![IVI Interchangeable Virtual Instruments logo]

IVI-4.7: IviPwrMeter Class Specification

October 14, 2016 Edition
Revision 2.0

Important Information

The IVI-4.7: IviPwrMeter Class Specification 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.

The IVI Foundation wants to receive your comments on this specification. You can contact the Foundation through the web site at www.ivifoundation.org.
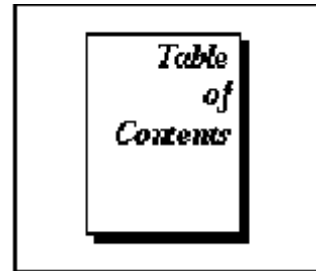
Warranty

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.

5. IviPwrMeterChannelAcquisition Extension Group 50

6. IviPwrMeterManualRange Extension Group 61

7. IviPwrMeterTriggerSource Extension Group 68

8. IviPwrMeterInternalTrigger Extension Group 76

15. IviPwrMeter Attribute ID Definitions 120

16. IviPwrMeter Attribute Value Definitions 121

17. IviPwrMeter Function Parameter Value Definitions 124

18. IviPwrMeter Error, Completion Code, and Exception Class Definitions   129

19. IviPwrMeter Hierarchies 132

Specific Driver Development Guidelines 143

Interchangeability Checking Rules...145

IviPwrMeter Class Specification

IviPwrMeter Revision History

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

Table 1. IviPwrMeter Class Specification Revisions

Revision Number	Date of Revision	Revision Notes
Revision 1.0	February 2002	First Approved Version.
Revision 1.0	April 2008	Editorial change to update the IVI Foundation contact information in the Important Information section to remove obsolete address information and refer only to the IVI Foundation web site.
Revision 1.0	April 2009	Editorial change to update repeated capabilities section to include both qualified and unqualified repeated capability names.
Revision 2.0	June 9, 2010	Incorporated IVI.NET
Revision 2.0	August 25, 2011	Editorial IVI.NET change. Change references to process-wide locking to AppDomain-wide locking. Add an overload to the Create factory method that takes locking related parameters.
Revision 2.0	June 21, 2013	Editorial IVI.NET change. Change the .NET MeasurementState property (section 4.2.6) to a method named GetMeasurementComplete (insert section 4.3.10). Modify all related sections (16, 17, 19.1) and renumber sections as needed.
Revision 2.0	September 24, 2015	Editorial Change – Clarified the use of one-based index for COM, and zero-based index for .NET for repeated capabilities in section 4.2.4.
Revision 2.0	October 14, 2016	Editorial Change – Modified header text for table 18.2 to indicate that the messages do not apply to .NET exceptions..

API Versions

Architecture	Drivers that comply with version 2.0 comply with all of the versions below

C	1.0, 2.0
COM	1.0, 2.0
.NET	2.0

Drivers that comply with this version of the specification also comply with earlier, compatible, versions of the specification as shown in the table above. The driver may benefit by advertising that it supports all the API versions listed in the table above.

1. Overview of the IviPwrMeter Specification

1.1 Introduction

This section introduces the *IviPwrMeter Class Specification*. This section summarizes the *IviPwrMeter Class 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:

- IviPwrMeter Class Overview

- References

- The definitions of terms and acronyms

1.2 IviPwrMeter Class Overview

This specification defines the IVI class for RF power meters. The IviPwrMeter class is designed to support the typical power meter as well as common extended functionality found in more complex instruments. The IviPwrMeter class conceptualizes a power meter as an instrument that can measure the average RF power of an input signal and can be applied to several different instruments.

The IviPwrMeter class is divided into a base capability group and several extension capability groups. The base capability group is used to configure a power meter for a typical measurement (this includes setting the units, the auto range mode, the auto averaging mode, and the correction frequency), initiating a measurement, and returning a measured value. The base capability group supports both single and dual channel measurements. The IviPwrMeter base capability group is described in Section 4, *IviPwrMeterBase Capability Group*.

The IviPwrMeter class also contains extension groups that configure the advanced trigger settings, the manual range, the averaging count, and the reference oscillator of the power meter. The class also contains extension groups that perform zero correction and calibration.

1.3 References

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

- IVI-3.1: Driver Architecture Specification

- IVI-3.2: Inherent Capabilities Specification

- IVI-3.3: Standard Cross Class Capabilities Specification

- IVI-3.4: API Style Guide

- IVI-3.18: IVI.NET Utility Classes and Interfaces Specification

- IVI- 5.0: Glossary

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. IviPwrMeter Class Capabilities

2.1 Introduction

The IviPwrMeter specification divides generic power meter capabilities into a base capability group and several extension capability groups. Each capability group will be 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 IviPwrMeter Group Names

The capability group names for the IviPwrMeter class are defined below. The Group Name is used to represent a particular capability group and is returned as one of the possible group names from the Class Group Capabilities attribute. Refer to *IVI-3.2: Inherent Capabilities Specification* for a description of the Class Group Capabilities attribute.

Table 2-1. IviPwrMeter Group Names

Capability Group Name	Description
IviPwrMeterBase	Base Capability Group: Power Meter that complies with the IviPwrMeterBase capability group.
IviPwrMeterChannelAcquisition	Channel Acquisition Extension Group: Power Meter that can acquire a measurement from a specified channel.
IviPwrMeterManualRange	Manual Range Extension Group: Power Meter that can manually set the measurement range.
IviPwrMeterTriggerSource	Trigger Source Extension Group: Power Meter that is able to specify a trigger source.
IviPwrMeterInternalTrigger	Internal Trigger Extension Group: Power Meter that can trigger internally on the measurement signal.
IviPwrMeterSoftwareTrigger	Software Trigger Extension Group: Power Meter that can trigger on a software trigger signal.
IviPwrMeterAveragingCount	Averaging Count Extension Group: Power Meter that can specify an averaging count in manual averaging mode.
IviPwrMeterZeroCorrection	Zero Correction Extension Group: Power Meter that can perform zero correction on an input channel for a given power sensor.
IviPwrMeterDutyCycleCorrection	Duty Cycle Correction Extension Group: Power Meter that can perform a duty cycle correction.
IviPwrMeterCalibration	Calibration Extension Group: Power Meter that can perform calibration for a given power sensor.
IviPwrMeterReferenceOscillator	Reference Oscillator Extension Group: Power Meter that can enable and configure an internal reference oscillator.

Refer to Section 16, *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 IviPwrMeter 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*, Section 4.3.9, *Repeated Capabilities*, and Section 5.9, *Repeated Capability Identifiers and Selectors*.

- Channel

2.3.1 Channel

In the configuration store, the repeated capability name for the channel repeated capability shall be exactly one of "Channel" or "IviPwrMeterChannel". Drivers that implement multiple repeated capabilities with the name "channel" shall use the latter form to disambiguate the names.

2.4 Boolean Attribute and Parameter Values

This specification uses True and False as the values for Boolean attributes and parameters. The following table defines the identifiers that are used for True and False in the IVI.NET, IVI-COM, and IVI-C architectures.

Boolean Value	IVI.NET Identifier	IVI-COM Identifier	IVI-C Identifier
True	`true`	`VARIANT_TRUE`	`VI_TRUE`
False	`false`	`VARIANT_FALSE`	`VI_FALSE`

2.5 .NET Namespace

The .NET namespace for the IviPwrMeter class is `Ivi.PwrMeter`.

2.6 .NET IviPwrMeter Session Factory

The IviPwrMeter .NET assembly contains a factory method called Create for creating instances of IviPwrMeter class-compliant IVI.NET drivers from driver sessions and logical names. Create is a static method accessible from the static IviPwrMeter class.

Refer to *IVI-3.5: Configuration Server Specification* for a description of how logical names and session names are defined in the configuration store.

Refer to Section 8, *IVI.NET Specific Driver Constructor*, of *IVI-3.2: Inherent Capabilities Specification*, for more details on how the `idQuery`, `reset`, and `options` parameters affect the instantiation of the driver.

Refer to Section 4.3.11, *Multithread Safety*, of *IVI-3.1: Driver Architecture Specification* for a complete description of IVI.NET driver locking. Refer to Section 8, Table 8.2 *Required Lock Type Behavior for Drivers With the Same Access Key*, of *IVI-3.2, Inherent Capability Specification*, for an explanation of how the values for `lockType` and `accessKey` are used to determine the kind of multithreaded lock to use for the driver instance.

.NET Method Prototype

```
IIviPwrMeter Ivi.PwrMeter.Create(String name);
```

```
IIviPwrMeter Ivi.PwrMeter.Create(String name,
                                 Boolean idQuery,
                                 Boolean reset);
IIviPwrMeter Ivi.PwrMeter.Create(String name,
                                 Boolean idQuery,
                                 Boolean reset,
                                 String options);

IIviPwrMeter Ivi.PwrMeter.Create(String resourceName,
                                 Boolean idQuery,
                                 Boolean reset,
                                 LockType lockType,
                                 String accessKey,
                                 String options);
```

Parameters

Inputs	Description	Base Type
name	A session name or a logical name that points to a session that uses an IVI.NET IviPwrMeter class-compliant driver.	String
idQuery	Specifies whether to verify the ID of the instrument. The default is False.	Boolean
reset	Specifies whether to reset the instrument. The default is False.	Boolean
lockType	Specifies whether to use AppDomain-wide locking or machine-wide locking.	Ivi.Driver.LockType
accessKey	Specifies a user-selectable access key to identify the lock. Driver instances that are created with the same accessKey will be protected from simultaneous access by multiple threads within an AppDomain or across AppDomains, depending upon the value of the lockType parameter.	String
options	A string that allows the user to specify the initial values of certain inherent attributes. The default is an empty string.	String

Outputs	Description	Base Type
Return Value	Interface pointer to the IIviPwrMeter interface of the driver referenced by session.	IIviPwrMeter

Defined Values

Name	Description		
	Language	Identifier	
AppDomain	The lock is AppDomain-wide.		
	.NET	Ivi.Driver.LockType.AppDomain	
Machine	The lock is machine-wide.		
	.NET	Ivi.Driver.LockType.Machine	

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

Usage

To create a driver that implements the IviPwrMeter instrument class API from the logical name "My LogicalName" use the following:

```
IIviPwrMeter pwrMeter = IviPwrMeter.Create("MyLogicalName");
```

In this case, the ID of the instrument will not be verified, the instrument will not be reset, and options will be supplied from the configuration store and/or driver defaults.

3. General Requirements

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

3.1 Minimum Class Compliance

To be compliant with the IviPwrMeter Class Specification, an IVI specific driver shall conform to all of the requirements for an IVI class-compliant specific driver as specified in *IVI-3.1: Driver Architecture Specification,* implement the inherent capabilities that *IVI-3.2: Inherent IVI Capabilities Specification* defines, and implements the IviPwrMeterBase capability group.

3.1.1 Disable

Refer to *IVI-3.2: Inherent Capabilities Specification* for the prototype of this function. The IviPwrMeter specification does not define additional requirements on the Disable function.

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. IviPwrMeterBase Capability Group

4.1 Overview

The IviPwrMeterBase capability group supports power meters that take a single measurement on one channel as well as instruments that can take synchronous measurements on two channels.

The IviPwrMeterBase capability group defines attributes and their values to configure the type of measurement and how the measurement is to be performed. These attributes include the units, the auto-range mode, the auto-averaging mode, the correction frequency, and the offset. The IviPwrMeterBase capability group also includes functions for configuring the power meter and for initiating and retrieving measurements.

4.2 IviPwrMeterBase Attributes

The IviPwrMeterBase capability group defines the following attributes:

- Averaging Auto Enabled

- Channel Count

- Channel Item (IVI-COM & IVI.NET Only)

- Channel Name (IVI-COM & IVI.NET Only)

- Measurement State

- Correction Frequency

- Offset

- Range Auto Enabled

- Units

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions*.

4.2.1 Averaging Auto Enabled

Data Type	Access	Applies To	Coercion	High Level Functions
ViBoolean	R/W	Channel	None	Configure Averaging Auto Enabled

.NET Property Name

    Channels[].Averaging.CountAuto

COM Property Name

    Channels.Item().Averaging.AutoEnabled

C Constant Name

    IVIPWRMETER_ATTR_AVERAGING_AUTO_ENABLED

Description

Specifies the auto-averaging mode used by the instrument for the specified input channel.

If this attribute is True, the instrument determines the best value for the averaging count automatically. The averaging count specifies the number of samples that the instrument takes before the measurement is complete.

If this attribute is False, the user specifies the averaging count explicitly by setting the Averaging Count attribute. Refer to Section 11.2.1, *Averaging Count* for more information.

If the driver implements the IviPwrMeterAveragingCount extension group, the actual averaging count the Power Meter is currently using can be determined from the Averaging Count attribute. If the driver does not implement the IviPwrMeterAveragingCount extension group, the driver shall return a Value Not Supported error (C/COM) or throw a Value Not Supported Exception (.NET) if the calling program attempts to set Auto Averaging Enabled to False.

For C and COM, if the Averaging Count attribute is set, the state of auto-averaging is indeterminate. For .NET, if the Averaging Count attribute is set, auto-averaging is disabled.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

Compliance Notes

1. If a specific driver implements the False value for this attribute, it shall also implement the IviPwrMeterAveragingCount extension group.

4.2.2 Channel Count

Data Type	Access	Applies to	Coercion	High Level Functions
ViInt32	RO	IIviPwrMeterChannel	None	None

.NET Property Name

> Channels.Count;
>
> This property is inherited from `IIviRepeatedCapabilityCollection`.

COM Property Name

> Channels.Count

C Constant Name

> IVIPWRMETER_ATTR_CHANNEL_COUNT

Description

> Returns the number of currently available channels. The count returned includes any of the supported reserved repeated capability names defined in Section 2.3, *Repeated Capability Names* as well as any custom repeated capability identifiers.

.NET Exceptions

> The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

Compliance Note

> The attribute ID value for this attribute shall be IVI_INHERENT_ATTR_BASE + 203.

4.2.3 Channel Item (IVI-COM & IVI.NET Only)

Data Type	Access	Applies to	Coercion	High Level Functions
IIviPwrMeterChannel*	RO	Channel	None	None

.NET Property Name

```
Channels[String name];
```

This indexer is inherited from `IIviRepeatedCapabilityCollection`. The name parameter uniquely identifies a particular channel in the Channel collection.

COM Property Name

```
Channels.Item ([in] BSTR ChannelName);
```

C Constant Name

```
N/A
```

Description

Channel Item uniquely identifies a channel in the channels collection. It returns an interface pointer which can be used to control the attributes and other functionality of that channel.

The Item property takes a channel name. If the user passes an invalid value for the `ChannelName` parameter, the property returns an error.

Valid Names include physical repeated capability identifiers and virtual repeated capability identifiers.

Parameters

Inputs	Description	Datatype
name (.NET) Channel Name (COM)	Specifies the name of the channel to retrieve.	ViConstString
index	Specifies the 0-based index of the channel to retrieve.	ViInt32

Return Values (C/COM)

If the IVI-COM driver cannot recognize the `Name` parameter, it returns an Unknown Name in Selector completion code as described in *IVI-3.2: Inherent Capabilities Specification*, Section 9.3.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

4.2.4 Channel Name (IVI-COM & IVI.NET Only)

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

.NET Property Name

```
Channels[].Name
```

This property is inherited from `IIviRepeatedCapabilityIdentification`.

COM Property Name

```
HRESULT Channels.Name ([in] LONG ChannelIndex);
```

C Constant Name

```
N/A.
```

Use the Get Channel Name function.

Description

Returns the physical repeated capability identifier defined by the specific driver for the channel that corresponds to the index that the user specifies. If the driver defines a qualified channel name, this property returns the qualified name.

In COM, the index is one-based. In .NET, the index is zero-based.

For C and COM, valid values for the `ChannelIndex` parameter are between one and the value of the Channel Count attribute, inclusive. If the user passes an invalid value for the `ChannelIndex` parameter, the value of this attribute is an empty string.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

4.2.5 Correction Frequency

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Real 64	R/ W	Channel	None	Configure Correction Frequency

.NET Property Name

    Channels[].CorrectionFrequency

COM Property Name

    Channels.Item().CorrectionFrequency

C Constant Name

    IVIPWRMETER_ATTR_CORRECTION_FREQUENCY

Description

Specifies the frequency of the input signal in Hertz. The instrument uses this value to determine the appropriate correction factor for the sensor.

In order to obtain the most accurate measurement, the user should specify the correction frequency as close as possible to the actual frequency of the input signal.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

4.2.6 Offset

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Real 64	R/ W	Channel	None	Configure Offset

.NET Property Name

> Channels[].Offset

COM Property Name

> Channels.Item().Offset

C Constant Name

> IVIPWRMETER_ATTR_OFFSET

Description

> Specifies an offset to be added to the measured value in units of dB. This attribute can be used to compensate for system losses or gains between the unit under test and the sensor of the power meter.
>
> A positive value shall be used for loss compensation whereas a negative value shall be used for gain compensation. For example, a cable loss of 2 dB could be compensated for by setting this attribute to +2. Similarly, a gain stage of 10 dB could be accounted for by setting the value of this attribute to −10. In both cases, the reading from the power meter will indicate the power at the unit under test rather than power at the power meter's sensor.

.NET Exceptions

> The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

4.2.7 Range Auto Enabled

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Boolean (C/COM)	R/W	Channel	None	Configure Range Auto Enabled
Vi Boolean (.NET)	R/W	Channel	None	Configure Range

.NET Property Name

    Channels[].Range.Auto

COM Property Name

    Channels.Item().Range.AutoEnabled

C Constant Name

    IVIPWRMETER_ATTR_RANGE_AUTO_ENABLED

Description

If this attribute is set to True, the instrument automatically determines the best range for the measurement That is, the instrument selects values for both the Range Lower and Range Upper attributes.

If this attribute is set to False, the user specifies the lower and upper limits of the measurement range by explicitly setting the Range Lower and Range Upper attributes. Refer to Section 6, *IviPwrMeterManualRange Extension Group* for more information.

If the driver implements the IviPwrMeterManualRange extension group, the actual range the power meter is currently using can be determined from the Range Lower and Range Upper attributes. If the driver does not implement the IviPwrMeterManualRange extension group, the driver shall return a Value Not Supported error (C/COM) or throw a Value Not Supported Exception (.NET) if the calling program attempts to set Range Auto Enabled to False.

For C and COM, if either the Range Lower or Range Upper attribute is set, the state of auto-range is indeterminate. For .NET, if either the Range Lower or Range Upper attribute is set, auto-range is disabled.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

Compliance Notes

1. If a specific driver implements the False value for this attribute, it shall also implement the IviPwrMeterManualRange extension capability group.

4.2.8 Units

Data Type	Access	Applies To	Coercion	High Level Functions
ViInt32	R/W	N/A	None	Configure Units

.NET Property Name

      Channels.Units

.NET Enumeration Name

      Ivi.PwrMeter.UnitsEnum

COM Property Name

      Channels.Units

COM Enumeration Name

      IviPwrMeterUnitsEnum

C Constant Name

      IVIPWRMETER_ATTR_UNITS

Description

Specifies the unit to which the RF power is converted after measurement.

The actual RF power of the signal on a channel is always measured in Watts. The value of this attribute is used to determine the units in which the Range Upper and Range Lower attributes are specified. The unit of the measurement result returned by the Read and Fetch functions also depends on the value of this attribute.

Defined Values

Name	Description		
		Language	*Identifier*
dBm	Sets the units to dBm.		
		.NET	Units.dBm
		C	IVIPWRMETER_VAL_DBM
		COM	IviPwrMeterUnitsdBm
dBmV	Sets the units to dB millivolts.		
		.NET	Units.dBmV
		C	IVIPWRMETER_VAL_DBMV
		COM	IviPwrMeterUnitsdBmV
dBuV	Sets the units to dB microvolts.		
		.NET	Units.dBuV
		C	IVIPWRMETER_VAL_DBUV

	COM	IviPwrMeterUnitsdBuV
Watts	Sets the units to Watts.
	.NET	Units.Watts
	C	IVIPWRMETER_VAL_ WATTS
	COM	IviPwrMeterUnitsWatts

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

Compliance Notes

1. If an IVI-C specific driver defines additional values for this attribute, the actual values shall be greater than or equal to `IVIPWRMETER_VAL_UNITS_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 `IVIPWRMETER_VAL_UNITS_CLASS_EXT_BASE` and less than `IVIPWRMETER_VAL_UNITS_SPECIFIC_EXT_BASE`.

3. When an IVI-COM specific driver implements this attribute with additional elements in its instrument specific interfaces, it is recommended that the actual values of the additional elements be greater than or equal to Units Specific Extension Base.

   See Section 16, Attribute Value Definitions, for the definitions of Units Specific Exention Base, `IVIPWRMETER_VAL_UNITS_SPECIFIC_EXT_BASE` and `IVIPWRMETER_VAL_UNITS_CLASS_EXT_BASE`.

4.3 IviPwrMeterBase Functions

The IviPwrMeterBase capability group defines the following functions:

- Abort

- Configure Averaging Auto Enabled (IVI-C Only)

- Configure Correction Frequency (IVI-C Only)

- Configure Measurement

- Configure Offset (IVI-C Only)

- Configure Range Auto Enabled (IVI-C Only)

- Configure Units (IVI-C Only)

- Fetch

- Get Channel Name (IVI-C Only)

- Initiate

- Is Measurement Complete (IVI-C & IVI-COM Only)

- Query Result Range Type

- Read

This section describes the behavior and requirements of each function.

4.3.1 Abort

Description

This function aborts all previously initiated measurements and returns the power meter to the *Idle* state.

This function does not check the instrument status. Typically, the end-user calls this function only in a sequence of calls to other low-level driver functions. The sequence performs one operation. The end-user uses the low-level functions to optimize one or more aspects of interaction with the instrument. To check the instrument status, call the Error Query function at the conclusion of the sequence.

.NET Method Prototype

```
void Measurement.Abort ();
```

COM Method Prototype

```
HRESULT Measurement.Abort ();
```

C Prototype

```
ViStatus IviPwrMeter_Abort (ViSession Vi);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle	ViSession

Return Values (C/COM)

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

.NET Exceptions

The IVI-3.2: *Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

4.3.2 Configure Averaging Auto Enabled (IVI-C Only)

Description

This function enables or disables the auto-averaging mode.

.NET Method Prototype

```
N/A
```

(Use the `Channels[].Averaging.AutoEnabled` property)

COM Method Prototype

```
N/A
```

(Use the `Channels.Item().Averaging.AutoEnabled` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureAveragingAutoEnabled (ViSession Vi,
                                          ViConstString Channel,
                                          ViBoolean AveragingAutoEnabled);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name of the channel to be configured.	ViConstString
AveragingAutoEnabled	The auto-averaging mode. Pass True to turn auto-averaging on. Pass False to turn auto-averaging off. The value of this parameter is used to set the Auto Averaging Enabled attribute. See the attribute description for more information.	ViBoolean

Return Values (C)

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

4.3.3 Configure Correction Frequency (IVI-C Only)

Description

This function specifies the frequency of the input signal in Hertz. The instrument uses this value to determine the appropriate correction factor for the sensor.

.NET Method Prototype

```
N/A
```

(Use the `Channels[].CorrectionFrequency` property)

COM Method Prototype

```
N/A
```

(Use the `Channels.Item().CorrectionFrequency` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureCorrectionFrequency (ViSession Vi,
                                          ViConstString Channel,
                                          ViReal64 CorrectionFrequency);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name of the channel to configure.	ViConstString
CorrectionFrequency	The frequency of the input signal. The value of this parameter is used to set the Correction Frequency attribute. See the attribute description for more information.	ViReal64

Return Values (C)

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

4.3.4 Configure Measurement

Description

This function configures the instrument to take single or dual channel measurements.

To configure the power meter to take single channel measurements, pass None as the value of the Operator parameter. To configure the power meter to take simultaneous dual channel measurements, pass one of the defined math operators as the value of the Operator parameter.

If the user passes None as the value of the Operator parameter, this function enables the channel specified by the Operand1 parameter and disables all other channels. The value of the Operand2 parameter is ignored. The result returned by the Fetch or Read functions is the measurement taken at the channel specified by the Operand1 parameter.

If the user does not pass None as the value of the Operator parameter, this function enables the channels specified by the Operand1 and Operand2 parameters and disables all other channels. The result returned by the Fetch or Read functions is the result of the math operation specified by the Operator parameter applied to the measurements on the channels specified by the Operand1 and Operand2 parameters.

The driver always measures the power on both channels in Watts. For single channel measurements, the result is converted to the same unit as the value of the Units attribute. For dual channel measurements, the math operation is performed on the measured values in Watts and the result is converted to the appropriate unit depending on the value of the Units attribute and the value of the Operator parameter as shown in the following table.

Table 4-1. Operator parameter and Result Units

Units	Operator	Result Units
dBm	Difference	dBm
	Quotient	dB
	Sum	dBm
dBmV	Difference	dBmV
	Quotient	dB
	Sum	dBmV
dBuV	Difference	dBuV
	Quotient	dB
	Sum	dBuV
Watts	Difference	Watts
	Quotient	No Unit
	Sum	Watts

.NET Method Prototype

```
void Measurement.Configure (Ivi.PwrMeter.MeasurementOperator measurementOperator,
                            String operand1,
                            String operand2);
```

COM Method Prototype

```
HRESULT Measurement.Configure ([in] IviPwrMeterMeasurementOperatorEnum Operator,
                               [in] BSTR Operand1,
                               [in] BSTR Operand2);
```

C Prototype

```
ViStatus IviPwrMeter_ConfigureMeasurement (ViSession Vi,
                                           ViInt32 Operator
                                           ViConstString Operand1,
                                           ViConstString Operand2);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Operator C/COM measurementOperator .NET	The math function applied to the operands.	ViInt32
Operand1	The name of the channel from which the value for the first operand of the math function is measured.	ViConstString
Operand2	The name of the channel from which the value for the second operand of the math function is measured.	ViConstString

Defined Values for the Operator Parameter

Name	*Description*	
	Language	*Identifier*
Difference	Operand1 - Operand2	
	.NET	Ivi.PwrMeter.MeasurementOperator.Difference
	C	IVIPWRMETER_VAL_DIFFERENCE
	COM	IviPwrMeterMeasurementOperatorDifference
Sum	Operand1 + Operand2	
	.NET	Ivi.PwrMeter.MeasurementOperator.Sum
	C	IVIPWRMETER_VAL_SUM
	COM	IviPwrMeterMeasurementOperatorSum
Quotient	Operand1 / Operand2	
	.NET	Ivi.PwrMeter.MeasurementOperator.Quotient
	C	IVIPWRMETER_VAL_QUOTIENT
	COM	IviPwrMeterMeasurementOperatorQuotient
None	Operand1	
	.NET	Ivi.PwrMeter.MeasurementOperator.None
	C	IVIPWRMETER_VAL_NONE
	COM	IviPwrMeterMeasurementOperatorNone

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

Compliance Notes

1. If an IVI-C specific driver defines additional values for the Operator parameter, the magnitude of the actual values shall be greater than or equal to IVIPWRMETER_VAL_OPERATOR_SPECIFIC_EXT_BASE.

2. If an IVI-C class driver defines additional values for the Operator parameter, the magnitude of the actual values shall be greater than or equal to IVIPWRMETER_VAL_OPERATOR_CLASS_EXT_BASE and less than IVIPWRMETER_VAL_OPERATOR_SPECIFIC_EXT_BASE.

3. When an IVI-COM specific driver implements the Operator parameter with additional elements in its instrument specific interfaces, it is recommended that the actual values of the additional elements be greater than or equal to Operator Specific Extension Base.

   See Section 17, Function Parameter Value Definitions, for the definitions of Operator Specific Extension Base, IVIPWRMETER_VAL_OPERATOR_SPECIFIC_EXT_BASE and IVIPWRMETER_VAL_OPERATOR_CLASS_EXT_BASE.

4.3.5 Configure Offset (IVI-C Only)

Description

This function specifies the offset to be added to the measured value in units of dB.

.NET Method Prototype

```
N/A
```

(Use the `Channels[].Offset` property)

COM Method Prototype

```
N/A
```

(Use the `Channels.Item().Offset` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureOffset (ViSession Vi,
                                      ViConstString Channel,
                                      ViReal64 Offset);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name of the channel for which to set the Offset.	ViConstString
Offset	The offset for the measured value. The value of this parameter is used to set the Offset attribute. See the attribute description for more information.	ViReal64

Return Values (C)

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

4.3.6 Configure Range Auto Enabled (IVI-C Only)

Description

This function configures the auto range mode for a given channel.

.NET Method Prototype

```
N/A
```

(Use the `Channels[].Range.AutoEnabled` property)

COM Method Prototype

```
N/A
```

(Use the `Channels.Item().Range.AutoEnabled` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureRangeAutoEnabled (ViSession Vi,
                                                ViConstString Channel,
                                                ViBoolean RangeAutoEnabled);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name of the channel to configure.	ViConstString
RangeAutoEnabled	The auto range mode. Pass True to turn auto ranging on. Pass False to turn auto ranging off. The value of this parameter is used to set the Range Auto Enabled attribute. See the attribute description for more information.	ViBoolean

Return Values (C)

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

4.3.7 Configure Units (IVI-C Only)

Description

This function configures the unit to which the RF power is converted after measurement.

.NET Method Prototype

```
N/A
```

(Use the `Channels.Units` property)

COM Method Prototype

```
N/A
```

(Use the `Channels.Units` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureUnits (ViSession Vi,
                                     ViInt32 Units);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Units	The unit of the measurement result. The value of this parameter is used to set the Units attribute. See the attribute description for more information.	ViInt32

Return Values (C)

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

4.3.8 Fetch

Description

This function returns the result from a previously initiated single or dual channel measurement. Call the Initiate function to initiate a measurement before calling this function.

After this function executes, the value of the `Result` parameter depends on the math operation that the user specifies in the Configure Measurement function.

For single channel measurements, the `Result` parameter contains an actual reading on the channel specified by the Configure Measurement function. The unit of the result is the same as the value of the Units attribute.

For dual channel measurements, the `Result` parameter contains the result of the math operation applied to the channels specified in the Configure Measurement function. The unit of the result depends on the value of the Units attribute and the value of the `Operator` parameter passed to the Configure Measurement function. Refer to Table 4-1 in Section 4.3.4, *Configure Measurement* for more details.

The behavior is different for IVI-C/IVI-COM and IVI.NET as follows:

> IVI-C & IVI-COM: If an out of range condition occurs on one or more enabled channels, the result is a value indicating that an out of range condition occurred. In such a case, the `Result` parameter contains an IEEE 754 defined -Inf (Negative Infinity) or +Inf (Positive Infinity) value and the function returns the Over Range or Under Range warning. Test if the measurement value is out of range with the Query Result Range Type function.

> IVI.NET: The reading is the return value of the method. If an overrange condition occurs, the result will be Double.PositiveInfinity or Double.NegativeInfinity. No exception shall be thrown. The end user may test the measurement value for overrange with Double.IsInfinity(), Double.IsPositiveInfinity() or Double.IsNegativeInfinity() methods.

This function does not check the instrument status. Typically, the end-user calls this function only in a sequence of calls to other low-level driver functions. The sequence performs one operation. The end-user uses the low-level functions to optimize one or more aspects of interaction with the instrument. To check the instrument status, call the Error Query function at the conclusion of the sequence.

.NET Method Prototype

```
Double Measurement.Fetch ();
Double Measurement.Fetch (out Boolean sampleOutOfRange);
```

COM Method Prototype

```
HRESULT Measurement.Fetch ([out, retval] DOUBLE* Result);
```

C Prototype

```
ViStatus IviPwrMeter_Fetch (ViSession Vi,
                            ViReal64 *Result);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle	ViSession

Outputs	Description	Datatype
sampleOutOfRange	True if any of the values are under or over range, otherwise false.	ViBoolean
Result (C/COM)	Measured value	ViReal64*
Return value (.NET)	Measured value	ViReal64

Return Values (C/COM)

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return. The table below specifies additional class-defined status codes for this function.

Completion Code	Description
Over Range	Measurement is over range.
Under Range	Measurement is under range.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

.NET does not have over and under range warnings. Instead, the method returns Double.PositiveInfinity or Double.NegativeInfinity respectively.

4.3.9 Get Channel Name (IVI-C Only)

Description

This function returns the physical channel identifier that corresponds to the one-based index that the user specifies. If the driver defines a qualified channel name, this property returns the qualified name. If the value that the user passes for the `ChannelIndex` parameter is less than one or greater than the value of the Channel Count attribute, the function returns an empty string in the `ChannelName` parameter and returns an error.

.NET Method Prototype

```
N/A
```

(Use the `Channels[].Name` property instead)

COM Method Prototype

```
N/A
```

(Use the `Channels.Name()` method instead)

C Prototype

```
ViStatus IviPwrMeter_GetChannelName (ViSession Vi,
                                     ViInt32 ChannelIndex,
                                     ViInt32 ChannelNameBufferSize,
                                     ViChar ChannelName[]);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ChannelIndex	A one-based index that defines which name to return.	ViInt32
ChannelName BufferSize	The number of bytes in the ViChar array that the user specifies for the `ChannelName` parameter.	ViInt32

Outputs	Description	Base Type
ChannelName	The buffer into which the function returns the channel name that corresponds to the index the user specifies. The caller may pass `VI_NULL` for this parameter if the `ChannelNameBufferSize` parameter is 0.	ViChar[]

Return Values (C)

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

4.3.10 Initiate

Description

This function initiates a measurement on all enabled channels. When this function executes, the power meter leaves the *Idle* state and takes a measurement on all enabled channels. Use the Fetch or Fetch Channel function to obtain the result of the measurements.

This function does not check the instrument status. Typically, the end-user calls this function only in a sequence of calls to other low-level driver functions. The sequence performs one operation. The end-user uses the low-level functions to optimize one or more aspects of interaction with the instrument. To check the instrument status, call the Error Query function at the conclusion of the sequence.

.NET Method Prototype

```
void Measurement.Initiate();
```

COM Method Prototype

```
HRESULT Measurement.Initiate();
```

C Prototype

```
ViStatus IviPwrMeter_Initiate (ViSession Vi);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle	ViSession

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

4.3.11 Is Measurement Complete

Description

This function queries the instrument to determine the status of the measurement initiated by the Initiate function. This function returns the Measurement Complete value only when measurements are complete on all enabled channels.

If some measurements are still in progress on one or more channels, the driver returns the Measurement In Progress value. If the driver cannot query the instrument to determine its state, the driver returns the Measurement Status Unknown value.

This function does not check the instrument status. Typically, the end-user calls this function only in a sequence of calls to other low-level driver functions. The sequence performs one operation. The end-user uses the low-level functions to optimize one or more aspects of interaction with the instrument. To check the instrument status, call the Error Query function at the conclusion of the sequence.

.NET Method Prototype

```
Ivi.PwrMeter.OperationState Measurement.GetMeasurementComplete ();
```

COM Method Prototype

```
HRESULT Measurement.IsMeasurementComplete ([out, retval]
                                    IviPwrMeterMeasurementStatusEnum* Status);
```

C Prototype

```
ViStatus IviPwrMeter_IsMeasurementComplete (ViSession Vi,
                                    ViInt32 *Status);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle	ViSession

Outputs	Description	Datatype
Status (C, COM) Return value (.NET)	Returns the status of the measurement.	ViInt32*

Defined Values for Status Parameter (C, COM) or Return Value (.NET)

Name	Description	
	Language	**Identifier**
Measurement Complete	The power meter has completed the measurement on all enabled channels.	
	.NET	Ivi.PwrMeter.OperationState.Complete
	C	IVIPWRMETER_VAL_MEAS_COMPLETE
	COM	IviPwrMeterMeasurementStatusComplete
Measurement In Progress	The power meter is still taking a measurement on one or more enabled channels.	

	.NET	Ivi.PwrMeter.OperationState.InProgress
	C	IVIPWRMETER_VAL_MEAS_IN_PROGRESS
	COM	IviPwrMeterMeasurementStatusInProgress
Measurement Status Unknown	The power meter cannot determine the status of the measurement.	
	.NET	Ivi.PwrMeter.OperationState.Unknown
	C	IVIPWRMETER_VAL_MEAS_STATUS_UNKNOWN
	COM	IviPwrMeterMeasurementStatusUnknown

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

4.3.12 Query Result Range Type

Description

This function takes a measurement value that is returned from one of the Fetch, Fetch Channel, Read, or Read Channel functions and determines if the value is a valid measurement value or a value indicating that an out-of-range condition occurred.

.NET Method Prototype

```
N/A
```
(See the documentation for the Read and Fetch methods for information about determining whether an out of range condition exists.)

COM Method Prototype

```
HRESULT Measurement.QueryResultRangeType([in] DOUBLE MeasurementValue,
                          [out, retval] IviPwrMeterResultRangeEnum* RangeType);
```

C Prototype

```
ViStatus IviPwrMeter_QueryResultRangeType (ViSession Vi,
                                   ViReal64 MeasurementValue,
                                   ViInt32 *RangeType);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle	ViSession
Measurement Value	The measurement value returned by one of the Fetch or Read functions.	ViReal64

Outputs	Description	Datatype
RangeType	Returns whether the Measurement Value is a valid measurement or a value indicating that the power meter encountered an out-of-range condition.	ViInt32*

Defined Values for the RangeType Parameter

Name	Description	
	Language	*Identifier*
In Range	The measurement is within the current range limits.	
	C	IVIPWRMETER_VAL_IN_RANGE
	COM	IviPwrMeterResultRangeInRange
Under Range	The measurement is below the current lower range limit.	
	C	IVIPWRMETER_VAL_UNDER_RANGE
	COM	IviPwrMeterResultRangeUnderRange
Over Range	The measurement is above the current upper range limit.	
	C	IVIPWRMETER_VAL_OVER_RANGE

	COM	IviPwrMeterResultRangeOverRange

Return Values (C/COM)

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

4.3.13 Read

Description

This function initiates a measurement, waits until the power meter has returned to the *Idle* state, and returns the result of the measurement.

After this function executes, the value of the `Result` parameter depends on the math operation that the user specifies in the Configure Measurement function.

For single channel measurements, the `Result` parameter contains an actual reading on the channel specified by the Configure Measurement function. The unit of the result is the same as the value of the Units attribute.

For dual channel measurements, the `Result` parameter contains the result of the math operation applied to the channels specified in the Configure Measurement function. The unit of the result depends on the value of the Units attribute and the math function. Refer to Table 4-1 in Section 4.3.4, *Configure Measurement* for more details.

The behavior is different for IVI-C/IVI-COM and IVI.NET as follows:

> IVI-C & IVI-COM: If an out of range condition occurs on one or more enabled channels, the result is a value indicating that an out of range condition occurred. In such a case, the `Result` parameter contains an IEEE 754 defined -Inf (Negative Infinity) or +Inf (Positive Infinity) value and the function returns the Over Range or Under Range warning. Test if the measurement value is out of range with the Query Result Range Type function.

> IVI.NET: The reading is the return value of the method. If an overrange condition occurs, the result will be Double.PositiveInfinity or Double.NegativeInfinity. No exception shall be thrown. The end user may test the measurement value for overrange with Double.IsInfinity(), Double.IsPositiveInfinity() or Double.IsNegativeInfinity() methods.

> For .NET a maximumTime of PrecisionTimeSpan.Zero indicates that the measurement should only be returned if it is already available. A maximumTime of PrecisionTimeSpan.MaxValue indicates that the measurement should wait until a measurement is available, with no timeout.

.NET Method Prototype

```
Double Measurement.Read (PrecisionTimeSpan maximumTime);
Double Measurement.Read (PrecisionTimeSpan maximumTime, out Boolean
                         sampleOutOfRange);
```

COM Method Prototype

```
HRESULT Measurement.Read ([in] LONG MaxTimeMilliseconds,
                          [out, retval] DOUBLE* Result);
```

C Prototype

```
ViStatus IviPwrMeter_Read (ViSession Vi,
                           ViInt32 MaxTimeMilliseconds,
                           ViReal64 *Result);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle	ViSession

MaxTimeMilliseconds (C, COM)	Specifies the maximum time in milliseconds to wait for a return value.	ViInt32
maximumTime (.NET)	Specifies the maximum time to wait for a return value.	PrecisionTimeSpan

Outputs	Description	Datatype
sampleOutOfRange	True if any of the values are under or over range, otherwise false.	ViBoolean
Result (C, COM)	Measured value.	ViReal64*
Return value (.NET)	Measured value.	ViReal64

Defined Values for the MaxTimeMilliseconds Parameter (C/COM)

Name	Description		
		Language	*Identifier*
Immediate Timeout	The function returns immediately. If no valid measurement value exists, the function returns an error.		
		C	IVIPWRMETER_VAL_MAX_TIME_IMMEDIATE
		COM	IviPwrMeterTimeOutImmediate
Infinite Timeout	The function waits indefinitely for the measurement to complete.		
		C	IVIPWRMETER_VAL_MAX_TIME_INFINITE
		COM	IviPwrMeterTimeOutInfinite

Defined Values for the MaximumTime Parameter (.NET)

Name	Description		
		Language	*Identifier*
Zero	The function returns immediately. If no valid measurement value exists, the function returns an error.		
		.NET	`PrecisionTimeSpan.Zero`
Max Value	The function waits indefinitely for the measurement to complete.		
		.NET	`PrecisionTimeSpan.MaxValue`

Return Values (C/COM)

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return. The table below specifies additional class-defined status codes for this function.

Completion Code	Description
Over Range	Measurement is over range.
Under Range	Measurement is under range.
Max Time Exceeded	Maximum timeout exceeded before operation could complete.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

.NET does not have over and under range warnings. Instead, the method returns Double.PositiveInfinity or Double.NegativeInfinity respectively.

Note that the .NET MaxTimeExceededException is defined in *IVI-3.2: Inherent Capabilities Specification*.

Compliance Notes

1. A specific driver is not required to implement the Immediate Timeout or the Infinite Timeout defined values for the `MaxTimeMilliseconds` parameter to be compliant with the IviPwrMeterBase capability group.

4.4 IviPwrMeterBase Behavior Model

The following state diagram shows relationships between the IviPwrMeterBase capability group and power meter behavior.
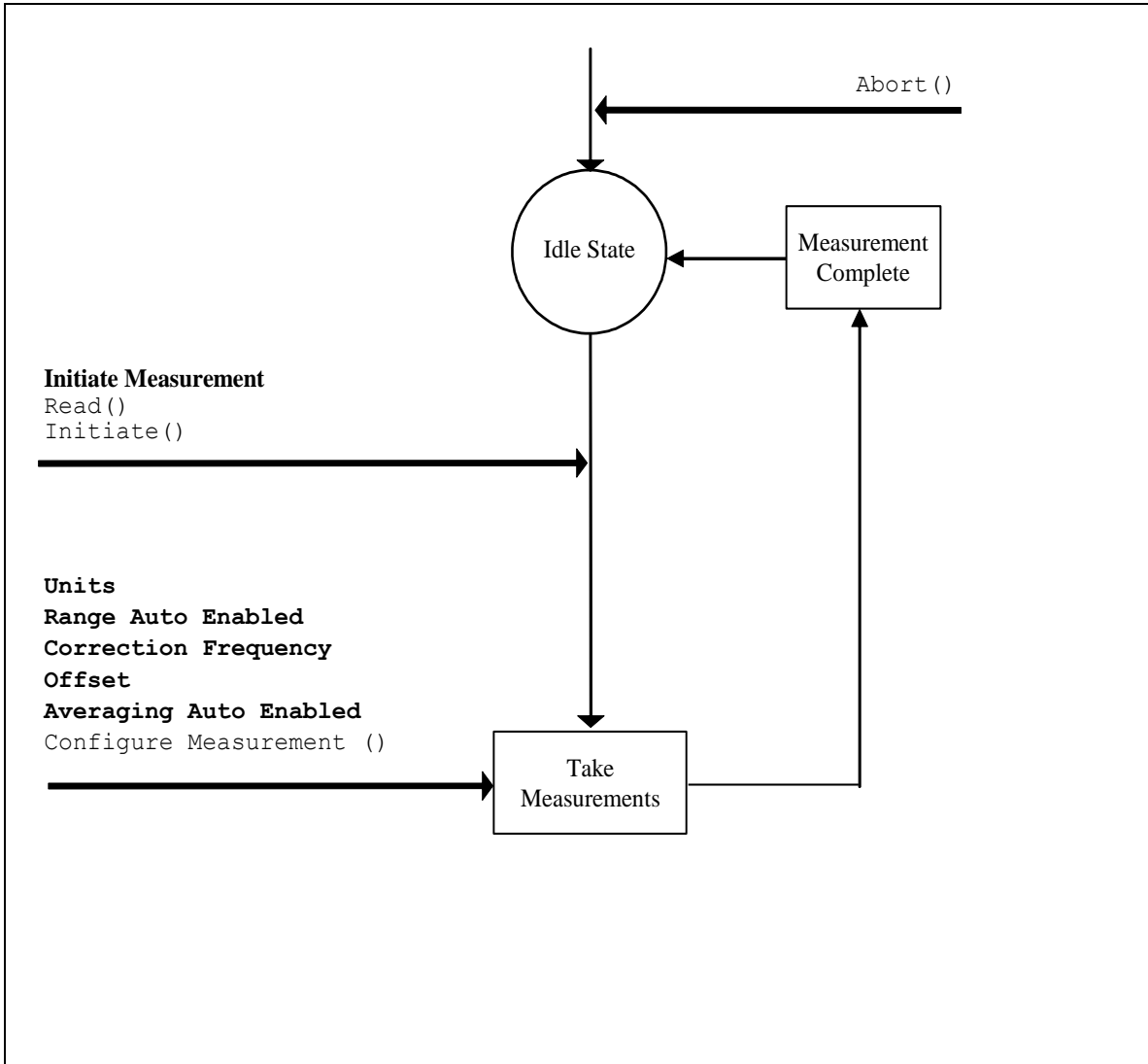
Figure 1. IviPwrMeterBase Behavior Model

The main state in the IviPwrMeter Class is the *Idle* state. The power meter enters the *Idle* state as the result of being "powered-on", successfully completing a measurement, or by being aborted from a previous measurement by the user with the Abort function. Typically, the user configures the power meter while it is in the *Idle* state. IviPwrMeter attributes can be configured individually with the Set Attribute function or with the high-level configuration functions defined in the IviPwrMeterBase capability group.

The Read and Initiate functions cause the power meter to leave the *Idle* state and take a measurement on all enabled channels. The Read function does not return until the measurement process is complete and the power meter has returned to the *Idle* state. The Initiate function returns as soon as the power meter leaves the *Idle* state. The Fetch function is used to retrieve measurements that were initiated by the Initiate function.

As soon as the power meter leaves the *Idle* state, it immediately takes a measurement on all enabled channels.You enable channels by calling the Configure Measurement function before initiating the measurement. The power meter takes a measurement on the channels specified in the Configure Measurement function and performs the specified math operation on the results.

After all measurements have been taken, the power meter (if it is capable of doing so) generates the Measurement Complete signal and returns to the *Idle* state. The IviPwrMeterBase capability group does not require that a power meter be able to generate a Measurement Complete signal. The IviPwrMeterBase capability group does not define how a Measurement Complete signal is configured. The Measurement Complete signal is presented in the IviPwrMeter behavior model diagram to define when the signal is generated as most power meters generate this signal but may not be able to configure it.

5. IviPwrMeterChannelAcquisition Extension Group

5.1 Overview

The IviPwrMeterChannelAcquisition extension capability group supports power meters that can perform simultaneous measurements on two or more channels and fetch the measurement from each specified channel. The IviPwrMeterChannelAcquisition extension capability also includes attributes for enabling a channel for measurement and functions for acquiring measurements on a specified channel.

5.2 IviPwrMeterChannelAcquisition Attributes

The IviPwrMeterChannelAcquisition capability group defines the following attributes:

- Channel Enabled

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions*.

5.2.1 Channel Enabled

Data Type	Access	Applies To	Coercion	High Level Functions
ViBoolean	R/W	Channel	None	Configure Channel Enabled

.NET Property Name

       Channels[].Enabled

COM Property Name

       Channels.Item().Enabled

C Constant Name

       IVIPWRMETER_ATTR_CHANNEL_ENABLED

Description

If set to True, enables the specified channel. If set to False, disables the specified channel. The power meter will take a measurement on a channel only if that channel is enabled.

Channels are also enabled when the user calls the Configure Measurement function. See the function description for more information.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

5.3 IviPwrMeterChannelAcquisition Functions

The IviPwrMeterChannelAcquisition extension capability group defines the following functions:

- Configure Channel Enabled (IVI-C Only)

- Fetch Channel

- Read Channel

This section describes the behavior and requirements of each function.

5.3.1 Configure Channel Enabled (IVI-C Only)

Description

This function enables or disables a specified channel for measurement.

.NET Method Prototype

```
N/A
```

(Use the `Channels[].Enabled` property)

COM Method Prototype

```
N/A
```

(Use the `Channels.Item().Enabled` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureChannelEnabled (ViSession Vi,
                                               ViConstString Channel,
                                               ViBoolean ChannelEnabled);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name of the channel.	ViConstString
Channel Enabled	Pass True to enable the channel. Pass False to disable the channel. The value of this parameter is used to set the Channel Enabled attribute. See the attribute description for more information.	ViBoolean

Return Values (C)

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

5.3.2 Fetch Channel

Description

This function returns the result from a previously initiated measurement on a specified channel. Call the Initiate function to initiate a measurement before calling this function.

After this function executes, the Result parameter contains an actual reading on the channel specified by the Channel parameter. If the specified channel is not enabled for measurement, this function returns the Channel Not Enabled error. The result is in the same unit as the value of the Units attribute.

The behavior is different for IVI-C/IVI-COM and IVI.NET as follows:

IVI-C & IVI-COM: If an out of range condition occurs, the result is a value indicating that an out of range condition occurred. In such a case, the Result parameter contains an IEEE 754 defined -Inf (Negative Infinity) or +Inf (Positive Infinity) value and the function returns the Over Range or Under Range warning. Test if the measurement value is out of range with the Query Result Range Type function.

IVI.NET: The reading is the return value of the method. If an overrange condition occurs, the result will be Double.PositiveInfinity or Double.NegativeInfinity. No exception shall be thrown. The end user may test the measurement value for overrange with Double.IsInfinity(), Double.IsPositiveInfinity() or Double.IsNegativeInfinity() methods.

This function does not check the instrument status. Typically, the end-user calls this function only in a sequence of calls to other low-level driver functions. The sequence performs one operation. The end-user uses the low-level functions to optimize one or more aspects of interaction with the instrument. To check the instrument status, call the Error Query function at the conclusion of the sequence.

.NET Method Prototype

```
Double Measurement.FetchChannel (String channelName);
Double Measurement.FetchChannel (String channelName,
                                 out Boolean sampleOutOfRange);
```

COM Method Prototype

```
HRESULT Measurement.FetchChannel ([in] BSTR ChannelName,
                                  [out, retval] DOUBLE* Result);
```

C Prototype

```
ViStatus IviPwrMeter_FetchChannel (ViSession Vi,
                                   ViConstString Channel,
                                   ViReal64 *Result);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle	ViSession
Channel	The name of the channel from which to fetch the measurement.	ViConstString

Outputs	Description	Datatype

sampleOutOfRange	True if any of the values are under or over range, otherwise false.	ViBoolean
Result (C/COM)	Measured value	ViReal64*
Return value (.NET)	Measured value	ViReal64

Return Values (C/COM)

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return. The table below specifies additional class-defined status codes for this function.

Completion Code	Description
Over Range	Measurement is over range.
Under Range	Measurement is under range.
Channel Not Enabled	The specified channel is not enabled for measurement.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

The table below specifies additional class-defined exceptions for this method.

Exception Class	Description
ChannelNotEnabledException	The specified channel is not enabled for measurement.

.NET does not have over and under range warnings. Instead, the method returns Double.PositiveInfinity or Double.NegativeInfinity respectively.

5.3.3 Read Channel

Description

This function initiates a measurement, waits until the power meter has returned to the *Idle* state, and returns the result of the measurement on the specified channel.

After this function executes, the `Result` parameter contains an actual reading on the channel specified by the `Channel` parameter. If the specified channel is not enabled for measurement, this function returns the Channel Not Enabled error. The result is in the same unit as the value of the Units attribute.

The behavior is different for IVI-C/IVI-COM and IVI.NET as follows:

IVI-C & IVI-COM: If an out of range condition occurs, the result is a value indicating that an out of range condition occurred. In such a case, the `Result` parameter contains an IEEE 754 defined -Inf (Negative Infinity) or +Inf (Positive Infinity) value and the function returns the Over Range or Under Range warning. Test if the measurement value is out of range with the Query Result Range Type function.

IVI.NET: The reading is the return value of the method. If an overrange condition occurs, the result will be Double.PositiveInfinity or Double.NegativeInfinity. No exception shall be thrown. The end user may test the measurement value for overrange with Double.IsInfinity(), Double.IsPositiveInfinity() or Double.IsNegativeInfinity() methods.

For .NET a maximumTime of PrecisionTimeSpan.Zero indicates that the measurement should only be returned if it is already available. A maximumTime of PrecisionTimeSpan.MaxValue indicates that the measurement should wait until a measurement is available, with no timeout.

.NET Method Prototype

```
Double Measurement.ReadChannel(String channelName,
                               PrecisionTimeSpan maximumTime);
Double Measurement.ReadChannel(String channelName,
                               PrecisionTimeSpan maximumTime,
                               out Boolean sampleOutOfRange);
```

COM Method Prototype

```
HRESULT Measurement.ReadChannel([in] BSTR ChannelName,
                                [in] LONG MaxTimeMilliseconds,
                                [out, retval] DOUBLE* Result);
```

C Prototype

```
ViStatus IviPwrMeter_ReadChannel (ViSession Vi,
                                  ViConstString Channel,
                                  ViInt32 MaxTimeMilliseconds,
                                  ViReal64 *Result);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle	ViSession
Channel	The name of the channel from which to take the measurement.	ViConstString

MaxTimeMilliseconds (C/COM)	Specifies the maximum time in milliseconds to wait for a return value.	ViInt32
maximumTime (.NET)	Specifies the maximum time to wait for a return value	PrecisionTimeSpan

Outputs	Description	Datatype
sampleOutOfRange	True if any of the values are under or over range, otherwise false.	ViBoolean
Result (C/COM)	Measured value.	ViReal64*
Return value (.NET)	Measured value.	ViReal64

Defined Values for the MaxTimeMilliseconds Parameter (C/COM)

Name	Description		
		Language	*Identifier*
Immediate Timeout	The function returns immediately. If no valid measurement value exists, the function returns an error.		
		C	IVIPWRMETER_VAL_MAX_TIME_IMMEDIATE
		COM	IviPwrMeterTimeOutImmediate
Infinite Timeout	The function waits indefinitely for the measurement to complete.		
		C	IVIPWRMETER_VAL_MAX_TIME_INFINITE
		COM	IviPwrMeterTimeOutInfinite

Defined Values for the MaximumTime Parameter (.NET)

Name	Description	
	Language	Identifier
Zero	The function returns immediately. If no valid measurement value exists, the function returns an error.	
	.NET	PrecisionTimeSpan.Zero
Max Value	The function waits indefinitely for the measurement to complete.	
	.NET	PrecisionTimeSpan.MaxValue

Return Values (C/COM)

The *IVI-3.2: Inherent Capabilities Specification* defines general status codes that this function can return. The table below specifies additional class-defined status codes for this function.

Completion Code	Description
Over Range	Measurement is over range.
Under Range	Measurement is under range.
Channel Not Enabled	The specified channel is not enabled for measurement.
Max Time Exceeded	Maximum timeout exceeded before operation could complete.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

The table below specifies additional class-defined exceptions for this method.

Exception Class	Description
ChannelNotEnabledException	The specified channel is not enabled for measurement.

.NET does not have over and under range warnings. Instead, the method returns Double.PositiveInfinity or Double.NegativeInfinity respectively.

Note that the .NET MaxTimeExceededException is defined in *IVI-3.2: Inherent Capabilities Specification*.

Compliance Notes

1. A specific driver is not required to implement the Immediate Timeout or the Infinite Timeout defined values for the MaxTimeMilliseconds parameter to be compliant with the IviPwrMeterChannelAcquisition extension group.

5.4 IviPwrMeterChannelAcquisition Behavior Model

The IviPwrMeterChannelAcquisition behavior model leverages the behavior model of the IviPwrMeterBase capability group.

6. IviPwrMeterManualRange Extension Group

6.1 Overview

The IviPwrMeterManualRange extension capability group supports power meters that can manually specify the upper and lower limits of the measurement range. The IviPwrMeterManualRange extension capability also includes functions for configuring the measurement range.

6.2 IviPwrMeterManualRange Attributes

The IviPwrMeterManualRange capability group defines the following attributes:

- Range Lower

- Range Upper

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions.*

6.2.1 Range Lower

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Real 64	R/ W	Channel	Down	Configure Range

.NET Property Name

> Channels[].Range.Lower

COM Property Name

> Channels.Item().Range.Lower

C Constant Name

> IVIPWRMETER_ATTR_RANGE_LOWER

Description

Specifies the lower limit (minimum) of the expected value of the measurement. The specific driver is expected to coerce this value to the appropriate range for the instrument. The value of this attribute is specified in the same units as the value of the Units attribute.

This attribute affects the behavior of the instrument only when the Range Auto Enabled attribute is set to False. For C and COM, if this attribute is set, the state of auto-range is indeterminate. For.NET, if this attribute is set, auto-range is disabled.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

6.2.2 Range Upper

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Real 64	R/ W	Channel	Up	Configure Range

.NET Property Name

    Channels[].Range.Upper

COM Property Name

    Channels.Item().Range.Upper

C Constant Name

    IVIPWRMETER_ATTR_RANGE_UPPER

Description

Specifies the upper limit (maximum) of the expected value of the measurement. The specific driver is expected to coerce this value to the appropriate range for the instrument. The value of this attribute is specified in the same units as the value of the Units attribute.

This attribute affects the behavior of the instrument only when the Range Auto Enabled attribute is set to False. For C and COM, if this attribute is set, the state of auto-range is indeterminate. For.NET, if this attribute is set, auto-range is disabled.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

6.3 IviPwrMeterManualRange Functions

The IviPwrMeterManualRange extension capability group defines the following functions:

- Configure Range

This section describes the behavior and requirements of each function.

6.3.1 Configure Range

Description

This function configures the lower and upper range values for a given channel.

For C and COM, if this function is executed, the state of auto-range is indeterminate.

For.NET, if the function overload that takes two doubles, lower and upper, is executed, the Range Upper and Range Lower attributes are explicitly set and auto-range is disabled. If the function overload that takes a Boolean is executed, and rangeAuto is set to true, auto-range is enabled. If rangeAuto is set to false, auto-range is disabled.

.NET Method Prototype

```
void Channels[].Range.Configure(Double lower,
                                Double upper);

void Channels[].Range.Configure(Boolean rangeAuto);
```

COM Method Prototype

```
HRESULT Channels.Item().Range.Configure([in] DOUBLE Lower,
                                        [in] DOUBLE Upper);
```

C Prototype

```
ViStatus IviPwrMeter_ConfigureRange (ViSession Vi,
                                     ViConstString Channel,
                                     ViReal64 RangeLower,
                                     ViReal64 RangeUpper);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name of the channel to configure.	ViConstString
RangeLower	The lower limit of the expected value of the measurement. The value of this parameter is used to set the Range Lower attribute. See the attribute description for more information.	ViReal64
RangeUpper	The upper limit of the expected value of the measurement. The value of this parameter is used to set the Range Upper attribute. See the attribute description for more information.	ViReal64
rangeAuto (.NET)	Specifies if the power meter should automatically determine the best range for the measurement. The value of this parameter is used to set the Range Auto Enabled attribute. See the attribute description for more information.	Boolean

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

6.4 IviPwrMeterManualRange Behavior Model

The IviPwrMeterManualRange behavior model leverages the behavior model of the IviPwrMeterBase capability group.

6.5 IviPwrMeterManualRange Compliance Notes

1. Specific drivers that implement this extension group shall implement the False value for the Range Auto Enabled attribute.

7. IviPwrMeterTriggerSource Extension Group

7.1 Overview

The IviPwrMeterTriggerSource extension capability group supports power meters that can specify a trigger source and a trigger event on which to trigger a measurement. The IviPwrMeterTriggerSource extension capability also includes functions for configuring the trigger source.

7.2 IviPwrMeterTriggerSource Attributes

The IviPwrMeterTriggerSource capability group defines the following attributes:

- Trigger Source

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions*.

7.2.1 Trigger Source

Data Type	Access	Applies To	Coercion	High Level Functions
ViInt32 (C, COM)	R/W	N/A	None	Configure Trigger Source
ViString (.NET)	R/W	N/A	None	N/A

.NET Property Name

> Trigger.Source

COM Property Name

> Trigger.Source

COM Enumeration Name

> IviPwrMeterTriggerSourceEnum

C Constant Name

> IVIPWRMETER_ATTR_TRIGGER_SOURCE

Description

> Specifies the trigger source the power meter monitors for the trigger event.
>
> When the trigger event occurs on the source specified by this attribute, the power meter leaves the *Wait-For-Trigger* state and takes a measurement on all enabled channels. See Section 7.4, *IviPwrMeterTriggerSource Behavior Model* for more information.
>
> If this attribute is set to the Internal defined value, the power meter uses the channel specified by the Internal Trigger Source Event attribute to monitor the internal trigger event.
>
> In IVI.NET the trigger source is a string. If an IVI driver supports a trigger source and the trigger source is listed in IVI-3.3 *Cross Class Capabilities Specification*, Section 3 then the IVI driver shall accept the standard string for that trigger source. This attribute is case insensitive, but case preserving. That is the setting is case insensitive but when reading it back the programmed case is returned. IVI specific drivers may define new trigger source strings for trigger sources that are not defined by IVI-3.3 *Cross Class Capabilities Specification* if needed.

Defined Values

Name	*Description*		
		Language	*Identifier*
Immediate	The power meter exits the Wait-For-Trigger state immediately after entering. It does not wait for a trigger of any kind.		
		C	IVIPWRMETER_VAL_IMMEDIATE
		COM	IviPwrMeterTriggerSourceImmediate
External	The power meter exits the Wait-For-Trigger state when a trigger occurs on the external trigger input.		
		C	IVIPWRMETER_VAL_EXTERNAL
		COM	IviPwrMeterTriggerSourceExternal

Name	Description		
		Language	*Identifier*
Internal	The power meter exits the Wait-For-Trigger state when an internal trigger event occurs on the measurement signal.		
		C	IVIPWRMETER_VAL_INTERNAL
		COM	IviPwrMeterTriggerSourceInternal
Software	The power meter exits the Wait-For-Trigger state when it receives a software trigger.		
		C	IVIPWRMETER_VAL_SOFTWARE_TRIG
		COM	IviPwrMeterTriggerSourceSoftware
TTL0	The power meter exits the Wait-For-Trigger state when it receives a trigger on TTL0.		
		C	IVIPWRMETER_VAL_TTL0
		COM	IviPwrMeterTriggerSourceTTL0
TTL1	The power meter exits the Wait-For-Trigger state when it receives a trigger on TTL1.		
		C	IVIPWRMETER_VAL_TTL1
		COM	IviPwrMeterTriggerSourceTTL1
TTL2	The power meter exits the Wait-For-Trigger state when it receives a trigger on TTL2.		
		C	IVIPWRMETER_VAL_TTL2
		COM	IviPwrMeterTriggerSourceTTL2
TTL3	The power meter exits the Wait-For-Trigger state when it receives a trigger on TTL3.		
		C	IVIPWRMETER_VAL_TTL3
		COM	IviPwrMeterTriggerSourceTTL3
TTL4	The power meter exits the Wait-For-Trigger state when it receives a trigger on TTL4.		
		C	IVIPWRMETER_VAL_TTL4
		COM	IviPwrMeterTriggerSourceTTL4
TTL5	The power meter exits the Wait-For-Trigger state when it receives a trigger on TTL5.		
		C	IVIPWRMETER_VAL_TTL5
		COM	IviPwrMeterTriggerSourceTTL5
TTL6	The power meter exits the Wait-For-Trigger state when it receives a trigger on TTL6.		
		C	IVIPWRMETER_VAL_TTL6
		COM	IviPwrMeterTriggerSourceTTL6
TTL7	The power meter exits the Wait-For-Trigger state when it receives a trigger on TTL7.		
		C	IVIPWRMETER_VAL_TTL7

Name	Description		
		Language	*Identifier*
		COM	IviPwrMeterTriggerSourceTTL7
ECL0	The power meter exits the Wait-For-Trigger state when it receives a trigger on ECL0.		
		C	IVIPWRMETER_VAL_ECL0
		COM	IviPwrMeterTriggerSourceECL0
ECL1	The power meter exits the Wait-For-Trigger state when it receives a trigger on ECL1.		
		C	IVIPWRMETER_VAL_ECL1
		COM	IviPwrMeterTriggerSourceECL1
PXI STAR	The power meter exits the Wait-For-Trigger state when it receives a trigger on PXI Star trigger bus.		
		C	IVIPWRMETER_VAL_PXI_STAR
		COM	IviPwrMeterTriggerSourcePXIStar
RTSI0	The power meter exits the Wait-For-Trigger state when it receives a trigger on the RTSI0 line.		
		C	IVIPWRMETER_VAL_RTSI_0
		COM	IviPwrMeterTriggerSourceRTSI0
RTSI1	The power meter exits the Wait-For-Trigger state when it receives a trigger on the RTSI1 line.		
		C	IVIPWRMETER_VAL_RTSI_1
		COM	IviPwrMeterTriggerSourceRTSI1
RTSI2	The power meter exits the Wait-For-Trigger state when it receives a trigger on RTSI2 line.		
		C	IVIPWRMETER_VAL_RTSI_2
		COM	IviPwrMeterTriggerSourceRTSI2
RTSI3	The power meter exits the Wait-For-Trigger state when it receives a trigger on the RTSI3 line.		
		C	IVIPWRMETER_VAL_RTSI_3
		COM	IviPwrMeterTriggerSourceRTSI3
RTSI4	The power meter exits the Wait-For-Trigger state when it receives a trigger on the RTSI4 line.		
		C	IVIPWRMETER_VAL_RTSI_4
		COM	IviPwrMeterTriggerSourceRTSI4
RTSI5	The power meter exits the Wait-For-Trigger state when it receives a trigger on the RTSI5 line.		
		C	IVIPWRMETER_VAL_RTSI_5
		COM	IviPwrMeterTriggerSourceRTSI5
RTSI6	The power meter exits the Wait-For-Trigger state when it receives a trigger on the RTSI6 line.		

Name	Description	
	Language	*Identifier*
	C	IVIPWRMETER_VAL_RTSI_6
	COM	IviPwrMeterTriggerSourceRTSI6

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

Compliance Notes

1. All specific drivers shall implement the Immediate defined value for this attribute.

2. If a specific driver implements any of the defined values in the following table, it shall also implement the corresponding extension capability group:

Value	Required Capability Group
Internal	IviPwrMeterInternalTrigger
Software	IviPwrMeterSoftwareTrigger

3. If an IVI-C specific driver defines additional values for this attribute, the actual values shall be greater than or equal to `IVIPWRMETER_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE`.

4. If an IVI-C class driver defines additional values for this attribute, the actual values shall be greater than or equal to `IVIPWRMETER_VAL_TRIGGER_SOURCE_CLASS_EXT_BASE` and less than `IVIPWRMETER_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE`.

5. When an IVI-COM specific driver implements this attribute with additional elements in its instrument specific interfaces, it is recommended that the actual values of the additional elements be greater than or equal to Trigger Source Specific Extension Base.

   See Section 16, Attribute Value Definitions, for the definitions of Trigger Source Specific Extension Base, `IVIPWRMETER_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE` and `IVIPWRMETER_VAL_TRIGGER_SOURCE_CLASS_EXT_BASE`.

7.3 IviPwrMeterTriggerSource Functions

The IviPwrMeterTriggerSource extension capability group defines the following functions:

- Configure Trigger Source (IVI-C Only)

This section describes the behavior and requirements of each function.

7.3.1 Configure Trigger Source (IVI-C Only)

Description

This function sets the trigger source of the power meter.

.NET Method Prototype

```
N/A
```

(Use the `Trigger.Source` property)

COM Method Prototype

```
N/A
```

(Use the `Trigger.Source` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureTriggerSource (ViSession Vi,
                                             ViInt32 TriggerSource);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Trigger Source	Specifies the trigger source. The value of this parameter is used to set the Trigger Source attribute. See the attribute description for more information.	ViInt32

Return Values (C)

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

7.4 IviPwrMeterTriggerSource Behavior Model

The IviPwrMeterTriggerSource behavior model leverages the behavior model of the IviPwrMeterBase capability group. Furthermore, it defines an additional *Wait-For-Trigger* state after the *Idle* state. The following state diagram shows the relationship between IviPwrMeterTriggerSource capabilities and the IviPwrMeterBase capability behavior model.
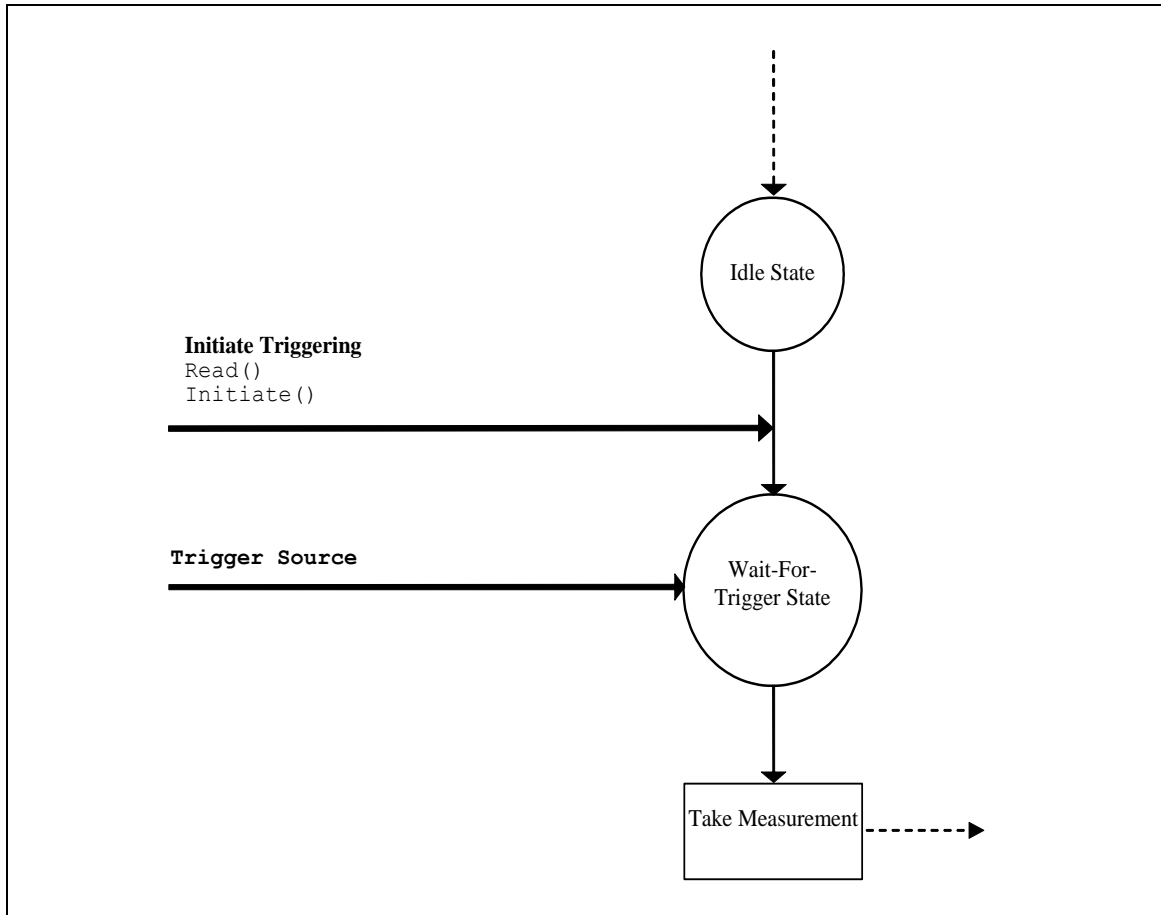
Figure 2. IviPwrMeterTriggerSource Behavior Model

The Initiate and Read functions cause the power meter to leave the *Idle* state and transition to the *Wait-For-Trigger* state. The Read function does not return until the measurement process is complete and the power meter returns to the *Idle* state. The Initiate function returns as soon as the power meter leaves the *Idle* state.

In the Wait-For-Trigger state, the power meter waits for a trigger event. The type of trigger event is specified by the Trigger Source attribute. When the specified trigger event occurs, the power meter leaves the Wait-For-Trigger state and takes a measurement on all enabled channels.

8. IviPwrMeterInternalTrigger Extension Group

8.1 Overview

The IviPwrMeterInternalTrigger extension capability group supports power meters that can trigger internally on the measurement signal. It specifies attributes to configure the internal trigger event source, trigger level, and the trigger slope. The IviPwrMeterInternalTrigger extension capability group also includes functions for configuring these attributes.

8.2 IviPwrMeterInternalTrigger Attributes

The IviPwrMeterInternalTrigger capability group defines the following attributes:

- Internal Trigger Event Source

- Internal Trigger Level

- Internal Trigger Slope

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions.*

8.2.1 Internal Trigger Event Source

Data Type	Access	Applies To	Coercion	High Level Functions
ViString	R/W	N/A	None	Configure Internal Trigger

.NET Property Name

```
Trigger.Internal.EventSource
```

COM Property Name

```
Trigger.Internal.EventSource
```

C Constant Name

```
IVIPWRMETER_ATTR_INTERNAL_TRIGGER_EVENT_SOURCE
```

Description

Specifies the channel that the power meter uses to monitor the internal trigger event. The power meter leaves the *Idle* state when the measurement signal on this channel meets the conditions set by the Internal Trigger Level and the Internal Trigger Slope attributes.

This attribute affects the behavior of the instrument only if the Trigger Source attribute is set to Internal.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

8.2.2 Internal Trigger Level

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Real 64	R/ W	N/A	None	Configure Internal Trigger Level

.NET Property Name

    Trigger.Internal.Level

COM Property Name

    Trigger.Internal.Level

C Constant Name

    IVIPWRMETER_ATTR_INTERNAL_TRIGGER_LEVEL

Description

Specifies the trigger level for the measurement signal. The power meter leaves the *Idle* state when the measurement signal on the channel specified by the Internal Trigger Source Event attribute crosses the value specified by this attribute. The value of this attribute is specified in the same unit as the value of the Units attribute.

This attribute affects the behavior of the instrument only if the Trigger Source attribute is set to Internal.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

8.2.3 Internal Trigger Slope

Data Type	Access	Applies To	Coercion	High Level Functions
ViInt32	R/W	N/A	None	Configure Internal Trigger

.NET Property Name

> Trigger.Internal.Slope

.NET Enumeration Name

> Slope

COM Property Name

> Trigger.Internal.Slope

COM Enumeration Name

> IviPwrMeterInternalTriggerSlopeEnum

C Constant Name

> IVIPWRMETER_ATTR_INTERNAL_TRIGGER_SLOPE

Description

> Specifies the polarity of the internal trigger slope. The power meter triggers on the rising or falling edge of the internal trigger source depending on the value of this attribute.

> This attribute affects the behavior of the instrument only if the Trigger Source attribute is set to Internal.

Defined Values

Name	Description		
		Language	*Identifier*
Positive	Sets the trigger event to occur on the rising edge of the trigger pulse.		
	.NET	Slope.Positive	
	C	IVIPWRMETER_VAL_POSITIVE	
	COM	IviPwrMeterInternalTriggerSlopePositive	
Negative	Sets the trigger event to occur on the falling edge of the trigger pulse.		
	.NET	Slope.Negative	
	C	IVIPWRMETER_VAL_NEGATIVE	
	COM	IviPwrMeterInternalTriggerSlopeNegative	

.NET Exceptions

> The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

Compliance Notes

1. If an IVI-C specific driver defines additional values for this attribute, the magnitude of the actual values shall be greater than or equal to
   `IVIPWRMETER_VAL_INTERNAL_TRIGGER_SLOPE_SPECIFIC_EXT_BASE`.

2. If an IVI-C class driver defines additional values for this attribute, the magnitude of the actual values shall be greater than or equal to `IVIPWRMETER_VAL_INTERNAL_TRIGGER_SLOPE_CLASS_EXT_BASE` and less than `IVIPWRMETER_VAL_INTERNAL_TRIGGER_SLOPE_SPECIFIC_EXT_BASE`.

3. When an IVI-COM specific driver implements this attribute with additional elements in its instrument specific interfaces, it is recommended that the actual values of the additional elements be greater than or equal to Internal Trigger Slope Specific Extension Base.

   See Section 16, Attribute Value Definitions, for the definitions of Internal Trigger Slope Specific Extension Base, `IVIPWRMETER_VAL_INTERNAL_TRIGGER_SLOPE_SPECIFIC_EXT_BASE` and `IVIPWRMETER_VAL_INTERNAL_TRIGGER_SLOPE_CLASS_EXT_BASE`.

8.3 IviPwrMeterInternalTrigger Functions

The IviPwrMeterInternalTrigger extension capability group defines the following functions:

- Configure Internal Trigger

- Configure Internal Trigger Level (IVI-C Only)

This section describes the behavior and requirements of each function.

8.3.1 Configure Internal Trigger

Description

This function configures the internal trigger event source and the internal trigger slope of the power meter.

.NET Method Prototype

```
void Trigger.Internal.Configure (String eventSource, Slope slope);
```

COM Method Prototype

```
HRESULT Trigger.Internal.Configure ([in] BSTR EventSource,
                                    [in] IviPwrMeterInternalTriggerSlopeEnum
                                    Slope);
```

C Prototype

```
ViStatus IviPwrMeter_ConfigureInternalTrigger (ViSession Vi,
                                               ViConstString EventSource,
                                               ViInt32 Slope);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument hcndle.	ViSession
Event Source	The name of the channel to use as the internal trigger event source. The driver uses this value to set the Internal Trigger Event Source attribute. See the attribute description for more information.	ViConstString
Slope	The internal trigger slope. The driver uses this value to set the Internal Trigger Slope attribute. See the attribute description for more information.	ViInt32

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

8.3.2 Configure Internal Trigger Level (IVI-C Only)

Description

This function configures the internal trigger level of the power meter.

.NET Method Prototype

```
N/A
```

(Use the `Trigger.Internal.Level` property)

COM Method Prototype

```
N/A
```

(Use the `Trigger.Internal.Level` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureInternalTriggerLevel (ViSession Vi,
                                                     ViReal64 TriggerLevel);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
TriggerLevel	The signal trigger level. The driver uses this value to set the Internal Trigger Level attribute. See the attribute description for more information.	ViReal64

Return Values (C)

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

8.4 IviPwrMeterInternalTrigger Behavior Model

The IviPwrMeterInternalTrigger behavior model leverages the behavior model of the IviPwrMeterBase capability group and the IviPwrMeterTriggerSource extension group. Furthermore, it defines an additional trigger event for the trigger source.

The power meter leaves the Wait-For-Trigger state when it receives an internal trigger event. When the Trigger Source attribute is set to the Internal defined value, and the measurement signal on the channel specified by the Internal Trigger Event Source attribute crosses the level specified by the Internal Trigger Level attribute in the direction specified by the Internal Trigger Slope attribute, the power meter leaves the Wait-For-Trigger state and takes a measurement on all enabled channels.

8.5 IviPwrMeterInternalTrigger Compliance Notes

1. Specific drivers that implement this extension group shall implement the Internal defined value for the Trigger Source attribute in the IviPwrMeterTriggerSource extension group.

9. IviPwrMeterSoftwareTrigger Extension Group

9.1 IviPwrMeterSoftwareTrigger Extension Group Overview

The IviPwrMeterSoftwareTrigger extension group supports power meters that can initiate a measurement based on a software trigger signal. The user can send a software trigger to cause the power meter to trigger a measurement.

9.2 IviPwrMeterSoftwareTrigger Functions

The IviPwrMeterSoftwareTrigger extension group defines the following function:

- Send Software Trigger

This section describes the behavior and requirements of this function.

9.2.1 Send Software Trigger

Refer to *IVI-3.3: Standard Cross Class Capabilities Specification* for the prototype and complete description of this function.

9.3 IviPwrMeterSoftwareTrigger Behavior Model

The IviPwrMeterSoftwareTrigger behavior model leverages the behavior model of the IviPwrMeterBase capability group and the IviPwrMeterTriggerSource extension group. Furthermore, it defines an additional trigger event for the trigger source.

The power meter leaves the Wait-For-Trigger state when it receives a trigger event specified by the Trigger Source attribute. When the trigger source is set to Software, the Send Software Trigger function is used to generate the trigger event. Calling this function causes the power meter to leave the Wait-For-Trigger state and take a measurement on all enabled channels.

9.4 IviPwrMeterSoftwareTrigger Compliance Notes

1. Specific drivers that implement this extension group shall implement the Software defined value for the Trigger Source attribute in the IviPwrMeterTriggerSource extension group.

10. IviPwrMeterDutyCycleCorrection Extension Group

10.1 Overview

The IviPwrMeterDutyCycleCorrection extension capability group supports power meters that perform a duty cycle correction. The IviPwrMeterDutyCycleCorrection extension capability also includes functions for enabling and configuring the duty cycle correction.

10.2 IviPwrMeterDutyCycleCorrection Attributes

The IviPwrMeterDutyCycleCorrection capability group defines the following attributes:

- Duty Cycle Correction

- Duty Cycle Correction Enabled

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions.*

10.2.1 Duty Cycle Correction

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Real 64	R/ W	Channel	Up	Configure Duty Cycle Correction

.NET Property Name

    Channels[].DutyCycle.Value

COM Property Name

    Channels.Item().DutyCycle.Value

C Constant Name

    IVIPWRMETER_ATTR_DUTY_CYCLE_CORRECTION

Description

Specifies the duty cycle correction the power meter uses to calculate the pulse power of a pulse-modulated signal. The power meter measures the average power of the pulsed input signal and then divides the result by the value specified for this attribute to obtain a pulse power reading.

The value of this attribute is specified as a percentage.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

10.2.2 Duty Cycle Correction Enabled

Data Type	Access	Applies To	Coercion	High Level Functions
ViBoolean	R/W	Channel	None	Configure Duty Cycle Correction

.NET Property Name

> Channels[].DutyCycle.Enabled

COM Property Name

> Channels.Item().DutyCycle.Enabled

C Constant Name

> IVIPWRMETER_ATTR_DUTY_CYCLE_CORRECTION_ENABLED

Description

> Specifies if the power meter performs a duty cycle correction on the specified channel. If set to True, enables the duty cycle correction. If set to False, disables the duty cycle correction.

.NET Exceptions

> The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

10.3 IviPwrMeterDutyCycleCorrection Functions

The IviPwrMeterDutyCycleCorrection extension capability group defines the following functions:

- Configure Duty Cycle Correction

This section describes the behavior and requirements of each function.

10.3.1 Configure Duty Cycle Correction

Description

This function enables or disables the duty cycle correction and sets the duty cycle correction for pulse power measurements.

.NET Method Prototype

```
void Channels[].DutyCycle.Configure (Boolean enabled,
                                     Double correction);
```

COM Method Prototype

```
HRESULT Channels.Item().DutyCycle.Configure ([in] VARIANT_BOOL Enabled,
                                             [in] DOUBLE Value);
```

C Prototype

```
ViStatus IviPwrMeter_ConfigureDutyCycleCorrection (ViSession Vi,
                                                   ViConstString Channel,
                                                   ViBoolean CorrectionEnabled,
                                                   ViReal64 Correction);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name of the channel for which to specify the duty cycle correction.	ViConstString
CorrectionEnabled	Enables or disables the duty cycle correction. The value of this parameter is used to set the Duty Cycle Correction Enabled attribute.	ViBoolean
Correction	Specifies the expected value of the duty cycle correction. The value of this parameter is used to set the Duty Cycle Correction attribute.	ViReal64

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

10.4 IviPwrMeterDutyCycleCorrection Behavior Model

The IviPwrMeterDutyCycleCorrection extension group follows the same behavior model as the IviPwrMeterBase capability group described in Section 4.4, *IviPwrMeterBase Behavior Model.*

11. IviPwrMeterAveragingCount Extension Group

11.1 Overview

The IviPwrMeterAveragingCount extension capability group supports power meters that can filter a signal by averaging it a specified number of times in manual averaging mode. The IviPwrMeterAveragingCount group defines an attribute and function to specify the averaging count.

11.2 IviPwrMeterAveragingCount Attributes

The IviPwrMeterAveragingCount capability group defines the following attributes:

- Averaging Count

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions*.

11.2.1 Averaging Count

Data Type	Access	Applies To	Coercion	High Level Functions
ViInt32	R/W	Channel	Up	Configure Averaging Count

.NET Property Name

    Channels[].Averaging.Count

COM Property Name

    Channels.Item().Averaging.Count

C Constant Name

    IVIPWRMETER_ATTR_AVERAGING_COUNT

Description

Specifies the average count that the instrument uses in manual averaging mode. When the Auto Averaging Enabled attribute is set to False, the driver filters the input signal by averaging it the number of times specified by this attribute. For C and COM, if this attribute is set, the state of auto-averaging is indeterminate. For .NET, when this attribute is set, auto-averaging is disabled.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

11.3 IviPwrMeterAveragingCount Functions

The IviPwrMeterAveragingCount extension capability group defines the following functions:

- Configure Averaging Count (IVI-C Only)

This section describes the behavior and requirements of each function.

11.3.1 Configure Averaging Count (IVI-C Only)

Description

This function sets the average count that the instrument uses in manual averaging mode. The averaging count specifies the number of samples that the instrument takes before the measurement is complete. For C, the impact of this function on the Averaging Auto Enabled property is undefined.

.NET Method Prototype

```
N/A
```

(Use the `Channels[].Averaging.Count` property)

COM Method Prototype

```
N/A
```

(Use the `Channels.Item().Averaging.Count` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureAveragingCount (ViSession Vi,
                                              ViConstString Channel,
                                              ViInt32 Count);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name of the channel to configure.	ViConstString
Count	The averaging count. The value of this parameter is used to set the Averaging Count attribute.	ViInt32

Return Values (C)

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

11.4 IviPwrMeterAveragingCount Behavior Model

The IviPwrMeterAveragingCount extension group follows the same behavior model as the IviPwrMeterBase capability group described in Section 4.4, *IviPwrMeterBase Behavior Model.*

12. IviPwrMeterZeroCorrection Extension Group

12.1 Overview

The IviPwrMeterZeroCorrection extension capability group supports power meters that can perform a zero correction on an input channel. The IviPwrMeterZeroCorrection capability group defines functions to perform the zero correction.

12.2 IviPwrMeterZeroCorrection Attributes

The IviPwrMeterZeroCorrection capability group defines the following attributes:

- Zero State (IVI.NET Only)

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions.*

12.2.1 Zero State (IVI.NET Only)

Data Type	Access	Applies To	Coercion	High Level Functions
ViInt32	R/O	Channel	None	N/A

.NET Property Name

> Channels.ZeroState

.NET Enumeration Name

> Ivi.PwrMeter.OperationState

COM Property Name

> N/A

(Use the Is Zero Complete function.)

C Constant Name

> N/A

(Use the Is Zero Complete function.)

Description

This property returns the results of a query to the instrument to determine the status of all zero correction operations initiated by the Zero or Zero All Channel functions. The driver returns Complete only when zero corrections are complete on all enabled channels.

If some zero correction operations are still in progress on one or more channels, the driver returns In Progress.

If the driver cannot query the instrument to determine its state, the driver returns State Unknown.

The driver does not check the instrument status to determine the measurement state. Typically, the end-user accesses this property only in a sequence of other driver calls. The sequence performs one operation. The end-user uses the low-level functions to optimize one or more aspects of interaction with the instrument. To check the instrument status, call the Error Query function at the conclusion of the sequence.

Defined Values

Name	*Description*		
		Language	*Identifier*
Complete	The power meter has completed all zero correction operations.		
		.NET	Ivi.PwrMeter.OperationState.Complete
In Progress	The power meter is still performing zero correction on one or more channels.		
		.NET	Ivi.PwrMeter.OperationState.InProgress
State Unknown	The power meter cannot determine the status of zero correction operations.		
		.NET	Ivi.PwrMeter.OperationState.Unknown

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

12.3 IviPwrMeterZeroCorrection Functions

The IviPwrMeterZeroCorrection extension capability group defines the following functions:

- Is Zero Complete

- Zero

- Zero All Channels

This section describes the behavior and requirements of each function.

12.3.1 Is Zero Complete (IVI-C & IVI-COM Only)

Description

This function queries the instrument to determine the status of all zero correction operations initiated by the Zero or Zero All Channel functions. This function returns the Zero Complete value in the Status parameter only when zero corrections are complete on all enabled channels.

If some zero correction operations are still in progress on one or more channels, the driver returns the Zero In Progress value. If the driver cannot query the instrument to determine its state, the driver returns the Zero Status Unknown value.

This function does not check the instrument status. Typically, the end-user calls this function only in a sequence of calls to other low-level driver functions. The sequence performs one operation. The end-user uses the low-level functions to optimize one or more aspects of interaction with the instrument. To check the instrument status, call the Error Query function at the conclusion of the sequence.

.NET Method Prototype

```
N/A
```

(Use the Zero State attribute.)

COM Method Prototype

```
HRESULT Channels.IsZeroComplete ([out, retval] IviPwrMeterZeroStatusEnum*
                                 Status);
```

C Prototype

```
ViStatus IviPwrMeter_IsZeroComplete (ViSession Vi,
                                     ViInt32 *Status);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession

Outputs	Description	Datatype
Status	Returns the status of the measurement.	ViInt32*

Defined Values for Status Parameter

Name	Description		
		Language	Identifier
Zero Complete	The power meter has completed all zero correction operations.		
		C	IVIPWRMETER_VAL_ZERO_COMPLETE
		COM	IviPwrMeterZeroStatusComplete
Zero In Progress	The power meter is still performing zero correction on one or more channels.		
		C	IVIPWRMETER_VAL_ZERO_IN_PROGRESS
		COM	IviPwrMeterZeroStatusInProgress

Zero Status Unknown	The power meter cannot determine the status of zero correction operations.	
	C	IVIPWRMETER_VAL_ZERO_STATUS_UNKNOWN
	COM	IviPwrMeterZeroStatusUnknown

Return Values (C/COM)

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

12.3.2 Zero

Description

This function performs a zero correction on the specified channel. The user may use the Is Zero Complete function to determine when the zero correction is complete.

.NET Method Prototype

```
void Channels[].Zero ();
```

COM Method Prototype

```
HRESULT Channels.Item().Zero ();
```

C Prototype

```
ViStatus IviPwrMeter_Zero (ViSession Vi,
                           ViConstString Channel);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name for the channel for which to perform the zero correction	ViConstString

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

12.3.3 Zero All Channels

Description

This function performs a zero correction on all enabled channels. The user may use the Is Zero Complete function to determine when the zero correction is complete.

.NET Method Prototype

```
void Channels.Zero ();
```

COM Method Prototype

```
HRESULT Channels.Zero ();
```

C Prototype

```
ViStatus IviPwrMeter_ZeroAllChannels (ViSession Vi);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

12.4 IviPwrMeterZeroCorrection Behavior Model

The IviPwrMeterZeroCorrection extension group follows the same behavior model as the IviPwrMeterBase capability group described in Section 4.4, *IviPwrMeterBase Behavior Model.*

13. IviPwrMeterCalibration Extension Group

13.1 Overview

The IviPwrMeterCalibration extension capability group supports power meters that can perform calibration for a given power sensor. The IviPwrMeterCalibration capability group defines functions to perform the calibration.

13.2 IviPwrMeterCalibration Attributes

The IviPwrMeterCalibration capability group defines the following attributes:

- Calibration State (IVI.NET Only)

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions.*

13.2.1 Calibration State (IVI.NET Only)

Data Type	Access	Applies To	Coercion	High Level Functions
ViInt32	R/O	Channel	None	N/A

.NET Property Name

> Channels.CalibrationState

.NET Enumeration Name

> Ivi.PwrMeter.OperationState

COM Property Name

> N/A

(Use the Is Calibration Complete function.)

C Constant Name

> N/A

(Use the Is Calibration Complete function.)

Description

This property returns the results of a query to the instrument to determine the status of all calibration operations initiated by the Calibrate function. The driver returns Complete only when calibration is complete on all enabled channels.

If some calibration operations are still in progress on one or more channels, the driver returns In Progress.

If the driver cannot query the instrument to determine its state, the driver returns State Unknown.

The driver does not check the instrument status to determine the measurement state. Typically, the end-user accesses this property only in a sequence of other driver calls. The sequence performs one operation. The end-user uses the low-level functions to optimize one or more aspects of interaction with the instrument. To check the instrument status, call the Error Query function at the conclusion of the sequence.

Defined Values

Name	Description	
	Language	*Identifier*
Complete	The power meter has completed all calibration operations.	
	.NET	Ivi.PwrMeter.OperationState.Complete
In Progress	The power meter is still performing calibration on one or more channels.	
	.NET	Ivi.PwrMeter.OperationState.InProgress
State Unknown	The power meter cannot determine the status of calibration operations.	
	.NET	Ivi.PwrMeter.OperationState.Unknown

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

13.3 IviPwrMeterCalibration Functions

The IviPwrMeterCalibration extension capability group defines the following functions:

- Calibrate

- Is Calibration Complete

This section describes the behavior and requirements of each function.

13.3.1 Calibrate

Description

This function performs calibration on the specified sensor. The user may use the Is Calibration Complete function to determine when the calibration is complete.

.NET Method Prototype

```
void Channels[].Calibrate ();
```

COM Method Prototype

```
HRESULT Channels.Item().Calibrate ();
```

C Prototype

```
ViStatus IviPwrMeter_Calibrate (ViSession Vi,
                                ViConstString Channel);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Channel	The name of the channel to calibrate.	ViConstString

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

13.3.2 Is Calibration Complete (IVI-C & IVI-COM Only)

Description

This function queries the instrument to determine the status of all calibration operations initiated by the Calibrate function. This function returns the Calibration Complete value in the Status parameter only when calibration is complete on all channels.

If some calibration operations are still in progress on one or more channels, the driver returns the Calibration In Progress value. If the driver cannot query the instrument to determine its state, the driver returns the Calibration Status Unknown value.

This function does not check the instrument status. Typically, the end-user calls this function only in a sequence of calls to other low-level driver functions. The sequence performs one operation. The end-user uses the low-level functions to optimize one or more aspects of interaction with the instrument. To check the instrument status, call the Error Query function at the conclusion of the sequence.

.NET Method Prototype

```
N/A
```

(Use the Calibration State attribute.)

COM Method Prototype

```
HRESULT Channels.IsCalibrationComplete ([out, retval]
                                        IviPwrMeterCalibrationStatusEnum*
                                        Status);
```

C Prototype

```
ViStatus IviPwrMeter_IsCalibrationComplete (ViSession Vi,
                                            ViInt32 *Status);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession

Outputs	Description	Datatype
Status	Returns the status of the measurement.	ViInt32*

Defined Values for Status Parameter

Name	Description		
	Language	**Identifier**	
Calibration Complete	The power meter has completed the calibration.		
	C	IVIPWRMETER_VAL_CALIBRATION_COMPLETE	
	COM	IviPwrMeterCalibrationStatusComplete	
Calibration In Progress	The power meter is still performing calibration.		
	C	IVIPWRMETER_VAL_CALIBRATION_IN_PROGRESS	
	COM	IviPwrMeterCalibrationStatusInProgress	

Calibration Status Unknown	The power meter cannot determine the status of the calibration.	
	C	IVIPWRMETER_VAL_CALIBRATION_STATUS_UNKNOWN
	COM	IviPwrMeterCalibrationStatusUnknown

Return Values (C/COM)

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

13.4 IviPwrMeterCalibration Behavior Model

The IviPwrMeterCalibration extension group follows the same behavior model as the IviPwrMeterBase capability group described in Section 4.4, *IviPwrMeterBase Behavior Model.*

14. IviPwrMeterReferenceOscillator Extension Group

14.1 Overview

The IviPwrMeterReferenceOscillator extension capability group supports power meters that can enable an internal reference oscillator. The IviPwrMeterReferenceOscillator capability group defines attributes that configure the reference oscillator. It also defines functions to set these attributes.

14.2 IviPwrMeterReferenceOscillator Attributes

The IviPwrMeterReferenceOscillator capability group defines the following attributes:

- Reference Oscillator Enabled

- Reference Oscillator Frequency

- Reference Oscillator Level

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 15, *IviPwrMeter Attribute ID Definitions*.

14.2.1 Reference Oscillator Enabled

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Boolean	R/W	N/A	None	Configure Reference Oscillator Enabled

.NET Property Name

ReferenceOscillator.Enabled

COM Property Name

ReferenceOscillator.Enabled

C Constant Name

IVIPWRMETER_ATTR_REF_OSCILLATOR_ENABLED

Description

If set to True, enables the internal reference oscillator. If set to False, disables the internal reference oscillator.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

14.2.2 Reference Oscillator Frequency

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Real 64	R/ W	N/A	None	Configure Reference Oscillator

.NET Property Name

ReferenceOscillator.Frequency

COM Property Name

ReferenceOscillator.Frequency

C Constant Name

IVIPWRMETER_ATTR_REF_OSCILLATOR_FREQUENCY

Description

Specifies the frequency of the signal generated by the reference oscillator in Hertz. This attribute affects the behavior of the instrument only if the Reference Oscillator Enabled attribute is set to True.

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

14.2.3 Reference Oscillator Level

Data Type	Access	Applies To	Coercion	High Level Functions
Vi Real 64	R/ W	N/A	None	Configure Reference Oscillator

.NET Property Name

> ReferenceOscillator.Level

COM Property Name

> ReferenceOscillator.Level

C Constant Name

> IVIPWRMETER_ATTR_REF_OSCILLATOR_LEVEL

Description

> Specifies the power level of the signal generated by the reference oscillator in dBm. This attribute affects the behavior of the instrument only if the Reference Oscillator Enabled attribute is set to True.

.NET Exceptions

> The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this property.

14.3 IviPwrMeterReferenceOscillator Functions

The IviPwrMeterReferenceOscillator extension capability group defines the following functions:

- Configure Reference Oscillator
- Configure Reference Oscillator Enabled (IVI-C Only)

This section describes the behavior and requirements of each function.

14.3.1 Configure Reference Oscillator

Description

This function sets the frequency and power level of the signal generated by the reference oscillator.

.NET Method Prototype

```
void ReferenceOscillator.Configure (Double frequency,
                                    Double level);
```

COM Method Prototype

```
HRESULT ReferenceOscillator.Configure ([in] DOUBLE Frequency,
                                       [in] DOUBLE Level);
```

C Prototype

```
ViStatus IviPwrMeter_ConfigureRefOscillator (ViSession Vi,
                                             ViReal64 Frequency,
                                             ViReal64 Level);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
Frequency	The frequency of the reference oscillator. The value of this parameter is used to set the Reference Oscillator Frequency attribute.	ViReal64
Level	The power level of the reference oscillator. The value of this parameter is used to set the Reference Oscillator Level attribute.	ViReal64

Return Values (C/COM)

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

.NET Exceptions

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method.

14.3.2 Configure Reference Oscillator Enabled (IVI-C Only)

Description

This function enables or disables the reference oscillator.

.NET Method Prototype

```
N/A
```

(Use the `ReferenceOscillator.Enabled` property)

COM Method Prototype

```
N/A
```

(Use the `ReferenceOscillator.Enabled` property)

C Prototype

```
ViStatus IviPwrMeter_ConfigureRefOscillatorEnabled (ViSession Vi,
                                              ViBoolean RefOscillatorEnabled);
```

Parameters

Inputs	Description	Datatype
Vi	Instrument handle.	ViSession
RefOscillatorEnabled	Pass True to enable the reference oscillator. Pass False to disable the reference oscillator. The value of this parameter is used to set the Reference Oscillator Enabled attribute. See the attribute description for more information.	ViBoolean

Return Values (C)

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

14.4 IviPwrMeterReferenceOscillator Behavior Model

The IviPwrMeterReferenceOscillator extension group follows the same behavior model as the IviPwrMeterBase capability group described in Section 4.4, *IviPwrMeterBase Behavior Model.*

15. IviPwrMeter Attribute ID Definitions

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

Table 15-1. IviPwrMeter Attributes ID Values

Attribute Name	ID Definition
IVIPWRMETER_ATTR_UNITS	IVI_CLASS_ATTR_BASE + 1
IVIPWRMETER_ATTR_RANGE_AUTO_ENABLED	IVI_CLASS_ATTR_BASE + 2
IVIPWRMETER_ATTR_AVERAGING_AUTO_ENABLED	IVI_CLASS_ATTR_BASE + 3
IVIPWRMETER_ATTR_CORRECTION_FREQUENCY	IVI_CLASS_ATTR_BASE + 4
IVIPWRMETER_ATTR_OFFSET	IVI_CLASS_ATTR_BASE + 5
IVIPWRMETER_ATTR_CHANNEL_COUNT	IVI_INHERENT_ATTR_BASE + 203
IVIPWRMETER_ATTR_CHANNEL_ENABLED	IVI_CLASS_ATTR_BASE + 51
IVIPWRMETER_ATTR_RANGE_LOWER	IVI_CLASS_ATTR_BASE + 101
IVIPWRMETER_ATTR_RANGE_UPPER	IVI_CLASS_ATTR_BASE + 102
IVIPWRMETER_ATTR_TRIGGER_SOURCE	IVI_CLASS_ATTR_BASE + 201
IVIPWRMETER_ATTR_INTERNAL_TRIGGER_EVENT_SOURCE	IVI_CLASS_ATTR_BASE + 251
IVIPWRMETER_ATTR_INTERNAL_TRIGGER_LEVEL	IVI_CLASS_ATTR_BASE + 252
IVIPWRMETER_ATTR_INTERNAL_TRIGGER_SLOPE	IVI_CLASS_ATTR_BASE + 253
IVIPWRMETER_ATTR_AVERAGING_COUNT	IVI_CLASS_ATTR_BASE + 301
IVIPWRMETER_ATTR_DUTY_CYCLE_CORRECTION_ENABLED	IVI_CLASS_ATTR_BASE + 401
IVIPWRMETER_ATTR_DUTY_CYCLE_CORRECTION	IVI_CLASS_ATTR_BASE + 402
IVIPWRMETER_ATTR_REF_OSCILLATOR_ENABLED	IVI_CLASS_ATTR_BASE + 501
IVIPWRMETER_ATTR_REF_OSCILLATOR_FREQUENCY	IVI_CLASS_ATTR_BASE + 502
IVIPWRMETER_ATTR_REF_OSCILLATOR_LEVEL	IVI_CLASS_ATTR_BASE + 503

16. IviPwrMeter Attribute Value Definitions

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

Internal Trigger Slope

Value Name	*Language*	*Identifier*	*Actual Value*
Negative	.NET	`Slope.Negative`	1
	C	`IVIPWRMETER_VAL_NEGATIVE`	0
	COM	`IviPwrMeterInternalTriggerSlopeNegative`	0
Positive	.NET	`Slope.Positive`	0
	C	`IVIPWRMETER_VAL_POSITIVE`	1
	COM	`IviPwrMeterInternalTriggerSlopePositive`	1
Internal Trigger Slope Class Extension Base	C	`IVIPWRMETER_VAL_INTERNAL_TRIGGER_SLOPE_CLASS_EXT_BASE`	500
Internal Trigger Slope Specific Extension Base	C	`IVIPWRMETER_VAL_INTERNAL_TRIGGER_SLOPE_SPECIFIC_EXT_BASE`	1000
	COM	`N/A`	

Trigger Source (IVI-C & IVI-COM Only)

Value Name	*Language*	*Identifier*	*Actual Value*
Immediate	C	`IVIPWRMETER_VAL_IMMEDIATE`	1
	COM	`IviPwrMeterTriggerSourceImmediate`	1
External	C	`IVIPWRMETER_VAL_EXTERNAL`	2
	COM	`IviPwrMeterTriggerSourceExternal`	2
Internal	C	`IVIPWRMETER_VAL_INTERNAL`	3
	COM	`IviPwrMeterTriggerSourceInternal`	3
Software	C	`IVIPWRMETER_VAL_SOFTWARE_TRIG`	4
	COM	`IviPwrMeterTriggerSourceSoftware`	4
TTL0	C	`IVIPWRMETER_VAL_TTL0`	100
	COM	`IviPwrMeterTriggerSourceTTL0`	100
TTL1	C	`IVIPWRMETER_VAL_TTL1`	101
	COM	`IviPwrMeterTriggerSourceTTL1`	101
TTL2	C	`IVIPWRMETER_VAL_TTL2`	102
	COM	`IviPwrMeterTriggerSourceTTL2`	102
TTL3	C	`IVIPWRMETER_VAL_TTL3`	103
	COM	`IviPwrMeterTriggerSourceTTL3`	103
TTL4	C	`IVIPWRMETER_VAL_TTL4`	104
	COM	`IviPwrMeterTriggerSourceTTL4`	104

Value Name	Language	Identifier	Actual Value
TTL5	C	`IVIPWRMETER_VAL_TTL5`	105
	COM	Ivi Pwr Meter Trigger Source TTL5	105
TTL6	C	`IVIPWRMETER_VAL_TTL6`	106
	COM	Ivi Pwr Meter Trigger Source TTL6	106
TTL7	C	`IVIPWRMETER_VAL_TTL7`	107
	COM	Ivi Pwr Meter Trigger Source TTL7	107
ECL0	C	`IVIPWRMETER_VAL_ECL0`	200
	COM	Ivi Pwr Meter Trigger Source ECL0	200
ECL1	C	`IVIPWRMETER_VAL_ECL1`	201
	COM	Ivi Pwr Meter Trigger Source ECL1	201
PXI STAR	C	`IVIPWRMETER_VAL_PXI_STAR`	300
	COM	Ivi Pwr Meter Trigger Source PXI Star	300
RTSI0	C	`IVIPWRMETER_VAL_RTSI_0`	400
	COM	Ivi Pwr Meter Trigger Source RTSI 0	400
RTSI1	C	`IVIPWRMETER_VAL_RTSI_1`	401
	COM	Ivi Pwr Meter Trigger Source RTSI 1	401
RTSI2	C	`IVIPWRMETER_VAL_RTSI_2`	402
	COM	Ivi Pwr Meter Trigger Source RTSI 2	402
RTSI3	C	`IVIPWRMETER_VAL_RTSI_3`	403
	COM	Ivi Pwr Meter Trigger Source RTSI 3	403
RTSI4	C	`IVIPWRMETER_VAL_RTSI_4`	404
	COM	Ivi Pwr Meter Trigger Source RTSI 4	404
RTSI5	C	`IVIPWRMETER_VAL_RTSI_5`	405
	COM	Ivi Pwr Meter Trigger Source RTSI 5	405
RTSI6	C	`IVIPWRMETER_VAL_RTSI_6`	406
	COM	Ivi Pwr Meter Trigger Source RTSI 6	406
Trigger Source Class Extension Base	C	`IVIPWRMETER_VAL_TRIGGER_SOURCE_CLASS_EXT_BASE`	500
Trigger Source Specific Extension Base	C	`IVIPWRMETER_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE`	1000
	COM	`N/A`	

Units

Value Name	Language	Identifier	Actual Value
dBm	.NET	Units.dBm	0
	C	IVIPWRMETER_VAL_DBM	1
	COM	IviPwrMeterUnitsdBm	1
dBmV	.NET	Units.dBmV	1
	C	IVIPWRMETER_VAL_DBMV	2
	COM	IviPwrMeterUnitsdBmV	2
dBuV	.NET	Units.dBuV	2
	C	IVIPWRMETER_VAL_DBUV	3
	COM	IviPwrMeterUnitsdBuV	3
Watts	.NET	Units.Watts	3
	C	IVIPWRMETER_VAL_WATTS	4
	COM	IviPwrMeterUnitsWatts	4
Units Class Extension Base	C	IVIPWRMETER_VAL_UNITS_CLASS_EXT_BASE	500
Units Specific Extension Base	C	IVIPWRMETER_VAL_UNITS_SPECIFIC_EXT_BASE	1000
	COM	N/A	

17. IviPwrMeter Function Parameter Value Definitions

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

Configure Measurement

Parameter: Operator

COM Enumeration Name: IviPwrMeterMeasurementOperatorEnum

Value Name	Language	Identifier	Actual Value
None	.NET	MeasurementOperator.None	0
	C	IVIPWRMETER_VAL_NONE	0
	COM	IviPwrMeterMeasurementOperatorNone	0
Difference	.NET	MeasurementOperator.Difference	1
	C	IVIPWRMETER_VAL_DIFFERENCE	1
	COM	IviPwrMeterMeasurementOperatorDifference	1
Sum	.NET	MeasurementOperator.Sum	2
	C	IVIPWRMETER_VAL_SUM	2
	COM	IviPwrMeterMeasurementOperatorSum	2
Quotient	.NET	MeasurementOperator.Quotient	3
	C	IVIPWRMETER_VAL_QUOTIENT	3
	COM	IviPwrMeterMeasurementOperatorQuotient	3
Operator Class Extension Base	C	IVIPWRMETER_VAL_OPERATOR_CLASS_EXT_BASE	500
Operator Specific Extension Base	C	IVIPWRMETER_VAL_OPERATOR_SPECIFIC_EXT_BASE	1000
	COM	N/A	

Is Calibration Complete (IVI-C & IVI-COM Only)

Parameter: Status

COM Enumerator Name: IviPwrMeterCalibrationStatusEnum

Value Name	Language	Identifier	Actual Value
Calibration Complete	C	IVIPWRMETER_VAL_CALIBRATION_COMPLETE	1
	COM	IviPwrMeterCalibrationStatusComplete	1
Calibration In Progress	C	IVIPWRMETER_VAL_CALIBRATION_IN_PROGRESS	0
	COM	IviPwrMeterCalibrationStatusInProgress	0
Calibration Status Unknown	C	IVIPWRMETER_VAL_CALIBRATION_STATUS_UNKNOWN	−1
	COM	IviPwrMeterCalibrationStatusUnknown	−1

Is Measurement Complete (IVI-C & IVI-COM Only)

Parameter: Status

COM Enumeration Name: IviPwrMeterMeasurementStatusEnum

Value Name	*Language*	*Identifier*	*Actual Value*
Measurement Complete	C	IVIPWRMETER_VAL_MEAS_COMPLETE	1
	COM	IviPwrMeterMeasurementStatusComplete	1
Measurement In Progress	C	IVIPWRMETER_VAL_MEAS_IN_PROGRESS	0
	COM	IviPwrMeterMeasurementStatusInProgress	0
Measurement Status Unknown	C	IVIPWRMETER_VAL_MEAS_STATUS_UNKNOWN	-1
	COM	IviPwrMeterMeasurementStatusUnknown	-1

Is Zero Complete (IVI-C & IVI-COM Only)

Parameter: Status

COM Enumeration Name: IviPwrMeterZeroStatusEnum

Value Name	*Language*	*Identifier*	*Actual Value*
Zero Complete	C	IVIPWRMETER_VAL_ZERO_COMPLETE	1
	COM	IviPwrMeterZeroStatusComplete	1
Zero In Progress	C	IVIPWRMETER_VAL_ZERO_IN_PROGRESS	0
	COM	IviPwrMeterZeroStatusInProgress	0
Zero Status Unknown	C	IVIPWRMETER_VAL_ZERO_STATUS_UNKNOWN	-1
	COM	IviPwrMeterZeroStatusUnknown	-1

Operation State (IVI.NET Only)

Value Name	*Language*	*Identifier*	*Actual Value*
Complete	.NET	OperationState.Complete	0
In Progress	.NET	OperationState.InProgress	1
State Unknown	.NET	OperationState.Unknown	2

Query Result Range Type

Parameter: RangeType

COM Enumeration Name: IviPwrMeterResultRangeEnum

Value Name	*Language*	*Identifier*	*Actual Value*
In Range	.NET	ResultRange.InRange	0
	C	IVIPWRMETER_VAL_IN_RANGE	0

	COM	IviPwrMeterResultRangeInRange	0
Under Range	.NET	Result Range. Under Range	1
	C	IVIPWRMETER_VAL_UNDER_RANGE	-1
	COM	IviPwrMeterResultRangeUnderRange	-1
Over Range	.NET	Result Range. Over Range	2
	C	IVIPWRMETER_VAL_OVER_RANGE	1
	COM	IviPwrMeterResultRangeOverRange	1

Read

Parameter: MaxTimeMilliseconds (C/COM)

Value Name	*Language*	*Identifier*	*Actual Value*
Immediate Timeout	C	IVIPWRMETER_VAL_MAX_TIME_IMMEDIATE	0
	COM	IviPwrMeterTimeOutImmediate	0
Infinite Timeout	C	IVIPWRMETER_VAL_MAX_TIME_INFINITE	0xFFFFFFFFUL
	COM	IviPwrMeterTimeOutInfinite	0xFFFFFFFFUL

Parameter: maximumTime (.NET)

Value Name	Language	Identifier	Actual Value
Zero	.NET	PrecisionTimeSpan.Zero	PrecisionTimeSpan.Zero
Max Value	.NET	PrecisionTimeSpan.MaxValue	PrecisionTimeSpan.MaxValue

Read Channel

Parameter: MaxTimeMilliseconds (C/COM)

Value Name	Language	Identifier	Actual Value
Immediate Timeout	C	IVIPWRMETER_VAL_MAX_TIME_IMMEDIATE	0
	COM	IviPwrMeterTimeOutImmediate	0
Infinite Timeout	C	IVIPWRMETER_VAL_MAX_TIME_INFINITE	0xFFFFFFFFUL
	COM	IviPwrMeterTimeOutInfinite	0xFFFFFFFFUL

Parameter: maximumTime (.NET)

Value Name	Language	Identifier	Actual Value
Zero	.NET	PrecisionTimeSpan.Zero	PrecisionTimeSpan.Zero
MaxValue	.NET	PrecisionTimeSpan.MaxValue	PrecisionTimeSpan.MaxValue

18. IviPwrMeter Error, Completion Code, and Exception Class Definitions

The IviPwrMeter class defines the following error codes in addition to the IVI defined error codes

Table 18-1. IviPwrMeter Completion Codes

Error Name	Description		
	Language	*Identifier*	*Value(hex)*
Under Range	Measurement is under range.		
	.NET		N/A
	C	IVIPWRMETER_WARN_UNDER_RANGE	0x3FFA2001
	COM	S_IVIPWRMETER_UNDER_RANGE	0x00042001
Over Range	Measurement is over range.		
	.NET		N/A
	C	IVIPWRMETER_WARN_OVER_RANGE	0x3FFA2002
	COM	S_IVIPWRMETER_OVER_RANGE	0x00042002
Channel Not Enabled	The specified channel is not enabled for measurement.		
	.NET	Ivi.PwrMeter.ChannelNotEnabledException	N/A
	C	IVIPWRMETER_ERROR_CHANNEL_NOT_ENABLED	0xBFFA2001
	COM	E_IVIPWRMETER_CHANNEL_NOT_ENABLED	0x80042001
Max Time Exceeded	Maximum timeout exceeded before operation could complete.		
	.NET	Ivi.Driver.MaxTimeExceededException	IVI Defined Exception (See IVI-3.2)
	C	IVIPWRMETER_ERROR_MAX_TIME_EXCEEDED	0xBFFA2020
	COM	E_IVIPWRMETER_MAX_TIME_EXCEEDED	0x80042020
Trigger Not Software	The trigger source is not set to software trigger.		
	.NET	Ivi.Driver.TriggerNotSoftwareException	IVI Defined Exception (See IVI-3.2)
	C	IVIPWRMETER_ERROR_TRIGGER_NOT_SOFTWARE	0xBFFA1001
	COM	E_IVIPWRMETER_TRIGGER_NOT_SOFTWARE	0x80041001

Table 18-2 defines the recommended format of the message string associated with the error. In C, these strings are returned by the Get Error function. In COM, these strings are the description contained in the ErrorInfo object. For .NET, exception default message strings are defined with the exception.

Note: In the description string table entries listed below, **%s** is always used to represent the component name.

Table 18-2. IviPwrMeter Error Message Strings

Name	Message String
Under Range	"%s: Measurement is under range"
Over Range	"%s: Measurement is over range"
Channel Not Enabled	"%s: The channel %s1 is not enabled for measurement" %s1 = Channel Name
Max Time Exceeded	"%s: Maximum timeout exceeded before operation could complete"
Trigger Not Software	"%s: The trigger source is not set to software trigger"

18.1 IVI.NET IviPwrMeter Exceptions and Warnings

This section defines the list of IVI.NET exceptions and warnings that are specific to the IviPwrMeter class. For general information on IVI.NET exceptions and warnings, refer to *IVI-3.1: Driver Architecture Specification* and section 12, *Common IVI.NET Exceptions and Warnings*, of *IVI-3.2: Inherent Capabilities Specification*.

The IVI.NET exceptions defined in this specification are declared in the Ivi.PwrMeter namespace.
- ChannelNotEnabledException

18.1.1 ChannelNotEnabledException

Description

This exception is used when the driver finds that a channel is not enabled for measurement.

Constructors

```
Ivi.PwrMeter.ChannelNotEnabledException(String message,
                                        String channelName);

Ivi.PwrMeter.ChannelNotEnabledException();

Ivi.PwrMeter.ChannelNotEnabledException(String message);

Ivi.PwrMeter.ChannelNotEnabledException(String message,
                                        System.Exception innerException);
```

Message String

```
The channel is not enabled for measurement.
Channel name: <channelName>
```

Parameters

Inputs	Description	Base Type
ChannelName	The name of the channel that is not enabled.	String

Usage

If driver developers use constructors that take a message string, they are responsible for message string localization.

19. IviPwrMeter Hierarchies

19.1 IviPwrMeter .NET Hierarchy

The full IviPwrMeter .NET Hierarchy includes the Inherent Capabilities Hierarchy as defined in Section 4.1, *.NET Inherent Capabilities*, of IVI-3.2, *Inherent Capabilities Specification*. To avoid redundancy, it is omitted from Table 19.1 IviPwrMeter .NET Hierarchy.

Table 19-1. IviPwrMeter .NET Hierarchy

.NET Interface Hierarchy	Generic Name	Type
Channels		
Count	Channel Count	P
Units	Units	P
[]	Item	
Name	Channel Name	P
Enabled	Channel Enabled	P
CorrectionFrequency	Correction Frequency	P
Offset	Offset	P
Averaging		
CountAuto	Averaging Auto Enabled	P
Count	Averaging Count	P
Range		
Auto	Range Auto Enabled	P
Lower	Range Lower	P
Upper	Range Upper	P
Configure	Configure Range	M
DutyCycle		
Enabled	Duty Cycle Correction Enabled	P
Value	Duty Cycle Correction	P
Configure	Configure Duty Cycle Correction	M
Zero	Zero	M
Calibrate	Calibrate	M
Zero	Zero All Channels	M
ZeroState	Zero State	P
CalibrationState	Calibration State	P
Trigger		
Source	Trigger Source	P
Internal		
Configure	Configure Internal Trigger	M
EventSource	Internal Trigger Event Source	P
Slope	Internal Trigger Slope	P

Table 19-1. IviPwrMeter .NET Hierarchy

.NET Interface Hierarchy	Generic Name	Type
Level	Internal Trigger Level	P
Measurement		
Configure	Configure Measurement	M
Read	Read	M
ReadChannel	Read Channel	M
Initiate	Initiate	M
Fetch	Fetch	M
FetchChannel	Fetch Channel	M
Abort	Abort	M
GetMeasurementComplete	Is Measurement Complete	M
SendSoftwareTrigger	Send Software Trigger	M
QueryResultRangeType	Query Result Range Type	M
ReferenceOscillator		
Enabled	Reference Oscillator Enabled	P
Configure	Configure Reference Oscillator	M
Frequency	Reference Oscillator Frequency	P
Level	Reference Oscillator Level	P

19.1.1 IviPwrMeter .NET Interfaces

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

- IIviPwrMeterChannels
- IIviPwrMeterMeasurement
- IIviPwrMeterReferenceOscillator
- IIviPwrMeterTrigger

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

The IIviPwrMeterChannel interface contains interface reference properties for accessing additional the following IviPwrMeter interfaces:

- IIviPwrMeterAveraging
- IIviPwrMeterDutyCycleCorrection
- IIviPwrMeterRange

19.1.2 Interface Reference Properties

Interface reference properties are used to navigate the IviDmm .NET hierarchy. This section describes the interface reference properties that the IviPwrMeter, IIviPwrMeterChannels, and IIviPwrMeterChannel interfaces define. All interface reference properties are read-only.

Data Type	Access
IIviPwrMeterReferenceOscillator	ReferenceOscillator
IIviPwrMeterMeasurement	Measurement
IIviPwrMeterChannels	Channels
IIviPwrMeterChannel	Channels[]
IIviPwrMeterTrigger	Trigger
IIviPwrMeterInternalTrigger	InternalTrigger
IIviPwrMeterAveraging	Channels[].Averaging
IIviPwrMeterRange	Channels[].Range
IIviPwrMeterDutyCycleCorrection	Channels[].DutyCycleCorrection

19.2 IviPwrMeter COM Hierarchy

Table 19-1. IviPwrMeter COM Hierarchy

COM Interface Hierarchy	Generic Name	Type
Channels		
Count	Channel Count	P
Name	Channel Name	P
Units	Units	P
Item		
Enabled	Channel Enabled	P

Table 19-1. IviPwrMeter COM Hierarchy

COM Interface Hierarchy	Generic Name	Type
CorrectionFrequency	Correction Frequency	P
Offset	Offset	P
Averaging		
AutoEnabled	Averaging Auto Enabled	P
Count	Averaging Count	P
Range		
AutoEnabled	Range Auto Enabled	P
Lower	Range Lower	P
Upper	Range Upper	P
Configure	Configure Range	M
DutyCycle		
Enabled	Duty Cycle Correction Enabled	P
Value	Duty Cycle Correction	P
Configure	Configure Duty Cycle Correction	M
Zero	Zero	M
Calibrate	Calibrate	M
Zero	Zero All Channels	M
IsZeroComplete	Is Zero Complete	M
IsCalibrationComplete	Is Calibration Complete	M
Trigger		
Source	Trigger Source	P
Internal		
Configure	Configure Internal Trigger	M
EventSource	Internal Trigger Event Source	P
Slope	Internal Trigger Slope	P
Level	Internal Trigger Level	P
Measurement		
Configure	Configure Measurement	M
Read	Read	M
ReadChannel	Read Channel	M
Initiate	Initiate	M
Fetch	Fetch	M
FetchChannel	Fetch Channel	M
Abort	Abort	M
IsMeasurementComplete	Is Measurement Complete	M
SendSoftwareTrigger	Send Software Trigger	M
QueryResultRangeType	Query Result Range Type	M
ReferenceOscillator		

Table 19-1. IviPwrMeter COM Hierarchy

COM Interface Hierarchy	Generic Name	Type
Enabled	Reference Oscillator Enabled	P
Configure	Configure Reference Oscillator	M
Frequency	Reference Oscillator Frequency	P
Level	Reference Oscillator Level	P

19.2.1 IviPwrMeter COM Interfaces

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

- IIviPwrMeterChannels
- IIviPwrMeterMeasurement
- IIviPwrMeterReferenceOscillator
- IIviPwrMeterTrigger

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

The IIviPwrMeterChannel interface contains interface reference properties for accessing additional the following IviPwrMeter interfaces:

- IIviPwrMeterAveraging
- IIviPwrMeterDutyCycleCorrection
- IIviPwrMeterRange

Table 19-1.1. IviPwrMeter lists the interfaces that this specification defines and their GUIDs.

Table 19-1.1. IviPwrMeter Interface GUIDs

Interface	GUID
IIviPwrMeter	{47ed5314-a398-11d4-ba58-000064657374}
IIviPwrMeterAveraging	{47ed5315-a398-11d4-ba58-000064657374}
IIviPwrMeterChannel	{47ed5316-a398-11d4-ba58-000064657374}
IIviPwrMeterChannels	{47ed5317-a398-11d4-ba58-000064657374}
IIviPwrMeterDutyCycleCorrection	{47ed5318-a398-11d4-ba58-000064657374}
IIviPwrMeterMeasurement	{47ed5319-a398-11d4-ba58-000064657374}
IIviPwrMeterRange	{47ed531a-a398-11d4-ba58-000064657374}
IIviPwrMeterReferenceOscillator	{47ed531b-a398-11d4-ba58-000064657374}
IIviPwrMeterTrigger	{47ed531c-a398-11d4-ba58-000064657374}

19.2.1 Interface Reference Properties

Interface reference properties are used to navigate the IviPwrMeter .NET hierarchy. This section describes the interface reference properties that the IviPwrMeter, IIviPwrMeterChannels, and IIviPwrMeterChannel interfaces define. All interface reference properties are read-only.

Data Type	Access
IIviPwrMeterReferenceOscillator*	ReferenceOscillator
IIviPwrMeterMeasurement*	Measurement
IIviPwrMeterChannels*	Channels
IIviPwrMeterChannel*	Channel
IIviPwrMeterTrigger*	Trigger
IIviPwrMeterInternalTrigger*	InternalTrigger

Data Type	Access
`IIviPwrMeterAveraging*`	Averaging
`IIviPwrMeterRange*`	Range
`IIviPwrMeterDutyCycleCorrection*`	DutyCycleCorrection

19.2.2 IviPwrMeter COM Category

The IviPwrMeter class COM Category shall be "IviPwrMeter", and the Category ID (CATID) shall be {47ed515a-a398-11d4-ba58-000064657374}.

19.3 IviPwrMeter C Function Hierarchy

The IviPwrMeter class function hierarchy is shown in the following table.

Table 19-2. IviPwrMeter C Function Hierarchy

Name or Class	Function Name
Configuration…	
Configure Units	IviPwrMeter_ConfigureUnits
Configure Measurement	IviPwrMeter_ConfigureMeasurement
Configure Range Auto Enabled	IviPwrMeter_ConfigureRangeAutoEnabled
Configure Averaging Auto Enabled	IviPwrMeter_ConfigureAveragingAutoEnabled
Configure Correction Frequency	IviPwrMeter_ConfigureCorrectionFrequency
Configure Offset	IviPwrMeter_ConfigureOffset
Range…	
Configure Range	IviPwrMeter_ConfigureRange
Trigger…	
Configure Trigger Source	IviPwrMeter_ConfigureTriggerSource
Internal Trigger…	
Configure Internal Trigger	IviPwrMeter_ConfigureInternalTrigger
Configure Internal Trigger Level	IviPwrMeter_ConfigureInternalTriggerLevel
Averaging…	
Configure Averaging Count	IviPwrMeter_ConfigureAveragingCount
Duty Cycle…	
Configure Duty Cycle Correction	IviPwrMeter_ConfigureDutyCycleCorrection
Reference Oscillator…	
Configure Reference Oscillator Enabled	IviPwrMeter_ConfigureRefOscillatorEnabled
Configure Reference Oscillator	IviPwrMeter_ConfigureRefOscillator
Channel…	
Get Channel Name	IviPwrMeter_GetChannelName
Channel Acquisition…	
Configure Channel Enabled	IviPwrMeter_ConfigureChannelEnabled
Zeroing…	
Zero	IviPwrMeter_Zero
Zero All Channels	IviPwrMeter_ZeroAllChannels
Is Zero Complete	IviPwrMeter_IsZeroComplete
Calibration…	
Calibrate	IviPwrMeter_Calibrate
Is Calibration Complete	IviPwrMeter_IsCalibrationComplete
Measurement…	

Table 19-2. IviPwrMeter C Function Hierarchy

Name or Class	Function Name
Read	IviPwrMeter_Read
Read Channel	IviPwrMeter_ReadChannel
Low-Level Measurement…	
Initiate	IviPwrMeter_Initiate
Is Measurement Complete	IviPwrMeter_IsMeasurementComplete
Fetch	IviPwrMeter_Fetch
Fetch Channel	IviPwrMeter_FetchChannel
Query Result Range Type	IviPwrMeter_QueryResultRangeType
Abort	IviPwrMeter_Abort
Send Software Trigger	IviPwrMeter_SendSoftwareTrigger

19.4 IviPwrMeter C Attribute Hierarchy

The IviPwrMeter class attribute hierarchy is shown below.

Table 19-3. IviPwrMeter C Attributes Hierarchy

Category or Generic Attribute Name	C Defined Constant
Basic Operation	
Units	`IVIPWRMETER_ATTR_UNITS`
Range Auto Enabled	`IVIPWRMETER_ATTR_RANGE_AUTO_ENABLED`
Averaging Auto Enabled	`IVIPWRMETER_ATTR_AVERAGING_AUTO_ENABLED`
Correction Frequency	`IVIPWRMETER_ATTR_CORRECTION_FREQUENCY`
Offset	`IVIPWRMETER_ATTR_OFFSET`
Channel Count	`IVIPWRMETER_ATTR_CHANNEL_COUNT`
Channel Acquisition	
Channel Enabled	`IVIPWRMETER_ATTR_CHANNEL_ENABLED`
Manual Range	
Range Lower	`IVIPWRMETER_ATTR_RANGE_LOWER`
Range Upper	`IVIPWRMETER_ATTR_RANGE_UPPER`
Trigger	
Trigger Source	`IVIPWRMETER_ATTR_TRIGGER_SOURCE`
Internal Trigger	
Internal Trigger Event Source	`IVIPWRMETER_ATTR_INTERNAL_TRIGGER_EVENT_SOURCE`
Internal Trigger Level	`IVIPWRMETER_ATTR_INTERNAL_TRIGGER_LEVEL`
Internal Trigger Slope	`IVIPWRMETER_ATTR_INTERNAL_TRIGGER_SLOPE`
Averaging Count	
Averaging Count	`IVIPWRMETER_ATTR_AVERAGING_COUNT`
Duty Cycle	
Duty Cycle Correction Enabled	`IVIPWRMETER_ATTR_DUTY_CYCLE_CORRECTION_ENABLED`
Duty Cycle Correction	`IVIPWRMETER_ATTR_DUTY_CYCLE_CORRECTION`
Reference Oscillator	
Reference Oscillator Enabled	`IVIPWRMETER_ATTR_REF_OSCILLATOR_ENABLED`
Reference Oscillator Frequency	`IVIPWRMETER_ATTR_REF_OSCILLATOR_FREQUENCY`
Reference Oscillator Level	`IVIPWRMETER_ATTR_REF_OSCILLATOR_LEVEL`

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 IviPwrMeter 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 *<prefix>_init*, *<prefix>_InitWithOptions*, or *<prefix>_reset* 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 shall 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 IviPwrMeterChannelAcquisition Extension Group

Setting the Channel Enabled attribute to False for all channels effectively disables the IviPwrMeterChannelAcquisition extension group.

Disabling the IviPwrMeterManualRange Extension Group

The IviPwrMeterManualRange extension group affects the instrument behavior only when the Range Auto Enabled attribute is set to False. Therefore, this specification does not recommend attribute values that disable the IviPwrMeterManualRange extension group.

Disabling the IviPwrMeterTriggerSource Extension Group

Setting the Trigger Source attribute to Immediate effectively disables the IviPwrMeterTriggerSource extension group.

Disabling the IviPwrMeterInternalTrigger Extension Group

The IviPwrMeterInternalTrigger extension group affects the instrument behavior only when the Trigger Source attribute is set to the Internal defined value. Therefore, this specification does not recommend attribute values that disable the IviPwrMeterInternalTrigger extension group.

Disabling the IviPwrMeterSoftwareTrigger Extension Group

The IviPwrMeterSoftwareTrigger extension group affects the instrument behavior only when the Trigger Source attribute is set to the Software defined value. Therefore, this specification does not recommend attribute values that disable the IviPwrMeterSoftwareTrigger extension group.

Disabling the IviPwrMeterDutyCycleCorrection Extension Group

Setting the Duty Cycle Correction Enabled attribute to False for all channels effectively disables the IviPwrMeterDutyCycleCorrection extension group.

Disabling the IviPwrMeterAveragingCount Extension Group

The IviPwrMeterAveragingCount extension group affects the instrument behavior only when the Auto Averaging Enabled attribute is set to False. Therefore, this specification does not recommend attribute values that disable the IviPwrMeterAveragingCount extension group.

Disabling the IviPwrMeterZeroCorrection Extension Group

The IviPwrMeterZeroCorrection extension group does not define any attributes. Therefore, this specification does not recommend attribute values that disable the IviPwrMeterZeroCorrection extension group.

Disabling the IviPwrMeterCalibration Extension Group

The IviPwrMeterCalibration extension group does not define any attributes. Therefore, this specification does not recommend attribute values that disable the IviPwrMeterCalibration extension group.

Disabling the IviPwrMeterReferenceOscillator Extension Group

Setting the Reference Oscillator Enabled attribute to False effectively disables the IviPwrMeterReferenceOscillator extension group.

A.3 Query Instrument Status

Based on the value of the Query Instrument Status attribute, a specific driver may check the status of the instrument to see if it has encountered an error. In specific driver functions, the status check should not occur in the lowest-level signal generation functions Initiate, Abort, and Send Software Trigger. These functions are intended to give the application developer low-level control over signal generation. When calling these functions, the application developer is responsible for checking the status of the instrument. Checking status in every function at this level would also add unnecessary overhead to the specific instrument driver.

Interchangeability Checking Rules

B.1   Introduction

IVI drivers might implement a feature called interchangeability checking. Interchangeability checking returns a warning when it encounters a situation were 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.

    – Initiate

    – Read

    – Read Channel

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 each extension capability group that the application program uses. An extension capability group is considered to be used by the application program after any of the following occur:

- The application program calls a function that belongs to the extension capability group.

- The application program accesses an attribute that belongs to the extension capability group.

- The application program sets an attribute in another capability group to a value that requires the presence of the extension capability 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.

IviPwrMeterBase Capability Group

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

IviPwrMeterChannelAcquisition Group

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

IviPwrMeterManualRange Capability Group

The driver performs interchangeability checking on the IviPwrMeterManualRange group only if the application sets the Range Auto Enabled attribute to False.

IviPwrMeterTriggerSource Capability Group

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

IviPwrMeterInternalTrigger Capability Group

The driver performs interchangeability checking on the IviPwrMeterInternalTrigger group only if the application sets the Trigger Source attribute to Internal.

IviPwrMeterSoftwareTrigger Capability Group

The driver performs interchangeability checking on the IviPwrMeterSoftwareTrigger group only if the application sets the Trigger Source attribute to Software.

IviPwrMeterDutyCycleCorrection Capability Group

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

IviPwrMeterAveragingCount Capability Group

The driver performs interchangeability checking on the IviPwrMeterAveragingCount group only if the application sets the Averaging Auto Enabled attribute to False.

IviPwrMeterZeroCorrection Capability Group

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

IviPwrMeterCalibration Capability Group

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

IviPwrMeterReferenceOscillator Capability Group

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