preprocessor Macros
//-----
--

#define HELP_DCPWR(x)  helpstring(HS_DCPWR_ ## x ## ), helpcontext(HC_DCPWR_
## x ## )

//-----
--
// Provides for Localization
//-----
--

#include "IviDCPwrEnglish.idl"

//-----
--
// Interface Declarations
//-----
--

interface IiIviDCPwr;
interface IiIviDCPwrOutputs;
interface IiIviDCPwrOutput;
interface IiIviDCPwrTrigger;

#define UUID_IIVI_DCPWR          47ed51b6-a398-11d4-ba58-000064657374
#define UUID_IIVI_DCPWR_OUTPUTS 47ed51b7-a398-11d4-ba58-000064657374
#define UUID_IIVI_DCPWR_OUTPUT  47ed51b8-a398-11d4-ba58-000064657374
#define UUID_IIVI_DCPWR_TRIGGER 47ed51b9-a398-11d4-ba58-000064657374

//-----
--

```

```

//      TYPEDEF ENUMS
//-----
--

[
    HELP_DCPWR(HRESULTS)
]
typedef enum IviDCPwrErrorCodesEnum
{
    E_IVIDCPWR_TRIGGER_NOT_SOFTWARE      = MAKE_HRESULT(SEVERITY_ERROR,
FACILITY_ITF, 0x1001),
} IviDCPwrErrorCodesEnum;

//-----
--
//      enum: IviDCPwrCurrentLimitBehaviorEnum
//-----
--

typedef
[
    public,
    v1_enum,
    HELP_DCPWR(CURRENT_LIMIT_BEHAVIOR_ENUM)
]
enum IviDCPwrCurrentLimitBehaviorEnum
{
    IviDCPwrCurrentLimitRegulate      = 0,
    IviDCPwrCurrentLimitTrip          = 1
} IviDCPwrCurrentLimitBehaviorEnum;

//-----
--
//      enum: IviDCPwrMeasurementTypeEnum
//-----
--

typedef
[
    public,
    v1_enum,
    HELP_DCPWR(MEASUREMENT_TYPE_ENUM)
]
enum IviDCPwrMeasurementTypeEnum
{
    IviDCPwrMeasurementCurrent        = 0,
    IviDCPwrMeasurementVoltage        = 1
} IviDCPwrMeasurementTypeEnum;

//-----
--
//      enum: IviDCPwrRangeTypeEnum
//-----
--

```

```

typedef
[
    public,
    vl_enum,
    HELP_DCPWR(RANGE_TYPE_ENUM)
]
enum IviDCPwrRangeTypeEnum
{
    IviDCPwrRangeCurrent          = 0,
    IviDCPwrRangeVoltage          = 1
} IviDCPwrRangeTypeEnum;

//-----
--
//  enum: IviDCPwrTriggerSourceEnum
//-----
--

typedef
[
    public,
    vl_enum,
    HELP_DCPWR(TRIGGER_SOURCE_ENUM)
]
enum IviDCPwrTriggerSourceEnum
{
    IviDCPwrTriggerSourceImmediate = 0,
    IviDCPwrTriggerSourceExternal  = 1,
    IviDCPwrTriggerSourceSwTrigFunc = 2,
    IviDCPwrTriggerSourceTTL0      = 3,
    IviDCPwrTriggerSourceTTL1      = 4,
    IviDCPwrTriggerSourceTTL2      = 5,
    IviDCPwrTriggerSourceTTL3      = 6,
    IviDCPwrTriggerSourceTTL4      = 7,
    IviDCPwrTriggerSourceTTL5      = 8,
    IviDCPwrTriggerSourceTTL6      = 9,
    IviDCPwrTriggerSourceTTL7      = 10,
    IviDCPwrTriggerSourceECL0      = 11,
    IviDCPwrTriggerSourceECL1      = 12,
    IviDCPwrTriggerSourcePXIStar    = 13,
    IviDCPwrTriggerSourceRTSI0      = 14,
    IviDCPwrTriggerSourceRTSI1      = 15,
    IviDCPwrTriggerSourceRTSI2      = 16,
    IviDCPwrTriggerSourceRTSI3      = 17,
    IviDCPwrTriggerSourceRTSI4      = 18,
    IviDCPwrTriggerSourceRTSI5      = 19,
    IviDCPwrTriggerSourceRTSI6      = 20
} IviDCPwrTriggerSourceEnum;

//-----
--
//  enum: IviDCPwrOutputStateEnum
//-----
--

```

```

typedef
[
    public,
    vl_enum,
    HELP_DCPWR(OUTPUT_STATE_ENUM)
]
enum IviDCPwrOutputStateEnum
{
    IviDCPwrOutputConstantVoltage    = 0,
    IviDCPwrOutputConstantCurrent    = 1,
    IviDCPwrOutputOverVoltage        = 2,
    IviDCPwrOutputOverCurrent        = 3,
    IviDCPwrOutputUnregulated        = 4
} IviDCPwrOutputStateEnum;

//-----
--
//  IVI DCPwr Driver Root Level Interface
//-----
--

[
    object,
    uuid(UUID_IIVI_DCPWR),
    HELP_DCPWR(I_IVI_DCPWR),
    oleautomation,
    pointer_default(unique)
]
interface IIVI_DCPwr : IIVI_Driver
{
//----- Outputs Interface Reference
    [propget, HELP_DCPWR(OUTPUTS)]
    HRESULT Outputs ([out, retval] IIVI_DCPwrOutputs **pVal);

//----- Trigger Interface Reference
    [propget, HELP_DCPWR(TRIGGER)]
    HRESULT Trigger ([out, retval] IIVI_DCPwrTrigger **pVal);
};

//-----
--
//  Outputs Interface
//-----
--

[
    object,
    uuid(UUID_IIVI_DCPWR_OUTPUTS),
    HELP_DCPWR(I_IVI_DCPWR_OUTPUTS),
    oleautomation,
    pointer_default(unique)
]

```

```

]
interface IIVI_DCPwrOutputs : IUnknown
{
//----- Item
    [ propget, HELP_DCPWR(ITEM) ]
    HRESULT Item (
        [in] BSTR OutputName,
        [out, retval] IIVI_DCPwrOutput **pVal);

//----- Count
    [ propget, HELP_DCPWR(COUNT) ]
    HRESULT Count ([out, retval] LONG *pVal);

//----- Name
    [ propget, HELP_DCPWR(NAME) ]
    HRESULT Name ([in] LONG Index, [out, retval] BSTR *Name);
};

//-----
--
// Output Interface
//-----
--

[
    object,
    uuid(UUID_IIVI_DCPWR_OUTPUT),
    HELP_DCPWR(I_IVI_DCPWR_OUTPUT),
    oleautomation,
    pointer_default(unique)
]
interface IIVI_DCPwrOutput : IUnknown
{
//----- ConfigureCurrentLimit
    [HELP_DCPWR(CONFIGURE_CURRENT_LIMIT)]
    HRESULT ConfigureCurrentLimit(
        [in] IIVI_DCPwrCurrentLimitBehaviorEnum Behavior,
        [in] DOUBLE Limit);

//----- ConfigureRange
    [HELP_DCPWR(CONFIGURE_RANGE)]
    HRESULT ConfigureRange(
        [in] IIVI_DCPwrRangeTypeEnum RangeType,
        [in] DOUBLE Range);

//----- ConfigureOVP
    [HELP_DCPWR(CONFIGURE_OVP)]
    HRESULT ConfigureOVP(
        [in] VARIANT_BOOL Enabled,
        [in] DOUBLE Limit);
};

```

```

//----- Measure
[HELP_DCPWR(MEASURE)]
HRESULT Measure (
    [in] IviDCPwrMeasurementTypeEnum MeasurementType,
    [out, retval] DOUBLE *Measurement);

//----- QueryCurrentLimitMax
[HELP_DCPWR(QUERY_CURRENT_LIMIT_MAX)]
HRESULT QueryCurrentLimitMax(
    [in] DOUBLE VoltageLevel,
    [out, retval] DOUBLE *MaxCurrentLimit);

//----- QueryState
[HELP_DCPWR(QUERY_STATE)]
HRESULT QueryState(
    [in] IviDCPwrOutputStateEnum OutputState,
    [out, retval] VARIANT_BOOL *InState);

//----- QueryVoltageLevelMax
[HELP_DCPWR(QUERY_VOLTAGE_LEVEL_MAX)]
HRESULT QueryVoltageLevelMax(
    [in] DOUBLE CurrentLimit,
    [out, retval] DOUBLE *MaxVoltageLevel);

//----- ResetOutputProtection
[HELP_DCPWR(RESET_OUTPUT_PROTECTION)]
HRESULT ResetOutputProtection();

//----- Enabled
[propput, HELP_DCPWR(ENABLED)]
HRESULT Enabled([in] VARIANT_BOOL newVal);

[propget, HELP_DCPWR(ENABLED)]
HRESULT Enabled([out, retval] VARIANT_BOOL *pVal);

//----- TriggerSource
[propput, HELP_DCPWR(SOURCE)]
HRESULT TriggerSource([in] IviDCPwrTriggerSourceEnum newVal);

[propget, HELP_DCPWR(SOURCE)]
HRESULT TriggerSource([out, retval] IviDCPwrTriggerSourceEnum *pVal);

//----- TriggeredCurrentLimit
[propput, HELP_DCPWR(TRIGGERED_CURRENT_LIMIT)]
HRESULT TriggeredCurrentLimit([in] DOUBLE newVal);

[propget, HELP_DCPWR(TRIGGERED_CURRENT_LIMIT)]
HRESULT TriggeredCurrentLimit([out, retval] DOUBLE *pVal);

```

```

//----- TriggeredVoltageLevel
    [propput, HELP_DCPWR(TRIGGERED_VOLTAGE_LEVEL)]
    HRESULT TriggeredVoltageLevel([in] DOUBLE newVal);

    [propget, HELP_DCPWR(TRIGGERED_VOLTAGE_LEVEL)]
    HRESULT TriggeredVoltageLevel([out, retval] DOUBLE *pVal);

//----- VoltageLevel
    [propput, HELP_DCPWR(VOLTAGE_LEVEL)]
    HRESULT VoltageLevel([in] DOUBLE newVal);

    [propget, HELP_DCPWR(VOLTAGE_LEVEL)]
    HRESULT VoltageLevel([out, retval] DOUBLE *pVal);

//----- CurrentLimit
    [propput, HELP_DCPWR(CURRENT_LIMIT)]
    HRESULT CurrentLimit([in] DOUBLE newVal);

    [propget, HELP_DCPWR(CURRENT_LIMIT)]
    HRESULT CurrentLimit([out, retval] DOUBLE *pVal);

//----- CurrentLimitBehavior
    [propput, HELP_DCPWR(CURRENT_LIMIT_BEHAVIOR)]
    HRESULT CurrentLimitBehavior(
        [in] IviDCPwrCurrentLimitBehaviorEnum newVal);

    [propget, HELP_DCPWR(CURRENT_LIMIT_BEHAVIOR)]
    HRESULT CurrentLimitBehavior(
        [out, retval] IviDCPwrCurrentLimitBehaviorEnum *pVal);

//----- OVPLimit
    [propput, HELP_DCPWR(OVP_LIMIT)]
    HRESULT OVPLimit([in] DOUBLE newVal);

    [propget, HELP_DCPWR(OVP_LIMIT)]
    HRESULT OVPLimit([out, retval] DOUBLE *pVal);

//----- OVPEnabled
    [propput, HELP_DCPWR(OVP_ENABLED)]
    HRESULT OVPEnabled([in] VARIANT_BOOL newVal);

    [propget, HELP_DCPWR(OVP_ENABLED)]
    HRESULT OVPEnabled([out, retval] VARIANT_BOOL *pVal);

};

//-----
--
//  Trigger Interface

```

```

//-----
--

[
    object,
    uuid(UUID_IIVI_DCPWR_TRIGGER),
    HELP_DCPWR(I_IVI_DCPWR_TRIGGER),
    oleautomation,
    pointer_default(unique)
]
interface IIVI_DCPwrTrigger : IUnknown
{
//----- Abort
    [HELP_DCPWR(ABORT)]
    HRESULT Abort();

//----- Initiate
    [HELP_DCPWR(INITIATE)]
    HRESULT Initiate();

//----- SendSoftwareTrigger
    [HELP_DCPWR(SEND_SOFTWARE_TRIGGER)]
    HRESULT SendSoftwareTrigger();
};

#endif // !defined(IVI_DCPWR_IDL_INCLUDED_)

```

D.3 IviDCPwrEnglish.idl

This file contains help strings and contexts for the interfaces, methods and properties.

```
#if !defined(IVI_DCPWR_IDL_ENGLISH_INCLUDED_)
#define IVI_DCPWR_IDL_ENGLISH_INCLUDED_
/*****
 *
 * (C) COPYRIGHT INTERCHANGEABLE VIRTUAL INSTRUMENTS FOUNDATION, 2001,2002
 * All rights reserved.
 *
 * FILENAME      : IviDCPwrEnglish.idl
 *
 * STATUS        : UN-PUBLISHED.
 * COMPILER      : MSVC++ 6.0, sp4 MIDL
 * CONTENT       : IVI DCPwr Instrument Class Standard IDL
 *               help context IDs and help strings
 *
 *****/

#define HC_DCPWR_BASE 100

//-----
//      TYPEDEF ENUMS
//-----

#define HC_DCPWR_HRESULTS                HC_DCPWR_BASE + 0

#define HC_DCPWR_CURRENT_LIMIT_BEHAVIOR_ENUM    HC_DCPWR_BASE + 1
#define HC_DCPWR_MEASUREMENT_TYPE_ENUM         HC_DCPWR_BASE + 2
#define HC_DCPWR_RANGE_TYPE_ENUM              HC_DCPWR_BASE + 3
#define HC_DCPWR_TRIGGER_SOURCE_ENUM          HC_DCPWR_BASE + 4
#define HC_DCPWR_OUTPUT_STATE_ENUM            HC_DCPWR_BASE + 5

//-----
//      IviDCPwr Class Root Level Interface
//-----

#define HC_DCPWR_I_IVI_DCPWR                HC_DCPWR_BASE + 10
#define HC_DCPWR_OUTPUTS                     HC_DCPWR_BASE + 11
#define HC_DCPWR_TRIGGER                     HC_DCPWR_BASE + 12

//-----
//      Ivi DCPwr Outputs Interface
//-----

#define HC_DCPWR_I_IVI_DCPWR_OUTPUTS        HC_DCPWR_BASE + 20
#define HC_DCPWR_ITEM                       HC_DCPWR_BASE + 21
#define HC_DCPWR_COUNT                      HC_DCPWR_BASE + 22
#define HC_DCPWR_NAME                       HC_DCPWR_BASE + 23

//-----
//      Ivi DCPwr Output Interface
//-----

#define HC_DCPWR_I_IVI_DCPWR_OUTPUT        HC_DCPWR_BASE + 30
#define HC_DCPWR_CONFIGURE_CURRENT_LIMIT    HC_DCPWR_BASE + 31
#define HC_DCPWR_CONFIGURE_RANGE           HC_DCPWR_BASE + 32
#define HC_DCPWR_CONFIGURE_OVP             HC_DCPWR_BASE + 33
#define HC_DCPWR_ENABLED                   HC_DCPWR_BASE + 34
#define HC_DCPWR_MEASURE                   HC_DCPWR_BASE + 35
#define HC_DCPWR_QUERY_CURRENT_LIMIT_MAX   HC_DCPWR_BASE + 36
```

```

#define HC_DCPWR_QUERY_STATE                HC_DCPWR_BASE + 37
#define HC_DCPWR_QUERY_VOLTAGE_LEVEL_MAX   HC_DCPWR_BASE + 38
#define HC_DCPWR_RESET_OUTPUT_PROTECTION   HC_DCPWR_BASE + 39
#define HC_DCPWR_TRIGGERED_CURRENT_LIMIT   HC_DCPWR_BASE + 40
#define HC_DCPWR_TRIGGERED_VOLTAGE_LEVEL   HC_DCPWR_BASE + 41
#define HC_DCPWR_VOLTAGE_LEVEL             HC_DCPWR_BASE + 42
#define HC_DCPWR_CURRENT_LIMIT             HC_DCPWR_BASE + 43
#define HC_DCPWR_CURRENT_LIMIT_BEHAVIOR    HC_DCPWR_BASE + 44
#define HC_DCPWR_OVP_LIMIT                 HC_DCPWR_BASE + 45
#define HC_DCPWR_OVP_ENABLED               HC_DCPWR_BASE + 46

//-----
//  IviDCPwr Trigger Interface
//-----

#define HC_DCPWR_I_IVI_DCPWR_TRIGGER        HC_DCPWR_BASE + 50
#define HC_DCPWR_ABORT                     HC_DCPWR_BASE + 51
#define HC_DCPWR_INITIATE                  HC_DCPWR_BASE + 52
#define HC_DCPWR_SEND_SOFTWARE_TRIGGER     HC_DCPWR_BASE + 53
#define HC_DCPWR_SOURCE                    HC_DCPWR_BASE + 54

//-----
//  Help Strings
//-----

//-----
//  TYPEDEF ENUMS
//-----

#define HS_DCPWR_HRESULTS \
"IviDCPwr class defined HRESULTS"

#define HS_DCPWR_CURRENT_LIMIT_BEHAVIOR_ENUM \
"IviDCPwr class-compliant values for the CurrentLimitBehavior property in the Output interface."

#define HS_DCPWR_MEASUREMENT_TYPE_ENUM \
"IviDCPwr class-compliant values for the MeasurementType parameter of the Measure method in
Output interface."

#define HS_DCPWR_RANGE_TYPE_ENUM \
"IviDCPwr class-compliant values for the RangeType parameter of the ConfigureRange method in the
Output interface."

#define HS_DCPWR_OUTPUT_STATE_ENUM \
"IviDCPwr class-compliant values for the OutputState parameter of the QueryState method in the
Output interface."

#define HS_DCPWR_TRIGGER_SOURCE_ENUM \
"IviDCPwr class-compliant values for the Source property in the Trigger interface."

//-----
//  IviDCPwr Class Root Level Interface
//-----

#define HS_DCPWR_I_IVI_DCPWR \
"IviDCPwr class-compliant root interface"

#define HS_DCPWR_OUTPUTS \
"Pointer to the class-compliant IiVidCpwrOutputs interface"

#define HS_DCPWR_TRIGGER \
"Pointer to the class-compliant IiVidCpwrTrigger interface"

//-----
//  Ivi DCPwr Outputs Interface
//-----

```

```

#define HS_DCPWR_I_IVI_DCPWR_OUTPUTS \
"IviDCPwr class-compliant IiVidCPwrOutputs collection interface"

#define HS_DCPWR_ITEM \
"Pointer to a class-compliant IiVidCPwrOutput interface. The OutputName \
parameter may be a string defined by the driver or supplied as a virtual \
name in the configuration store."

#define HS_DCPWR_COUNT \
"property Count contains how many outputs this DC power supply has. It is \
also the maximum valid value for the Index parameter to the Name method of \
the Outputs interface."

#define HS_DCPWR_NAME \
"property Name is the string associated by the driver to the value of Index \
without ever referring to the configuration store."

//-----
//   IviDCPwr Output Interface
//-----

#define HS_DCPWR_I_IVI_DCPWR_OUTPUT \
"IviDCPwr class-compliant IiVidCPwrOutput interface"

#define HS_DCPWR_CONFIGURE_CURRENT_LIMIT \
"method Configure Current Limit specifies the output current limit value and \
the behavior of the power supply when the output current is greater than or \
equal to that value."

#define HS_DCPWR_CONFIGURE_RANGE \
"method Configure Range specifies the output's range, either current or \
voltage."

#define HS_DCPWR_CONFIGURE_OVP \
"method Configure OVP specifies the over-voltage limit and the behavior of \
the power supply when the output voltage is greater than or equal to the \
limit."

#define HS_DCPWR_MEASURE \
"method Measure takes a measurement on the output signal and returns the \
measured value."

#define HS_DCPWR_QUERY_CURRENT_LIMIT_MAX \
"method Query Current Limit Max returns the maximum programmable current \
limit that the power supply accepts for a particular voltage level on an \
output."

#define HS_DCPWR_QUERY_STATE \
"method Query State returns whether the power supply is in a particular \
output state."

#define HS_DCPWR_QUERY_VOLTAGE_LEVEL_MAX \
"method Query Voltage Level Max returns the maximum programmable voltage \
level that the power supply accepts for a particular current limit on an \
output."

#define HS_DCPWR_RESET_OUTPUT_PROTECTION \
"method Reset Output Protection resets the power supply's output protection \
after an over-voltage or over-current condition occurs."

#define HS_DCPWR_ENABLED \
"property Enabled specifies whether the signal the power supply produces \
appears at the output connector."

#define HS_DCPWR_TRIGGERED_CURRENT_LIMIT \
"property Triggered Current Limit specifies the value to which the power \
supply sets the current limit after a trigger event occurs."

#define HS_DCPWR_TRIGGERED_VOLTAGE_LEVEL \
"property Triggered Voltage Level specifies the value to which the power \

```

```

supply sets the voltage level after a trigger event occurs. "

#define HS_DCPWR_VOLTAGE_LEVEL \
"property Voltage Level specifies the voltage level the DC power supply \
attempts to generate. The units are Volts."

#define HS_DCPWR_CURRENT_LIMIT \
"property Current Limit specifies the output current limit. The units are \
Amps."

#define HS_DCPWR_CURRENT_LIMIT_BEHAVIOR \
"property Current Limit Behavior specifies the behavior of the power supply \
when the output current is equal to or greater than the value of the Current Limit property."

#define HS_DCPWR_OVP_LIMIT \
"property OVP Limit specifies the voltage the power supply allows. The units \
are Volts."

#define HS_DCPWR_OVP_ENABLED \
"property OVP Enabled specifies whether the power supply provides \
over-voltage protection. If this property is set to True, the power \
supply disables the output when the output voltage is greater than or \
equal to the OVP Limit."

//-----
//   IviDCPwr Trigger Interface
//-----

#define HS_DCPWR_I_IVI_DCPWR_TRIGGER \
"IviDCPwr class-compliant IiIviDCPwrTrigger interface"

#define HS_DCPWR_ABORT \
"method Abort returns the power supply to the ignore triggers state if the \
power supply is currently waiting for a trigger to change the output signal. \
If the power supply is not waiting for a trigger, this method does nothing."

#define HS_DCPWR_INITIATE \
"method Initiate causes the power supply to wait for a trigger if the power \
supply is not currently waiting for a trigger. If the power supply is already \
waiting for a trigger, this method does nothing "

#define HS_DCPWR_SEND_SOFTWARE_TRIGGER \
"method Send Software Trigger supplies a trigger signal when the trigger \
source is set to Software Trigger"

#define HS_DCPWR_SOURCE \
"property Source specifies the trigger source."

#endif // !defined(IVI_DCPWR_IDL_ENGLISH_INCLUDED_)

```