IVI-5: IviDmm Class Specification

November 1999 Edition
Revision 2.0

Important Information

The IviDmm Class Specification (IVI-5) is authored by the IVI Foundation member companies. For a vendor membership roster list, please visit the IVI Foundation web site at `www.ivifoundation.org`, or contact the IVI Foundation at 11500 North Mopac Expressway, Austin, Texas, 78759-3504.

The IVI Foundation wants to receive your comments on this specification. You can contact the Foundation through email at `ivilistserver@ivifoundation.org`, through the web site at `www.ivifoundation.org`, or you can write to the IVI Foundation, 11500 North Mopac Expressway, Austin, Texas, 78759-3504.

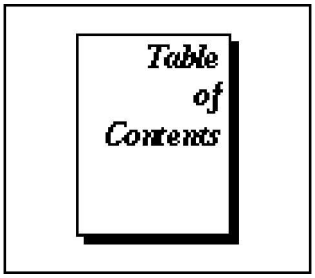
Warranty

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.



IviDmm Class Specification

IviDmm Revision History

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

Table 1. IviDmm Class Specification Revisions

Revision Number	Date of Revision	Revision Notes
Revision 0.2	April 15, 1997	Original draft.
Revision 0.3	May 15, 1997	This edition reflects the addition of the new IviDmm trigger model, the extension defaults, interchangeability checking, and guidelines for specific driver development.
Revision 0.4	July 24, 1997	This edition incorporates the channel parameter into the API as well as which attributes are channel-based.
Revision 0.5	August 12, 1997	This edition incorporates edits based on user feedback and adds introductory text.
Revision 0.6	September 24, 1997	This edition incorporates the new specification style.
Revision 0.7	June 26, 1998	This edition refines the existing documentation, and adds guidelines for specific and class drivers, attribute ID definitions, and attribute value definitions.
Revision 1.0	August 21, 1998	Technical Publications review and edit. Changes to template information.
Revision 2.0	November 22, 1999	This edition refines the organization of the specification based on feedback at the July 1999 IVI Foundation meeting. It replaces the IviDmm Miscellaneous Capabilities extension group by defining a new extension for every attribute in the group. It also defines new extension groups for AC, Frequency, and Temperature measurements.

1. Overview of the IviDmm Specification

1.1 Introduction

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

- IviDmm Class Overview
- The definitions of terms and acronyms
- References

1.2 IviDmm Class Overview

This specification defines the IVI class for digital multimeters (DMMs). The IviDmm class is designed to support the typical DMM as well as common extended functionality found in more complex instruments. The IviDmm class conceptualizes a DMM as an instrument that can measure scalar quantities of an input signal and can be applied to a wide variety of instruments. Typically the measured quantity is a voltage (AC and DC), current, or resistance. However, the IviDmm class can support instruments that measure other quantities such as temperature and frequency etc.

The IviDmm class is divided into a base capability group and several extension groups. The base capability group is used to configure a DMM for a typical measurement (this includes setting the measurement function, desired range, desired resolution, and trigger source), initiating that measurement, and returning a measured value. The IviDmm base capability group is described in Section 4, *IviDmmBase Capability Group*.

Many DMMs support measurement types that require additional parameters to be configured, such as the minimum and maximum frequency of the input signal for AC measurements. The IviDmm class defines extension groups for each measurement type that requires these additional parameters.

The IviDmm class also defines an extension group called IviDmmMultiPoint. The IviDmmMultiPoint extension group is used to configure DMMs that can acquire multiple measurements based on multiple triggers and take multiple measurements per trigger. This type of instrument used in conjunction with a scanner is typically used to implement a scanning DMM. The IviDmmMultiPoint extensions are described in Section 11, *IviDmmMultiPoint Extension Group*.

In addition, the IviDmm class defines extension groups that configure advanced settings such as auto-zero and power line frequency, or return additional information about the current state of the instrument such as aperture time. These extension groups are defined in Sections 12 through 17.

1.3 References

Several other documents and specifications are related to this specification. These other related documents are as follows:

- IVI Charter Document
- IVI Specification
- VPP-3.*x*—VXI*plug&play* Instrument Driver Specifications
- VPP-4.*x*—Virtual Instrument Software Architecture Specifications

1.4 Definitions of Terms and Acronyms

Instrument Driver	Library of functions for controlling a specific instrument
Temperature Transducer	A device that converts thermal energy into electrical energy. Used for measuring temperature.
Reference Junction	Also known as the Cold Junction. The junction of a thermocouple that is kept at a known temperature or one for which the temperature may be measured.

2. IviDmm Class Capabilities

2.1 Introduction

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

2.2 IviDmm Group Names

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

Table 2-1. IviDmm Group Names

Group Name	Description
`IviDmmBase`	Base Capability Group: DMM that complies with the IviDmmBase Capability Group.
`IviDmmACMeasurement`	Extension Group: DMM with the capability to measure AC voltage, AC current, AC plus DC voltage, and AC plus DC current.
`IviDmmFrequencyMeasurement`	Extension Group: DMM with the capability to measure frequency and period.
`IviDmmTemperatureMeasurement`	Extension Group: DMM with the capability to measure temperature.
`IviDmmThermocouple`	Extension Group: DMM with the capability to measure temperature using a thermocouple.
`IviDmmResistanceTemperatureDevice`	Extension group: DMM with the capability to measure temperature using a temperature resistance device.
`IviDmmThermistor`	Extension group: DMM with the capability to measure temperature using a thermistor.
`IviDmmMultiPoint`	Extension Group: DMM with the capability to accept multiple triggers and acquire multiple samples per trigger.
`IviDmmTriggerSlope`	Extension Group: DMM with the capability to specify the trigger slope.
`IviDmmSoftwareTrigger`	Extension Group: DMM with the capability to send a software trigger.
`IviDmmDeviceInfo`	Extension Group: DMM with the capability to return extra information concerning the instrument's state such as aperture time.
`IviDmmAutoRangeValue`	Extension Group: DMM with the capability to return the actual range when auto ranging.

Table 2-1. IviDmm Group Names

Group Name	Description
`IviDmmAutoZero`	Extension Group: DMM with the capability to take an auto-zero reading.
`IviDmmPowerLineFrequency`	Extension Group: DMM with the capability to specify the power line frequency.

2.3 Capability Group Section Layout

Each capability group section is composed of the following subsections. Optional subsections are noted:

Overview:

An overview of the capability group that describes its purpose and general use.

Attributes: (Optional)

Defines the attributes that are a part of the capability group. For each attribute the capability group defines the name of the attribute, the data type, the access (read and write - R/W, or read only - RO), a description, defined values, and additional compliance requirements. Refer to Section 2.3.1, *Attribute Section Layout* for more information regarding the layout of attribute subsections.

Functions: (Optional)

Defines the functions that are part of the capability group. For each function the capability group defines the function name, description, input parameters, output parameters, completion codes, and additional compliance requirements. Refer to Section 2.3.2, *Function Section Layout* for more information regarding the layout of function subsections.

Behavior Model:

Defines the relationships between instrument driver attributes and functions with instrument behavior.

Group Compliance Notes: (Optional)

~~Chapter~~ Section 3, *General Requirements* defines the general rules a specific driver must follow to be compliant with a capability group. This section specifies additional compliance requirements and exceptions that apply to a particular capability group.

2.3.1 Attribute Section Layout

Each Attribute section is composed of the following subsections. Optional subsections are noted:

Capabilities Table:

A table that defines the following for the attribute:

DataType:

Specifies the VXI*plug&play* data type of the attribute. Possible values include `ViInt32`, `ViReal64`, `ViBoolean`, `ViString`, and `ViSession`.

Access:

Specifies the kind of access the user has to the attribute. Possible values are *RO, WO,* and *R/W.*

- R/W (read/write) – indicates that the user can get and set the value of the attribute.
- RO (read-only) – indicates that the user can only get the value of the attribute.
- WO (write-only) – indicates that the user can only set the value of the attribute.

Channel-Based:

Specifies whether the attribute applies to the instrument as a whole or applies to each channel. Possible values are *Yes* and *No.*

- Yes – indicates that the attribute is channel-based and applies separately to each channel.
- No – indicates that the attribute is not channel-based and applies to the instrument as a whole.

Coercion:

Some attributes represent a continuous range of values, but allow the driver to coerce the value that the user requests to a value that is more appropriate for the instrument. For these cases the specification defines the direction in which the driver is allowed to coerce a value. Possible values are *Up, Down,* and *None.*

- Up – indicates that a specific driver is allowed to coerce a user-requested value to the nearest value that the instrument supports that is greater than or equal to the user-requested value.
- Down – indicates that a specific driver is allowed to coerce a user-requested value to the nearest value that the instrument supports that is less than or equal to the user-requested value.
- None – indicates that the specific driver is not allowed to coerce a user-requested value. If the instrument cannot be set to the user-requested value, the driver must return an error.

Any other kind of coercion is indicated with a documentation note.

High Level Function(s):

Lists all high level functions that access the attribute.

Description:

Describes the attribute and its intended use.

Defined Values: (Optional)

Defines all the attribute values that the class specifies for the attribute.

Compliance Notes: (Optional)

~~Chapter~~ Section 3, *General Requirements* defines the general rules a specific driver must follow to be compliant with an attribute. This section specifies additional compliance requirements and exceptions that apply to a particular attribute.

2.3.2 Function Section Layout

Each Function section is composed of the following subsections. Optional subsections are noted:

Description

Describes the behavior and intended use of the function.

C Prototype:

Defines the C Language prototype.

Parameters:

Describes each function parameter.

Return Values:

Defines the possible completion codes for the function.

Compliance Notes: (Optional)

~~Chapter3~~Section 3, *General Requirements* defines the general rules a specific driver must follow to be compliant with a function. This section specifies additional compliance requirements and exceptions that apply to a particular function.

3. General Requirements

This section describes the general requirements a specific 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 IviDmm Class Specification, a specific driver must implement the inherent capabilities that *IVI- 3.2: Inherent IVI Capabilities Specification* defines and the IviDmmBase capability group.

3.2 Capability Group Compliance

This section defines the general rules for a specific driver to be compliant with a capability group.

3.2.1 Capability Group Compliance Rules

To comply with a particular capability group, a specific driver must do the following:

- Implement all attributes that the capability group defines.
- Implement all functions that the capability group defines.
- Implement the behavior model that the capability group defines.
- If the capability group is an extension capability group, the group can affect the behavior of the instrument only if the application program explicitly uses the extension capability group. See the *Disabling Unused Extension Capability Groups* section for more details.

Note: If a particular capability group defines additional compliance rules or exceptions to the above rules, the additional rules and exceptions are defined in the compliance notes section for the capability group.

3.2.1.1 Disabling Unused Extension Capability Groups

It is possible that instrument features that a user neither accesses nor explicitly disables can cause non-interchangeable behavior. Such cases can occur when a user does not use an extension capability group in their application and wants the application to be interchangeable across instruments regardless of whether specific drivers for the instruments support the extension capability group. To help maximize interchangeability in such situations, the specific driver must ensure that each extension capability group does not affect instrument behavior unless an application program uses the extension capability group.

In general, a specific driver must ensure that instrument settings that correspond to a particular extension capability group do not affect the behavior of the instrument until one of the following conditions 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. This applies regardless of whether the application sets the attribute directly or through a high-level function call.

A specific driver can ensure that a particular unused extension capability group does not affect the behavior of the instrument by configuring the attributes of the extension capability group such that the extension capability group is *disabled*. The *Appendix A, Specific Driver Development Guidelines* section recommends how specific drivers can disable unused extension capability groups.

3.2.2 Attribute Compliance Rules

To comply with a particular attribute that a capability group defines, a specific driver must do the following:

- Implement the attribute with the behavior that the capability group defines for the attribute.
- If the attribute has defined values, the specific driver must support at least one of the defined values.
- If an attribute has defined values and the specific driver adds instrument-specific values for the attribute, the specific driver must define these values to be equal or greater than the base extension value that the attribute defines.
- If the attribute does not have defined values, the specific driver is required to support only the values supported by the instrument.

Note: If a particular attribute defines additional compliance rules or exceptions to the above rules, the additional rules and exceptions are defined in the compliance notes section for the attribute.

3.2.3 Function Compliance Rules

To comply with a particular function that a capability group defines, a specific driver must do the following:

- Implement the function with the behavior that the capability group defines for the function.
- If no error or warning conditions occur, the function returns `VI_SUCCESS`.
- If an error occurs, the function must return a status value that is less than zero.
- If a warning occurs and no errors occur, the function must return a status value that is greater than zero.
- If both an error and a warning occur, the function must return a status value that corresponds to the error.
- If a specific driver defines errors other than the ones defined for the function, the actual values of the instrument-specific errors must be greater than or equal to `IVI_SPECIFIC_ERROR_BASE` and less than or equal to `IVI_MAX_SPECIFIC_ERROR_CODE`.
- If a driver defines warnings other than the ones defined for the function, the actual values of the instrument-specific warnings must be greater than or equal to `IVI_SPECIFIC_WARN_BASE` and less than or equal to `IVI_MAX_SPECIFIC_WARN_CODE`.
- If the description of an input parameter specifies that the specific driver uses the parameter value to set a particular attribute, the specific driver must implement the parameter with the same compliance requirements that the specification defines for the attribute.
- If the description of an output parameter specifies that the specific driver returns the value of a

particular attribute, the specific driver must implement the parameter with the same compliance requirements that the specification defines for the attribute.

- If a function parameter has defined values, the specific driver must support at least one of the defined values.
- If a function parameter has defined values and the specific driver adds instrument-specific values for the parameter, the specific driver must define these values to be equal or greater than the base extension value that the attribute defines.
- If a function parameter does not have defined values, the specific driver is required to support only the values supported by the instrument.

Note: If a particular function defines additional compliance rules or exceptions to the above rules, the additional rules and exceptions are defined in the compliance notes section for the function.

4. IviDmmBase Capability Group

4.1 Overview

The IviDmmBase Capability Group supports DMMs that take a single measurement at a time. The IviDmmBase 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 measurement function, range, resolution, and trigger source. The IviDmmBase capability group also includes functions for configuring the DMM as well as initiating and retrieving measurements.

4.2 IviDmmBase Attributes

The IviDmmBase capability group defines the following attributes:

- `IVIDMM_ATTR_FUNCTION`
- `IVIDMM_ATTR_RANGE`
- `IVIDMM_ATTR_RESOLUTION_ABSOLUTE`
- `IVIDMM_ATTR_TRIGGER_DELAY`
- `IVIDMM_ATTR_TRIGGER_SOURCE`

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

4.2.1 IVIDMM_ATTR_FUNCTION

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureMeasurement

Description

Specifies the measurement function.

The value of this attribute determines the units for the IVIDMM_ATTR_RANGE and IVIDMM_ATTR_RESOLUTION_ABSOLUTE attributes, and the measurement values that are returned by the IviDmm_Read, IviDmm_ReadMultiPoint, IviDmm_Fetch, and IviDmm_FetchMultiPoint functions.

Defined Values

Value	Description
IVIDMM_VAL_DC_VOLTS	Sets the DMM to measure DC voltage.
IVIDMM_VAL_AC_VOLTS	Sets the DMM to measure AC voltage. Use the IviDmmACMeasurement extension group to configure additional parameters for this measurement type.
IVIDMM_VAL_DC_CURRENT	Sets the DMM to measure DC current..
IVIDMM_VAL_AC_CURRENT	Sets the DMM to measure AC current. Use the IviDmmACMeasurement extension group to configure additional parameters for this measurement type.
IVIDMM_VAL_2_WIRE_RES	Sets the DMM to measure 2-wire resistance.
IVIDMM_VAL_4_WIRE_RES	Sets the DMM to measure 4-wire resistance.
IVIDMM_VAL_AC_PLUS_DC_VOLTS	Sets the DMM to measure AC plus DC voltage. Use the IviDmmACMeasurement extension group to configure additional parameters for this measurement type.
IVIDMM_VAL_AC_PLUS_DC_CURRENT	Sets the DMM to measure AC plus DC current. Use the IviDmmACMeasurement extension group to configure additional parameters for this measurement type.
IVIDMM_VAL_FREQ	Sets the DMM to measure frequency. Use the IviDmmFrequencyMeasurement extension group to configure additional parameters for this measurement type.
IVIDMM_VAL_PERIOD	Sets the DMM to measure period. Use the IviDmmFrequencyMeasurement extension group to configure additional parameters for this measurement type.
IVIDMM_VAL_TEMPERATURE	Sets the DMM to measure temperature. Use the IviDmmTemperatureMeasurement extension group to configure additional parameters for this measurement type.

Compliance Notes

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

Value	Required Capability Group
IVIDMM_VAL_AC_VOLTS	IviDmmACMeasurement
IVIDMM_VAL_AC_CURRENT	IviDmmACMeasurement
IVIDMM_VAL_AC_PLUS_DC_VOLTS	IviDmmACMeasurement
IVIDMM_VAL_AC_PLUS_DC_CURRENT	IviDmmACMeasurement
IVIDMM_VAL_FREQ	IviDmmFrequencyMeasurement
IVIDMM_VAL_PERIOD	IviDmmFrequencyMeasurement
IVIDMM_VAL_TEMPERATURE	IviDmmTemperatureMeasurement

2. If a specific driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_FUNC_SPECIFIC_EXT_BASE.

3. If a class driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_FUNC_CLASS_EXT_BASE and less than IVIDMM_VAL_FUNC_SPECIFIC_EXT_BASE.

4.2.2 IVIDMM_ATTR_RANGE

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViReal64	R/W	No	Up	IviDmm_ConfigureMeasurement

Description

Specifies the measurement range. Positive values represent the absolute value of the maximum measurement expected. The specific driver is expected to coerce this value to the appropriate range for the instrument. Negative values represent the Auto Range mode.

There is a dependency between the IVIDMM_ATTR_RANGE attribute and the IVIDMM_ATTR_RESOLUTION_ABSOLUTE attribute. The allowed values of IVIDMM_ATTR_RESOLUTION_ABSOLUTE attribute depend on the IVIDMM_ATTR_RANGE attribute.

Typically, when the value of the IVIDMM_ATTR_RANGE attribute changes, the instrument settings that correspond to the IVIDMM_ATTR_RESOLUTION_ABSOLUTE attribute change as well. This is true regardless of how the change of measurement range occurs.

There are two possible ways that the measurement range can change. The application program can set the value of the IVIDMM_ATTR_RANGE attribute. Or, the instrument changes the measurement range because IVIDMM_ATTR_RANGE attribute is set to IVIDMM_VAL_AUTO_RANGE_ON and the input signal changes. In both cases, the instrument resolution is likely to change.

The value of the IVIDMM_ATTR_FUNCTION attribute determines the units for this attribute. The following table shows the defined values for the IVIDMM_ATTR_FUNCTION attribute and the corresponding units for the IVIDMM_ATTR_RANGE attribute.

Values for IVIDMM_ATTR_FUNCTION	Units for IVIDMM_ATTR_RANGE
IVIDMM_VAL_DC_VOLTS	Volts
IVIDMM_VAL_AC_VOLTS	Volts RMS
IVIDMM_VAL_DC_CURRENT	Amps
IVIDMM_VAL_AC_CURRENT	Amps
IVIDMM_VAL_2_WIRE_RES	Ohms
IVIDMM_VAL_4_WIRE_RES	Ohms
IVIDMM_VAL_AC_PLUS_DC_VOLTS	Volts
IVIDMM_VAL_AC_PLUS_DC_CURRENT	Amps
IVIDMM_VAL_FREQ	Hertz
IVIDMM_VAL_PERIOD	Seconds
IVIDMM_VAL_TEMPERATURE	Degrees Celsius

Defined Values

Value	Description
IVIDMM_VAL_AUTO_RANGE_ON	Sets the DMM to calculate the range before each measurement automatically. You can obtain the actual range the DMM is currently using by getting the value of the IVIDMM_ATTR_AUTO_RANGE_VALUE attribute in the IviDmmAutoRangeValue extension group. Setting this attribute to a manual range or to IVIDMM_VAL_AUTO_RANGE_OFF disables auto-ranging.
IVIDMM_VAL_AUTO_RANGE_OFF	Disables auto-ranging. The DMM sets the range to the value it most recently calculated. Further queries of this attribute return the actual range.
IVIDMM_VAL_AUTO_RANGE_ONCE	Sets the DMM to calculate the range before the next measurement. The DMM uses this range value for all subsequent measurements. Further queries of this attribute return the actual range.

Compliance Notes

1. If a specific driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_RANGE_SPECIFIC_EXT_BASE.

2. If a class driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_RANGE_CLASS_EXT_BASE and less than IVIDMM_VAL_RANGE_SPECIFIC_EXT_BASE.

4.2.3 IVIDMM_ATTR_RESOLUTION_ABSOLUTE

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViReal64	R/W	No	Down	IviDmm_ConfigureMeasurement

Description

Specifies the measurement resolution in absolute units.

The value of the IVIDMM_ATTR_FUNCTION attribute determines the units for this attribute. The following table shows the defined values for the IVIDMM_ATTR_FUNCTION attribute and the corresponding units for the IVIDMM_ATTR_RESOLUTION_ABSOLUTE attribute.

Values for IVIDMM_ATTR_FUNCTION	Units for IVIDMM_ATTR_RESOLUTION_ABSOLUTE
IVIDMM_VAL_DC_VOLTS	Volts
IVIDMM_VAL_AC_VOLTS	Volts RMS
IVIDMM_VAL_DC_CURRENT	Amps
IVIDMM_VAL_AC_CURRENT	Amps
IVIDMM_VAL_2_WIRE_RES	Ohms
IVIDMM_VAL_4_WIRE_RES	Ohms
IVIDMM_VAL_AC_PLUS_DC_VOLTS	Volts
IVIDMM_VAL_AC_PLUS_DC_CURRENT	Amps
IVIDMM_VAL_FREQ	Hertz
IVIDMM_VAL_PERIOD	Seconds
IVIDMM_VAL_TEMPERATURE	Degrees Celsius

4.2.4 IVIDMM_ATTR_TRIGGER_DELAY

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViReal64	R/W	No	Note	IviDmm_ConfigureTrigger

Note

Many DMMs have a small, non-zero value as the minimum value for this attribute. To configure the instrument to use the shortest trigger delay, the user can specify a value of zero for this attribute. Therefore, the specific driver must coerce any value between zero and the minimum value to the minimum value. No other coercion is allowed on this attribute.

Description

Specifies the length of time between when the DMM receives the trigger and when it takes a measurement. Use positive values to set the trigger delay in seconds. Negative values are reserved for the auto delay mode.

Defined Values

Value	Description
IVIDMM_VAL_AUTO_DELAY_ON	Sets the DMM to calculate the trigger delay before each measurement. Setting this attribute to a manual trigger delay or IVIDMM_VAL_AUTO_DELAY_OFF disables the auto delay mode.
IVIDMM_VAL_AUTO_DELAY_OFF	Stops the DMM from calculating the trigger delay. Sets the trigger delay to the last trigger delay the DMM calculated. Note: After the user sets this attribute to IVIDMM_VAL_AUTO_DELAY_OFF, further queries of this attribute returns the actual delay.

Compliance Notes

1. If a specific driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_DELAY_SPECIFIC_EXT_BASE

2. If a class driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_DELAY_CLASS_EXT_BASE and less than IVIDMM_VAL_TRIGGER_DELAY_SPECIFIC_EXT_BASE.

4.2.5 IVIDMM_ATTR_TRIGGER_SOURCE

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureTrigger

Description

Specifies the trigger source.

Defined Values

Value	Description
IVIDMM_VAL_IMMEDIATE	The DMM exits the Wait-For-Trigger state immediately after entering. It does not wait for a trigger of any kind.
IVIDMM_VAL_EXTERNAL	The DMM exits the Wait-For-Trigger state when a trigger occurs on the external trigger input.
IVIDMM_VAL_SOFTWAREW_TRIG~~_FUNC~~	The DMM exits the Wait-For-Trigger state when the IviDmm_SendSoftwareTrigger function executes. Refer to the Standardized Cross Class Capabilities specification for a complete description of this value and the IviDmm_SendSoftwareTrigger function.
IVIDMM_VAL_TTL0	The DMM exits the Wait-For-Trigger state when it receives a trigger on TTL0.
IVIDMM_VAL_TTL1	The DMM exits the Wait-For-Trigger state when it receives a trigger on TTL1.
IVIDMM_VAL_TTL2	The DMM exits the Wait-For-Trigger state when it receives a trigger on TTL2.
IVIDMM_VAL_TTL3	The DMM exits the Wait-For-Trigger state when it receives a trigger on TTL3.
IVIDMM_VAL_TTL4	The DMM exits the Wait-For-Trigger state when it receives a trigger on TTL4.
IVIDMM_VAL_TTL5	The DMM exits the Wait-For-Trigger state when it receives a trigger on TTL5.
IVIDMM_VAL_TTL6	The DMM exits the Wait-For-Trigger state when it receives a trigger on TTL6.
IVIDMM_VAL_TTL7	The DMM exits the Wait-For-Trigger state when it receives a trigger on TTL7.
IVIDMM_VAL_ECL0	The DMM exits the Wait-For-Trigger state when it receives a trigger on ECL0.
IVIDMM_VAL_ECL1	The DMM exits the Wait-For-Trigger state when it receives a trigger on ECL1.
IVIDMM_VAL_PXI_STAR	The DMM exits the Wait-For-Trigger state when it receives a trigger on the PXI Star trigger bus.
IVIDMM_VAL_RTSI_0	The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI0.

Value	Description
IVIDMM_VAL_RTSI_1	The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI1.
IVIDMM_VAL_RTSI_2	The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI2.
IVIDMM_VAL_RTSI_3	The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI3.
IVIDMM_VAL_RTSI_4	The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI4.
IVIDMM_VAL_RTSI_5	The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI5.
IVIDMM_VAL_RTSI_6	The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI6.

~~IVIDMM_VAL_RTSI_2~~	~~The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI2.~~
~~IVIDMM_VAL_RTSI_3~~	~~The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI3.~~
~~IVIDMM_VAL_RTSI_4~~	~~The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI4.~~
~~IVIDMM_VAL_RTSI_5~~	~~The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI5.~~
~~IVIDMM_VAL_RTSI_6~~	~~The DMM exits the Wait-For-Trigger state when it receives a trigger on RTSI6.~~

Compliance Notes

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

Value	Required Capability Group
IVIDMM_VAL_SOFTWARE_TRIG~~IVIDMM_VAL_SW_TRIG_FUNC~~	IviDmmSoftwareTrigger

2. If a specific driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE.

3. If a class driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_SOURCE_CLASS_EXT_BASE and less than IVIDMM_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE.

4.3 IviDmmBase Functions

The IviDmmBase capability group defines the following functions:

- `IviDmm_Abort`
- `IviDmm_ConfigureMeasurement`
- `IviDmm_ConfigureTrigger`
- `IviDmm_Fetch`
- `IviDmm_Initiate`
- `IviDmm_IsOverRange`
- `IviDmm_Read`

This section describes the behavior and requirements of each function.
The required function hierarchy for these functions as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

4.3.1 IviDmm_Abort

Description

This function aborts a previously initiated measurement and returns the DMM 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 `IviDmm_error_query` function at the conclusion of the sequence.

C Prototype

```
ViStatus IviDmm_Abort (ViSession vi);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`

Return Values

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

4.3.2 IviDmm_ConfigureMeasurement

Description

This function configures the common attributes of the DMM. These attributes include the measurement function, maximum range, and the resolution of the DMM.

If the value of the `range` parameter is `IVIDMM_VAL_AUTO_RANGE_ON`, then the `resolution` parameter is ignored and the `IVIDMM_ATTR_RESOLUTION_ABSOLUTE` attribute is not set.

C Prototype

```
ViStatus IviDmm_ConfigureMeasurement (ViSession vi,
                                      ViInt32 function,
                                      ViReal64 range,
                                      ViReal64 resolution);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`function`	Specifies the measurement function. The driver uses this value to set the `IVIDMM_ATTR_FUNCTION` attribute. See the attribute description for more details.	`ViInt32`
`range`	Specifies the measurement range. The driver uses this value to set the `IVIDMM_ATTR_RANGE` attribute. See the attribute description for more details.	`ViReal64`
`resolution`	Specifies the resolution. The driver uses this value to set the `IVIDMM_ATTR_RESOLUTION_ABSOLUTE` attribute. See the attribute description for more details.	`ViReal64`

Return Values

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

4.3.3 IviDmm_ConfigureTrigger

Description

This function configures the common DMM trigger attributes. These attributes include the trigger source and the trigger delay.

C Prototype

```
ViStatus IviDmm_ConfigureTrigger (ViSession vi,
                                  ViInt32 triggerSource,
                                  ViReal64 triggerDelay);
```

Parameters

Inputs	Description	Base Type
vi	Instrument handle	ViSession
triggerSource	Specifies the trigger source. The driver uses this value to set the IVIDMM_ATTR_TRIGGER_SOURCE attribute. See the attribute description for more details.	ViInt32
triggerDelay	Specifies the trigger delay. The driver uses this value to set the IVIDMM_ATTR_TRIGGER_DELAY attribute. See the attribute description for more details.	ViReal64

Return Values

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

4.3.4 IviDmm_Fetch

Description

This function returns the measured value from a measurement that the `IviDmm_Initiate` function initiates. After this function executes, the `reading` parameter contains an actual reading or a value indicating that an overrange condition occurred.

If an overrange condition occurs, the `reading` parameter contains an IEEE defined NaN (Not a Number) value and the function returns `IVIDMM_WARN_OVER_RANGE`. Test the measurement value for overrange with the `IviDmm_IsOverRange` function.

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 `IviDmm_error_query` function at the conclusion of the sequence.

In most instrument classes, there is a programmatic way to determine when a measurement has completed and data is available. Therefore, a `maxTime` parameter is not needed in the "Fetch" function for these classes. This is not true for the majority of DMMs. The `maxTime` parameter specifies how long to wait in the `IviDmm_Fetch` operation since it is possible that no data is available or the trigger event did not occur.

The measurement value has the same units as the measurement type. The units for each measurement type are defined in the defined values table of Section 4.2.1 *`IVIDMM_ATTR_FUNCTION`*.

The value of the `IVIDMM_ATTR_FUNCTION` attribute determines the units for the `reading` parameter. The following table shows the defined values for the `IVIDMM_ATTR_FUNCTION` attribute and the corresponding units for the `reading` parameter.

Values for `IVIDMM_ATTR_FUNCTION`	Units for `reading` parameter
`IVIDMM_VAL_DC_VOLTS`	Volts
`IVIDMM_VAL_AC_VOLTS`	Volts RMS
`IVIDMM_VAL_DC_CURRENT`	Amps
`IVIDMM_VAL_AC_CURRENT`	Amps
`IVIDMM_VAL_2_WIRE_RES`	Ohms
`IVIDMM_VAL_4_WIRE_RES`	Ohms
`IVIDMM_VAL_AC_PLUS_DC_VOLTS`	Volts
`IVIDMM_VAL_AC_PLUS_DC_CURRENT`	Amps
`IVIDMM_VAL_FREQ`	Hertz
`IVIDMM_VAL_PERIOD`	Seconds
`IVIDMM_VAL_TEMPERATURE`	Degrees Celsius

This function is not guaranteed to return valid data if the user performs other operations on the instrument after the call to `IviDmm_Initiate` and prior to calling this function. This includes other calls to `IviDmm_Fetch`.

C Prototype

```
ViStatus IviDmm_Fetch (ViSession vi,
                       ViInt32 maxTime,
                       ViReal64 *reading);
```

Parameters

Inputs	Description	Base Type
vi	Instrument handle	ViSession
maxTime	Specifies the maximum length of time allowed for the function to complete in milliseconds. Defined Values: IVIDMM_VAL_MAX_TIME_IMMEDIATE – The function returns immediately. If no valid measurement value exists, the function returns an error. IVIDMM_VAL_MAX_TIME_INFINITE – The function waits indefinitely for the measurement to complete.	ViInt32

Outputs	Description	Base Type
reading	Measurement value.	ViReal64

Return Values

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 Codes	Description
IVIDMM_WARN_OVER_RANGE	Overrange warning
IVIDMM_ERROR_MAX_TIME_EXCEEDED	Max time exceeded before the operation completed

Compliance Notes

~~1.~~A specific driver is not required to implement the IVIDMM_VAL_MAX_TIME_IMMEDIATE or the IVIDMM_VAL_MAX_TIME_INFINITE defined values for the maxTime parameter to be compliant with the IviDmmBase Capability group.

4.3.5 IviDmm_Initiate

Description

This function initiates a measurement. When this function executes, the DMM leaves the idle state and waits for a trigger.

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 `IviDmm_error_query` function at the conclusion of the sequence.

C Prototype

```
ViStatus IviDmm_Initiate (ViSession vi);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`

Return Values

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

4.3.6 IviDmm_IsOverRange

Description

This function takes a measurement value that you obtain from one of the Read or Fetch functions and determines if the value is a valid measurement value or a value indicating that an overrange condition occurred.

C Prototype

```
ViStatus IviDmm_IsOverRange (ViSession vi,
                             ViReal64 measurementValue,
                             ViBoolean *isOverRange);
```

Parameters

Inputs	Description	Base Type
vi	Instrument handle	ViSession
measurementValue	Pass the measurement value you obtain from one of the Read or Fetch functions.	ViReal64

Outputs	Description	Base Type
isOverRange	Returns whether the measurementValue is a valid measurement or a value indicating that the DMM encountered an overrange condition. Valid Return Values: VI_TRUE - The measurementValue indicates that an overrange condition occurred. VI_FALSE - The measurementValue is a valid measurement.	ViBoolean

Return Values

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

4.3.7 IviDmm_Read

Description

This function initiates a measurement, waits until the DMM has returned to the idle state, and returns the measured value.

After this function executes, the `reading` parameter contains an actual reading or a value indicating that an overrange condition occurred. If an overrange condition occurs, the `reading` parameter contains an IEEE defined NaN (Not a Number) value and the function returns `IVIDMM_WARN_OVER_RANGE`. Test the measurement value for overrange with the `IviDmm_IsOverRange` function.

The value of the `IVIDMM_ATTR_FUNCTION` attribute determines the units for the `reading` parameter. Refer to Section 4.3.3, ***`IviDmm_Fetch`***, for more details.

C Prototype

```
ViStatus IviDmm_Read (ViSession vi,
                      ViInt32 maxTime,
                      ViReal64 *reading);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`maxTime`	Specifies the maximum length of time allowed for the function to complete in milliseconds. Defined Values: `IVIDMM_VAL_MAX_TIME_IMMEDIATE` – The function returns immediately. If no valid measurement value exists, the function returns an error. `IVIDMM_VAL_MAX_TIME_INFINITE` – The function waits indefinitely for the measurement to complete.	`ViInt32`

Outputs	Description	Base Type
`reading`	Measurement value.	`ViReal64`

Return Values

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 Codes	Description
`IVIDMM_WARN_OVER_RANGE`	Overrange warning
`IVIDMM_ERROR_MAX_TIME_EXCEEDED`	Max time exceeded before the operation completed

Compliance Notes

1. A specific driver is not required to implement the `IVIDMM_VAL_MAX_TIME_IMMEDIATE` or the `IVIDMM_VAL_MAX_TIME_INFINITE` defined values for the `maxTime` parameter to be compliant with the IviDmmBase extension group.

4.4 IviDmmBase Behavior Model

The following behavior model shows the relationship between the IviDmmBase capability group and DMM behavior.

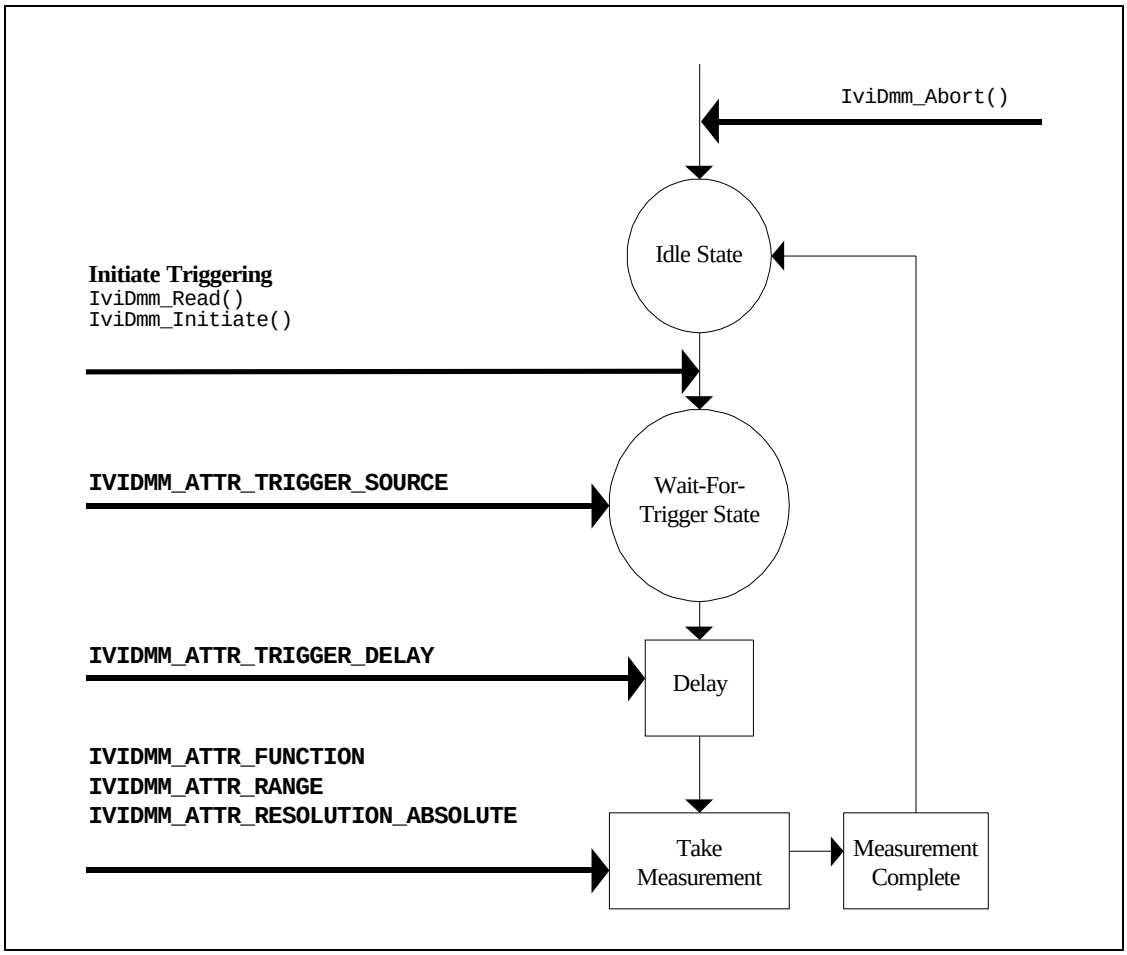


Figure 4-1. IviDmm Behavior Model

The main state in the IviDmm Class is the Idle state. The DMM 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 `IviDmm_Abort` function. Typically, the user configures the DMM while it is in the Idle state. IviDmm attributes can be configured individually with the `IviDmm_SetAttribute` function or with the high-level `IviDmm_Configure` function.

The `IviDmm_Read` and `IviDmm_Initiate` functions cause the DMM to leave the Idle state and transition to the Wait-For-Trigger state. The `IviDmm_Read` function does not return until the measurement process is complete and the DMM returns to the Idle state. The `IviDmm_Initiate` function returns as soon as the DMM leaves the Idle state.

The DMM leaves the Wait-For-Trigger state when it receives a trigger event. The type of trigger event is specified by the attribute `IVIDMM_ATTR_TRIGGER_SOURCE`.

After the specified trigger event occurs, the DMM waits the amount of time specified by the attribute `IVIDMM_ATTR_TRIGGER_DELAY` and then takes a measurement. The type of measurement is specified by

the attributes IVIDMM_ATTR_FUNCTION, IVIDMM_ATTR_RANGE, and IVIDMM_ATTR_RESOLUTION_ABSOLUTE.

If the IVIDMM_ATTR_FUNCTION attribute is set to a value that requires an extension capability group, the attributes of that capability group further configure the measurement.

After the measurement is taken, the DMM (if it is capable of doing so) generates the Measurement Complete signal and returns to the Idle state.

The IviDmmBase capability group does not require that a DMM be able to generate a Measurement Complete signal. The IviDmmMultiPoint capability group defines how the Measurement Complete signal is configured. The Measurement Complete signal is presented in the IviDmm behavior model diagram to define when the signal is generated as most DMMs generate this signal but may not be able to configure it.

The IviDmm_Fetch function is used to retrieve measurements that were initiated by the IviDmm_Initiate function. The measurement data returned from the IviDmm_Read and IviDmm_Fetch functions is acquired after the the DMM has left the Wait-For-Trigger state.

5. IviDmmACMeasurement Extension Group

5.1 IviDmmACMeasurement Extension Group Overview

The IviDmmACMeasurement extension group supports DMMs that take AC voltage or AC current measurements. It defines attributes that configure additional settings for AC measurements. These attributes are the minimum and maximum frequency components of the input signal. This extension group also defines functions that configure these attributes.

5.2 IviDmmACMeasurement Attributes

The IviDmmACMeasurement extension group defines the following attributes:

- `IVIDMM_ATTR_AC_MAX_FREQ`
- `IVIDMM_ATTR_AC_MIN_FREQ`

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

5.2.1 IVIDMM_ATTR_AC_MAX_FREQ

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViReal64	R/W	No	Up	IviDmm_ConfigureACBandwidth

Description

Specifies the maximum frequency component of the input signal for AC measurements. The value of this attribute affects instrument behavior only when the IVIDMM_ATTR_FUNCTION attribute is set to an AC voltage or AC current measurement.

5.2.2 IVIDMM_ATTR_AC_MIN_FREQ

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViReal64	R/W	No	Down	IviDmm_ConfigureACBandwidth

Description

Specifies the minimum frequency component of the input signal for AC measurements. The value of this attribute affects instrument behavior only when the IVIDMM_ATTR_FUNCTION attribute is set to an AC voltage or AC current measurement.

5.3 IviDmmACMeasurement Functions

The IviDmmACMeasurement extension group defines the following function:

- `IviDmm_ConfigureACBandwidth`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

5.3.1 IviDmm_ConfigureACBandwidth

Description

This function configures additional parameters for DMMs that take AC voltage or AC current measurements. These attributes are the AC minimum and maximum frequency.

C Prototype

```
ViStatus IviDmm_ConfigureACBandwidth (ViSession vi,
                                      ViReal64 minFreq,
                                      ViReal64 maxFreq);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`minFreq`	Specifies the AC minimum frequency. The driver uses this value to set the `IVIDMM_ATTR_AC_MIN_FREQ` attribute. See the attribute description for more details.	`ViReal64`
`maxFreq`	Specifies the AC maximum frequency. The driver uses this value to set the `IVIDMM_ATTR_AC_MAX_FREQ` attribute. See the attribute description for more details.	`ViReal64`

Return Values

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

5.4 IviDmmACMeasurement Behavior Model

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

5.5 IviDmmACMeasurement Compliance Notes

1. Specific drivers that implement this extension group must implement at least one of the following values for the `IVIDMM_ATTR_FUNCTION` attribute in the IviDmmBase capability group:

   - `IVIDMM_VAL_AC_VOLTS`
   - `IVIDMM_VAL_AC_CURRENT`
   - `IVIDMM_AC_PLUS_DC_VOLTS`
   - `IVIDMM_AC_PLUS_DC_VOLTS`

6. IviDmmFrequencyMeasurement Extension Group

6.1 IviDmmFrequencyMeasurement Extension Group Overview

The IviDmmFrequencyMeasurement extension group supports DMMs that take frequency measurements. It defines attributes that are required to configure additional parameters needed for frequency measurements.

6.2 IviDmmFrequencyMeasurement Attributes

The IviDmmFrequencyMeasurement extension group defines the following attribute:

- `IVIDMM_ATTR_FREQ_VOLTAGE_RANGE`

This section describes the behavior and requirements of this attribute. The actual value for this attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

6.2.1 IVIDMM_ATTR_FREQ_VOLTAGE_RANGE

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViReal64	R/W	No	Up	IviDmm_ConfigureFrequencyVoltageRange

Description

Specifies the expected maximum value of the input signal for frequency and period measurements. Positive values represent the manual range. Negative values represent the Auto Range mode.

The value of this attribute affects instrument behavior only when the IVIDMM_ATTR_FUNCTION attribute is set to a frequency or period measurement.

The units are specified in Volts RMS.

Defined Values

Value	Description
IVIDMM_VAL_AUTO_RANGE_ON	Sets the DMM to calculate the frequency voltage range before each measurement automatically. Setting this attribute to a manual range or to IVIDMM_VAL_AUTO_RANGE_OFF disables auto-ranging.
IVIDMM_VAL_AUTO_RANGE_OFF	Disables auto-ranging. Further queries of this attribute return the actual frequency voltage range.

Compliance Notes

1. If a specific driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_FREQ_VOLT_RANGE_SPECIFIC_EXT_BASE.

2. If a class driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_FREQ_VOLT_RANGE_CLASS_EXT_BASE and less than IVIDMM_VAL_FREQ_VOLT_RANGE_SPECIFIC_EXT_BASE.

6.3 IviDmmFrequencyMeasurement Functions

The IviDmmFrequencyMeasurement extension group defines the following function:

- `IviDmm_ConfigureFrequencyVoltageRange`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

6.3.1 IviDmm_ConfigureFrequencyVoltageRange

Description

This function configures the frequency voltage range of the DMM.

C Prototype

```
ViStatus IviDmm_ConfigureFrequencyVoltageRange (ViSession vi,
                                                ViReal64
frequencyVoltageRange);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`frequencyVoltageRange`	Specifies the expected maximum value of the input signal. The driver uses this value to set the `IVIDMM_ATTR_FREQ_VOLTAGE_RANGE` attribute. See the attribute description for more details.	`ViReal64`

Return Values

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

6.4 IviDmmFrequencyMeasurement Behavior Model

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

6.5 IviDmmFrequencyMeasurement Compliance Notes

1. Specific drivers that implement this extension group must implement at least one of the following values for the `IVIDMM_ATTR_FUNCTION` attribute in the IviDmmBase capability group:

   - `IVIDMM_VAL_FREQ`
   - `IVIDMM_VAL_PERIOD`

7. IviDmmTemperatureMeasurement Extension Group

7.1 IviDmmTemperatureMeasurement Extension Group Overview

The IviDmmTemperatureMeasurement extension group supports DMMs that take temperature measurements with a thermocouple, an RTD, or a thermistor transducer type. This extension group selects the transducer type. Other capability groups further configure temperature settings based on the transducer type.

7.2 IviDmmTemperatureMeasurement Attributes

The IviDmmTemperatureMeasurement extension group defines the following attribute:

- `IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE`

This section describes the behavior and requirements of this attribute. The actual value for the attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

7.2.1 IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureTransducerType

Description

Specifies the device used to measure the temperature. The value of this attribute affects instrument behavior only when the IVIDMM_ATTR_FUNCTION attribute is set to a temperature measurement.

Defined Values

Value	Description
IVIDMM_VAL_THERMOCOUPLE	Sets the DMM to measure temperature using a thermocouple. Use the IviDmmThermocouple extension group to configure additional settings for this transducer type.
IVIDMM_VAL_THERMISTOR	Sets the DMM to measure temperature using a thermistor. Use the IviDmmThermistor extension group to configure additional settings for this transducer type.
IVIDMM_VAL_2_WIRE_RTD	Sets the DMM to measure temperature using a 2-wire temperature resistance device. Use the IviDmmResistanceTemperatureDevice extension group to configure additional settings for this transducer type.
IVIDMM_VAL_4_WIRE_RTD	Sets the DMM to measure temperature using a 4-wire temperature resistance device. Use the IviDmmResistanceTemperatureDevice extension group to configure additional settings for this transducer type.

Compliance Notes

~~1.~~If a specific driver implements any of the defined values in the following table, it must also implement the corresponding capability group:

Value	Required Capability Group
IVIDMM_VAL_THERMOCOUPLE	IviDmmThermocouple
IVIDMM_VAL_THERMISTOR	IviDmmThermistor
IVIDMM_VAL_2_WIRE_RTD	IviDmmResistanceTemperatureDevice
IVIDMM_VAL_4_WIRE_RTD	IviDmmResistanceTemperatureDevice

~~2.~~1. If a specific driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_TRANSDUCER_SPECIFIC_EXT_BASE.

~~3.~~2. If a class driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_TRANSDUCER_CLASS_EXT_BASE and less than IVIDMM_VAL_TRANSDUCER_SPECIFIC_EXT_BASE.

7.3 IviDmmTemperatureMeasurement Functions

The IviDmmTemperatureMeasurement extension group defines the following function:

- `IviDmm_ConfigureTransducerType`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

7.3.1 IviDmm_ConfigureTransducerType

Description

This function configures the DMM to take temperature measurements from a specified transducer type.

C Prototype

```
ViStatus IviDmm_ConfigureTransducerType (ViSession vi,
                                         ViInt32 transducerType);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`transducerType`	Specifies the device used to measure the temperature. The driver uses this value to set the `IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE` attribute. See the attribute description for more details.	`ViInt32`

Return Values

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

7.4 IviDmmTemperatureMeasurement Behavior Model

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

7.5 IviDmmTemperatureMeasurement Compliance Notes

1. Specific drivers that implement this extension group must implement the `IVIDMM_VAL_TEMPERATURE` value for the `IVIDMM_ATTR_FUNCTION` attribute in the IviDmmBase capability group.

8. IviDmmThermocouple Extension Group

8.1 IviDmmThermocouple Extension Group Overview

The IviDmmThermocouple extension group supports DMMs that take temperature measurements using a thermocouple transducer type.

8.2 IviDmmThermocouple Attributes

The IviDmmThermocouple extension group defines the following attributes:

- `IVIDMM_ATTR_TEMP_TC_FIXED_REF_JUNC`
- `IVIDMM_ATTR_TEMP_TC_REF_JUNC_TYPE`
- `IVIDMM_ATTR_TEMP_TC_TYPE`

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

8.2.1 IVIDMM_ATTR_TEMP_TC_FIXED_REF_JUNC

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViReal64	R/W	No	None	IviDmm_ConfigureFixedRefJunction

Description

Specifies the external reference junction temperature when a fixed reference junction type thermocouple is used to take the temperature measurement. The temperature is specified in degrees Celsius.

This attribute may also be used to specify the thermocouple junction temperature of an instrument that does not have an internal temperature sensor.

The value of this attribute affects instrument behavior only when the IVIDMM_ATTR_TEMP_TC_REF_JUNC_TYPE is set to IVIDMM_VAL_TEMP_REF_JUNC_FIXED.

Compliance Notes

~~1.~~Specific drivers that implement this attribute must implement the IVIDMM_VAL_TEMP_REF_JUNC_FIXED defined value for the IVIDMM_ATTR_TEMP_TC_REF_JUNC_TYPE attribute.

8.2.2 IVIDMM_ATTR_TEMP_TC_REF_JUNC_TYPE

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureThermocouple

Description

Specifies the type of reference junction to be used in the reference junction compensation of a thermocouple measurement. The value of this attribute affects instrument behavior only when the IVIDMM_VAL_TEMP_TRANSDUCER_TYPE is set to IVIDMM_VAL_THERMOCOUPLE.

Defined Values

Value	Description
IVIDMM_VAL_TEMP_REF_JUNC_INTERNAL	Sets the DMM to use an internal sensor at the thermocouple junction for the junction compensation.
IVIDMM_VAL_TEMP_REF_JUNC_FIXED	Sets the DMM to use a fixed value for the thermocouple junction compensation. Use the IVIDMM_ATTR_TEMP_TC_FIXED_REF_JUNC attribute to set the fixed reference junction value.

Compliance Notes

~~1.~~If a specific driver implements the IVIDMM_VAL_TEMP_REF_JUNC_FIXED defined value, then it must implement the IVIDMM_ATTR_TEMP_TC_FIXED_REF_JUN attribute.

~~2.~~1. If a specific driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_REF_JUNC_SPECIFIC_EXT_BASE.

~~3.~~2. If a class driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_REF_JUNC_CLASS_EXT_BASE and less than IVIDMM_VAL_REF_JUNC_SPECIFIC_EXT_BASE.

8.2.3 IVIDMM_ATTR_TEMP_TC_TYPE

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureThermocouple

Description

Specifies the type of thermocouple used to measure the temperature. The value of this attribute affects instrument behavior only when the IVIDMM_VAL_TEMP_TRANSDUCER_TYPE is set to IVIDMM_VAL_THERMOCOUPLE.

Defined Values

Value	Description
IVIDMM_VAL_TEMP_TC_B	Sets the DMM to measure temperature from a B-type thermocouple.
IVIDMM_VAL_TEMP_TC_C	Sets the DM7M to measure temperature from a C-type thermocouple.
IVIDMM_VAL_TEMP_TC_D	Sets the DMM to measure temperature from a D-type thermocouple.
IVIDMM_VAL_TEMP_TC_E	Sets the DMM to measure temperature from an E-type thermocouple.
IVIDMM_VAL_TEMP_TC_G	Sets the DMM to measure temperature from a G-type thermocouple.
IVIDMM_VAL_TEMP_TC_J	Sets the DMM to measure temperature from a J-type thermocouple.
IVIDMM_VAL_TEMP_TC_K	Sets the DMM to measure temperature from a K-type thermocouple.
IVIDMM_VAL_TEMP_TC_N	Sets the DMM to measure temperature from an N-type thermocouple.
IVIDMM_VAL_TEMP_TC_R	Sets the DMM to measure temperature from an R-type thermocouple.
IVIDMM_VAL_TEMP_TC_S	Sets the DMM to measure temperature from an S-type thermocouple.
IVIDMM_VAL_TEMP_TC_T	Sets the DMM to measure temperature from a T-type thermocouple.
IVIDMM_VAL_TEMP_TC_U	Sets the DMM to measure temperature from a U-type thermocouple.
IVIDMM_VAL_TEMP_TC_V	Sets the DMM to measure temperature from a V-type thermocouple.

Compliance Notes

1. If a specific driver defines additional values for this attribute, the actual values must be greater than or equal to `IVIDMM_VAL_TEMP_TC_TYPE_SPECIFIC_EXT_BASE`.

2. If a class driver defines additional values for this attribute, the actual values must be greater than or equal to `IVIDMM_VAL_TEMP_TC_TYPE_CLASS_EXT_BASE` and less than `IVIDMM_VAL_TEMP_TC_TYPE_SPECIFIC_EXT_BASE`.

8.3 IviDmmThermocouple Functions

The IviDmmTemperatureMeasurement extension group defines the following functions:

- `IviDmm_ConfigureFixedRefJunction`
- `IviDmm_ConfigureThermocouple`

This section describes the behavior and requirements of each function.
The required function hierarchy for these functions as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

8.3.1 IviDmm_ConfigureFixedRefJunction

Description

This function configures the fixed reference junction for a thermocouple with a fixed reference junction type.

C Prototype

```
ViStatus IviDmm_ConfigureFixedRefJunction (ViSession vi,
                                           ViReal64 fixedRefJunction);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`fixedRefJunction`	Specifies the fixed reference junction. The driver uses this value to set the `IVIDMM_ATTR_TEMP_TC_FIXED_REF_JUNC` attribute. See the attribute description for more details.	`ViReal64`

Return Values

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

Compliance Notes

~~1.~~The specific driver is required to implement this function only if the `IVIDMM_ATTR_TEMP_TC_REF_JUNC_TYPE` attribute implements the `IVIDMM_VAL_TEMP_REF_JUNC_FIXED` defined value.

8.3.2 IviDmm_ConfigureThermocouple

Description

This function configures the thermocouple type and the reference junction type of the thermocouple for DMMs that take temperature measurements using a thermocouple transducer type.

C Prototype

```
ViStatus IviDmm_ConfigureThermocouple (ViSession vi,
                                       ViInt32 thermocoupleType
                                       ViInt32 refJunctionType);
```

Parameters

Inputs	Description	Base Type
vi	Instrument handle	ViSession
thermocoupleType	Specifies the type of thermocouple used to measure the temperature. The driver uses this value to set the IVIDMM_ATTR_TEMP_TC_TYPE attribute. See the attribute description for more details.	ViInt32
refJunctionType	Specifies the type of reference junction to be used. The driver uses this value to set the IVIDMM_ATTR_TEMP_TC_REF_JUNC_TYPE attribute. See the attribute description for more details.	ViInt32

Return Values

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

8.4 IviDmmThermocouple Behavior Model

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

8.5 IviDmmThermocouple Compliance Notes

1. Specific drivers that implement this extension group must implement the IviDmmTemperatureMeasurement extension group.

2. Specific drivers that implement this extension group must implement the `IVIDMM_VAL_THERMOCOUPLE` value for the `IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE` attribute in the IviDmmTemperatureMeasurement extension group.

9. IviDmmResistanceTemperatureDevice Extension Group

9.1 IviDmmResistanceTemperatureDevice Extension Group Overview

The IviDmmResistanceTemperatureDevice extension group supports DMMs that take temperature measurements using a resistance temperature device (RTD) transducer type.

The IviDmm class assumes that you are using a Platinum Resistance Temperature Device.

9.2 IviDmmResistanceTemperatureDevice Attributes

The IviDmmResistanceTemperatureDevice extension group defines the following attributes:

- `IVIDMM_ATTR_TEMP_RTD_ALPHA`
- `IVIDMM_ATTR_TEMP_RTD_RES`

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions*.

9.2.1 IVIDMM_ATTR_TEMP_RTD_ALPHA

Data Type	Access	Channel-Based	Coercion	High Level Functions
`ViReal64`	`R/W`	No	None	`IviDmm_ConfigureRTD`

Description

Specifies the alpha parameter for a resistance temperature device (RTD).

The value of this attribute affects instrument behavior only when the `IVIDMM_VAL_TEMP_TRANSDUCER_TYPE` is set to the `IVIDMM_VAL_2_WIRE_RTD` or the `IVIDMM_VAL_4_WIRE_RTD` defined values.

9.2.2 IVIDMM_ATTR_TEMP_RTD_RES

Data Type	Access	Channel-Based	Coercion	High Level Functions
`ViReal64`	`R/W`	No	None	`IviDmm_ConfigureRTD`

Description

Specifies the R_0 parameter (resistance) for a resistance temperature device (RTD). The RTD resistance is also known as the RTD reference value.

The value of this attribute affects instrument behavior only when the `IVIDMM_VAL_TEMP_TRANSDUCER_TYPE` is set to the `IVIDMM_VAL_RTD` defined value.

9.3 IviDmmResistanceTemperatureDevice Functions

The IviDmmResistanceTemperatureDevice extension group defines the following function:

- `IviDmm_ConfigureRTD`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

9.3.1 IviDmm_ConfigureRTD

Description

This function configures the alpha and resistance parameters for a resistance temperature device.

C Prototype

```
ViStatus IviDmm_ConfigureRTD (ViSession vi,
                              ViReal64 alpha
                              ViReal64 resistance);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`alpha`	Specifies the alpha parameter for an RTD. The driver uses this value to set the `IVIDMM_ATTR_TEMP_RTD_ALPHA` attribute. See the attribute description for more details.	`ViReal64`
`resistance`	Specifies the resistance of the RTD. The driver uses this value to set the `IVIDMM_ATTR_TEMP_RTD_RES` attribute. See the attribute description for more details.	`ViReal64`

Return Values

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

9.4 IviDmmResistanceTemperatureDevice Behavior Model

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

9.5 IviDmmResistanceTemperatureDevice Compliance Notes

1. Specific drivers that implement this extension group must implement the IviDmmTemperatureMeasurement extension group.

2. Specific drivers that implement this extension group must implement at least one of the following values for the `IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE` attribute in the IviDmmTemperatureMeasurement extension group:

   - `IVIDMM_VAL_2_WIRE_RTD`
   - `IVIDMM_VAL_4_WIRE_RTD`

10. IviDmmThermistor Extension Group

10.1 IviDmmThermistor Extension Group Overview

The IviDmmThermistor extension group supports DMMs that take temperature measurements using a thermistor transducer type.

The IviDmm class assumes that you are using an interchangeable thermistor. Interchangeable thermistors are thermistors that exhibit similar behavior for a given resistance value.

10.2 IviDmmThermistor Attributes

The IviDmmThermistor extension group defines the following attribute:

- `IVIDMM_ATTR_TEMP_THERMISTOR_RES`

This section describes the behavior and requirements of this attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

10.2.1 IVIDMM_ATTR_TEMP_THERMISTOR_RES

Data Type	Access	Channel-Based	Coercion	High Level Functions
`ViReal64`	`R/W`	No	None	`IviDmm_ConfigureThermistor`

Description

Specifies the resistance of the thermistor in Ohms.

The value of this attribute affects instrument behavior only when the `IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE` attribute is set to the `IVIDMM_VAL_THERMISTOR` defined value.

10.3 IviDmmThermistor Functions

The IviDmmThermistor extension group defines the following function:

- `IviDmm_ConfigureThermistor`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

10.3.1 IviDmm_ConfigureThermistor

Description

This function configures the resistance for a thermistor temperature measurement device.

C Prototype

```
ViStatus IviDmm_ConfigureThermistor (ViSession vi,
                                     ViReal64 resistance);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`resistance`	Specifies the resistance of the thermistor. The driver uses this value to set the `IVIDMM_ATTR_TEMP_THERMISTOR_RES` attribute. See the attribute description for more details.	`ViReal64`

Return Values

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

10.4 IviDmmThermistor Behavior Model

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

10.5 IviDmmThermistor Compliance Notes

1. Specific drivers that implement this extension group must implement the IviDmmTemperatureMeasurement extension group.

2. Specific drivers that implement this extension group must implement the `IVIDMM_VAL_THERMISTOR` value for the `IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE` attribute in the IviDmmTemperatureMeasurement capability group.

11. IviDmmMultiPoint Extension Group

11.1 IviDmmMultiPoint Extension Group Overview

The IviDmmMultiPoint extension group defines extensions for DMMs capable of acquiring measurements based on multiple triggers, and acquiring multiple measurements for each trigger.

The IviDmmMultiPoint extension group defines additional attributes such sample count, sample trigger, trigger count, and trigger delay to control “multi-point” DMMs. The IviDmmMultiPoint extension group also adds functions for configuring the DMM as well as starting acquisitions and retrieving multiple measured values.

11.2 IviDmmMultiPoint Attributes

The IviDmmBase capability group defines the following attributes:

- `IVIDMM_ATTR_MEAS_COMPLETE_DEST`
- `IVIDMM_ATTR_SAMPLE_COUNT`
- `IVIDMM_ATTR_SAMPLE_INTERVAL`
- `IVIDMM_ATTR_SAMPLE_TRIGGER`
- `IVIDMM_ATTR_TRIGGER_COUNT`

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions*.

11.2.1 IVIDMM_ATTR_MEAS_COMPLETE_DEST

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureMeasCompleteDest

Description

After each measurement, the DMM generates a measurement-complete signal. This attribute specifies the destination of the measurement-complete signal. This signal is commonly referred to as Voltmeter Complete.

Defined Values

Value	Description
IVIDMM_VAL_NONE	The measurement complete signal is not routed.
IVIDMM_VAL_EXTERNAL	Routes the measurement complete signal to the external connector.
IVIDMM_VAL_TTL0	Routes the measurement complete signal to TTL0.
IVIDMM_VAL_TTL1	Routes the measurement complete signal to TTL1.
IVIDMM_VAL_TTL2	Routes the measurement complete signal to TTL2.
IVIDMM_VAL_TTL3	Routes the measurement complete signal to TTL3.
IVIDMM_VAL_TTL4	Routes the measurement complete signal to TTL4.
IVIDMM_VAL_TTL5	Routes the measurement complete signal to TTL5.
IVIDMM_VAL_TTL6	Routes the measurement complete signal to TTL6.
IVIDMM_VAL_TTL7	Routes the measurement complete signal to TTL7.
IVIDMM_VAL_ECL0	Routes the measurement complete signal to ECL0.
IVIDMM_VAL_ECL1	Routes the measurement complete signal to ECL1.
IVIDMM_VAL_PXI_STAR	Routes the measurement complete signal to the PXI Star trigger bus.
IVIDMM_VAL_RTSI_0	Routes the measurement complete signal to RTSI0.
IVIDMM_VAL_RTSI_1	Routes the measurement complete signal to RTSI1.
IVIDMM_VAL_RTSI_2	Routes the measurement complete signal to RTSI2.
IVIDMM_VAL_RTSI_3	Routes the measurement complete signal to RTSI3.
IVIDMM_VAL_RTSI_4	Routes the measurement complete signal to RTSI4.
IVIDMM_VAL_RTSI_5	Routes the measurement complete signal to RTSI5.
IVIDMM_VAL_RTSI_6	Routes the measurement complete signal to RTSI6.

Compliance Notes

1. If a specific driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE.

2. If a class driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_SOURCE_CLASS_EXT_BASE and less than IVIDMM_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE.

11.2.2 IVIDMM_ATTR_SAMPLE_COUNT

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureMultiPoint

Description

Specifies the number of measurements the DMM takes each time it receives a trigger.

11.2.3 IVIDMM_ATTR_SAMPLE_INTERVAL

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViReal64	R/W	No	Up	IviDmm_ConfigureMultiPoint

Description

Specifies the interval between samples in seconds. This attribute affects instrument behavior only when IVIDMM_ATTR_SAMPLE_COUNT is greater than 1 and IVIDMM_ATTR_SAMPLE_TRIGGER is IVIDMM_VAL_INTERVAL.

Compliance Notes

1. Specific drivers that implement this attribute must implement the IVIDMM_VAL_INTERVAL value for the IVIDMM_ATTR_SAMPLE_TRIGGER attribute.

11.2.4 IVIDMM_ATTR_SAMPLE_TRIGGER

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureMultiPoint

Description

Specifies the sample trigger source. If the value of the IVIDMM_ATTR_SAMPLE_COUNT is greater than 1, the DMM enters the Wait-For-Sample-Trigger state after taking a single measurement. When the event specified by this attribute occurs, the DMM exits the Wait-For-Sample-Trigger state and takes the next measurement.

This attribute affects instrument behavior only when IVIDMM_ATTR_SAMPLE_COUNT is greater than 1.

Defined Values

Value	Description
IVIDMM_VAL_IMMEDIATE	The DMM exits the Wait-For-Sample-Trigger state immediately after entering. It does not wait for a trigger of any kind.
IVIDMM_VAL_EXTERNAL	The DMM exits the Wait-For-Sample-Trigger state when a trigger occurs on the external trigger input.
IVIDMM_VAL_SOFTWARE_TRIG~~IVIDMM_VAL_SW_TRIG_FUNC~~	The DMM exits the Wait-For-Sample-Trigger state when the IviDmm_SendSoftwareTrigger ~~IviDmm_SendSWTrigger()~~ function executes. Refer to the Standardized Cross Class Capabilities specification for a complete description of this value and the IviDmm_SendSoftwareTrigger function.
IVIDMM_VAL_INTERVAL	The DMM exits the Wait-For-Sample-Trigger state when the length of time specified by the IVIDMM_ATTR_SAMPLE_INTERVAL attribute elapses.
IVIDMM_VAL_TTL0	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on TTL0.
IVIDMM_VAL_TTL1	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on TTL1.
IVIDMM_VAL_TTL2	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on TTL2.
IVIDMM_VAL_TTL3	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on TTL3.
IVIDMM_VAL_TTL4	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on TTL4.
IVIDMM_VAL_TTL5	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on TTL5.
IVIDMM_VAL_TTL6	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on TTL6.
IVIDMM_VAL_TTL7	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on TTL7.

Value	Description
IVIDMM_VAL_ECL0	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on ECL0.
IVIDMM_VAL_ECL1	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on ECL1.
IVIDMM_VAL_PXI_STAR	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on the PXI Star trigger bus.
IVIDMM_VAL_RTSI_0	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on RTSI0.
IVIDMM_VAL_RTSI_1	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on RTSI1.
IVIDMM_VAL_RTSI_2	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on RTSI2.
IVIDMM_VAL_RTSI_3	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on RTSI3.
IVIDMM_VAL_RTSI_4	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on RTSI4.
IVIDMM_VAL_RTSI_5	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on RTSI5.
IVIDMM_VAL_RTSI_6	The DMM exits the Wait-For-Sample-Trigger state when it receives a trigger on RTSI6.

Compliance Notes

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

Value	Required Capability Group
IVIDMM_VAL_SOFTWARE_TRIG~~IVIDMM_VAL_SW_TRIG_FUNC~~	IviDmmSoftwareTrigger

2. If the specific driver implements the IVIDMM_VAL_INTERVAL defined value, then it must also implement the IVIDMM_ATTR_SAMPLE_INTERVAL attribute.

3. If a specific driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE.

4. If a class driver defines additional values for this attribute, the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_SOURCE_CLASS_EXT_BASE and less than IVIDMM_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE.

11.2.5 IVIDMM_ATTR_TRIGGER_COUNT

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureMultiPoint

Description

Specifies the number of triggers the DMM accepts before it returns to the idle state.

11.3 IviDmmMultiPoint Functions

The IviDmmMultiPoint extension group defines the following functions:

- `IviDmm_ConfigureMeasCompleteDest`
- `IviDmm_ConfigureMultiPoint`
- `IviDmm_FetchMultiPoint`
- `IviDmm_ReadMultiPoint`

This section describes the behavior and requirements of each function.
The required function hierarchy for these functions as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

11.3.1 IviDmm_ConfigureMeasCompleteDest

Description

This function configures the destination of the measurement-complete signal. This signal is commonly referred to as Voltmeter Complete.

C Prototype

```
ViStatus IviDmm_ConfigureMeasCompleteDest (ViSession vi,
                                           ViInt32 measCompleteDest);
```

Parameters

Inputs	Description	Base Type
vi	Instrument handle	ViSession
measCompleteDest	Specifies the measurement complete destination. The driver uses this value to set the IVIDMM_ATTR_MEAS_COMPLETE_DEST attribute. See the attribute description for more details.	ViInt32

Return Values

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

11.3.2 IviDmm_ConfigureMultiPoint

Description

This function configures the attributes for multi-point measurements. These attributes include the trigger count, sample count, sample trigger and sample interval.

If the value of the `sampleCount` parameter is 1, then the `sampleTrigger` and `sampleInterval` parameters are ignored and are not used to set the `IVIDMM_ATTR_SAMPLE_TRIGGER` and `IVIDMM_ATTR_SAMPLE_INTERVAL` attributes.

If the value of the `sampleTrigger` parameter is not `IVIDMM_VAL_INTERVAL`, then the `sampleInterval` parameter is ignored and is not used to set the `IVIDMM_ATTR_SAMPLE_INTERVAL` attribute.

C Prototype

```
ViStatus IviDmm_ConfigureMultiPoint (ViSession vi,
                                     ViInt32 triggerCount,
                                     ViInt32 sampleCount,
                                     ViInt32 sampleTrigger,
                                     ViReal64 sampleInterval);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`triggerCount`	Specifies the trigger count. The driver uses this value to set the `IVIDMM_ATTR_TRIGGER_COUNT` attribute. See the attribute description for more details.	`ViInt32`
`sampleCount`	Specifies the sample count. The driver uses this value to set the `IVIDMM_ATTR_SAMPLE_COUNT` attribute. See the attribute description for more details	`ViInt32`
`sampleTrigger`	Specifies the sample trigger source. The driver uses this value to set the `IVIDMM_ATTR_SAMPLE_TRIGGER` attribute. See the attribute description for more details	`ViInt32`
`sampleInterval`	Specifies the sample interval. The driver uses this value to set the `IVIDMM_ATTR_SAMPLE_INTERVAL` attribute. See the attribute description for more details	`ViReal64`

Return Values

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

11.3.3 IviDmm_FetchMultiPoint

Description

This function returns an array of values from a measurement that the `IviDmm_Initiate` function initiates. After this function executes, each element in the `readingArray` parameter is an actual reading or a value indicating that an overrange condition occurred. The number of measurements the DMM takes is determined by the Trigger Count and Sample Count.

If an overrange condition occurs, the corresponding `readingArray` element contains an IEEE defined NaN (Not a Number) value and the function returns `IVIDMM_WARN_OVER_RANGE`. Test each element in the `readingArray` parameter for overrange with the `IviDmm_IsOverRange` function.

In most instrument classes, there is a programmatic way to determine when a measurement has completed and data is available. Therefore, a `maxTime` parameter is not needed in the Fetch function for these classes. This is not true for the majority of DMMs. The `maxTime` parameter specifies how long to wait in the `IviDmm_FetchMultiPoint()` operation since it is possible that no data is available or the trigger event did not occur.

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 `IviDmm_error_query` function at the conclusion of the sequence.

The value of the `IVIDMM_ATTR_FUNCTION` attribute determines the units for each element of the `readingArray` parameter. The following table shows the defined values for the `IVIDMM_ATTR_FUNCTION` attribute and the corresponding units for each element of the `readingArray` parameter.

Values for `IVIDMM_ATTR_FUNCTION`	Units for `readingArray` parameter
`IVIDMM_VAL_DC_VOLTS`	Volts
`IVIDMM_VAL_AC_VOLTS`	Volts
`IVIDMM_VAL_DC_CURRENT`	Amps
`IVIDMM_VAL_AC_CURRENT`	Amps
`IVIDMM_VAL_2_WIRE_RES`	Ohms
`IVIDMM_VAL_4_WIRE_RES`	Ohms
`IVIDMM_VAL_AC_PLUS_DC_VOLTS`	Volts
`IVIDMM_VAL_AC_PLUS_DC_CURRENT`	Amps
`IVIDMM_VAL_FREQ`	Hertz
`IVIDMM_VAL_PERIOD`	Hertz
`IVIDMM_VAL_TEMPERATURE`	Degrees Celsius

This function is not guaranteed to return valid data if the user performs other operations on the instrument after the call to `IviDmm_Initiate` and prior to calling this function. This includes other calls to `IviDmm_FetchMultiPoint`.

C Prototype

```
ViStatus IviDmm_FetchMultiPoint (ViSession vi,
                                 ViInt32 maxTime,
                                 ViInt32 arraySize,
                                 ViReal64 readingArray[],
                                 ViInt32 *actualPts);
```

Parameters

Inputs	Description	Base Type
vi	Instrument handle	ViSession
maxTime	Specifies the maximum length of time allowed for the function to complete in milliseconds. Defined Values: IVIDMM_VAL_MAX_TIME_IMMEDIATE – The function returns immediately. If no valid measurement value exists, the function returns an error. IVIDMM_VAL_MAX_TIME_INFINITE – The function waits indefinitely for the measurement to complete.	ViInt32
arraySize	Specifies the number of elements in the readingArray.	ViInt32

Outputs	Description	Base Type
readingArray	Array of measured values. This array must be allocated by the user.	ViReal64[]
actualPts	The number of measured values placed in readingArray.	ViInt32

Return Values

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 Codes	Description
IVIDMM_WARN_OVER_RANGE	Overrange warning
IVIDMM_ERROR_MAX_TIME_EXCEEDED	Max time exceeded before the operation completed

11.3.4 IviDmm_ReadMultiPoint

Description

This function initiates the measurement, waits for the DMM to return to the Idle state, and returns an array of measured values. The number of measurements the DMM takes is determined by the Trigger Count and Sample Count.

After this function executes, each element in the `readingArray` parameter is an actual reading or a value indicating that an overrange condition occurred. If an overrange condition occurs, the corresponding `readingArray` element contains an IEEE defined NaN (Not a Number) value and the function returns `IVIDMM_WARN_OVER_RANGE`. Test each element in the `readingArray` parameter for overrange with the `IviDmm_IsOverRange` function.

The value of the `IVIDMM_ATTR_FUNCTION` attribute determines the units for each element of the `readingArray` parameter. Refer to Section 11.3.3, ***`IviDmm_FetchMultiPoint`*** for more details.

C Prototype

```
ViStatus IviDmm_ReadMultiPoint (ViSession vi,
                                ViInt32 maxTime,
                                ViInt32 arraySize,
                                ViReal64 readingArray[],
                                ViInt32 *actualPts);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`maxTime`	Specifies the maximum length of time allowed for the function to complete in milliseconds. Defined Values: `IVIDMM_VAL_MAX_TIME_IMMEDIATE` – The function returns immediately. If no valid measurement value exists, the function returns an error. `IVIDMM_VAL_MAX_TIME_INFINITE` – The function waits indefinitely for the measurement to complete.	`ViInt32`
`arraySize`	Specifies the number of elements in the readingArray.	`ViInt32`

Outputs	Description	Base Type
`readingArray`	Array of measured values. This array must be allocated by the user.	`ViReal64[]`
`actualPts`	The number of measured values placed in readingArray.	`ViInt32`

Return Values

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 Codes	Description
IVIDMM_WARN_OVER_RANGE	Overrange warning
IVIDMM_ERROR_MAX_TIME_EXCEEDED	Max time exceeded before the operation completed

11.4 IviDmmMultiPoint Behavior Model

The following behavior model shows the relationship between the IviDmmMultiPoint extension group and DMM behavior.

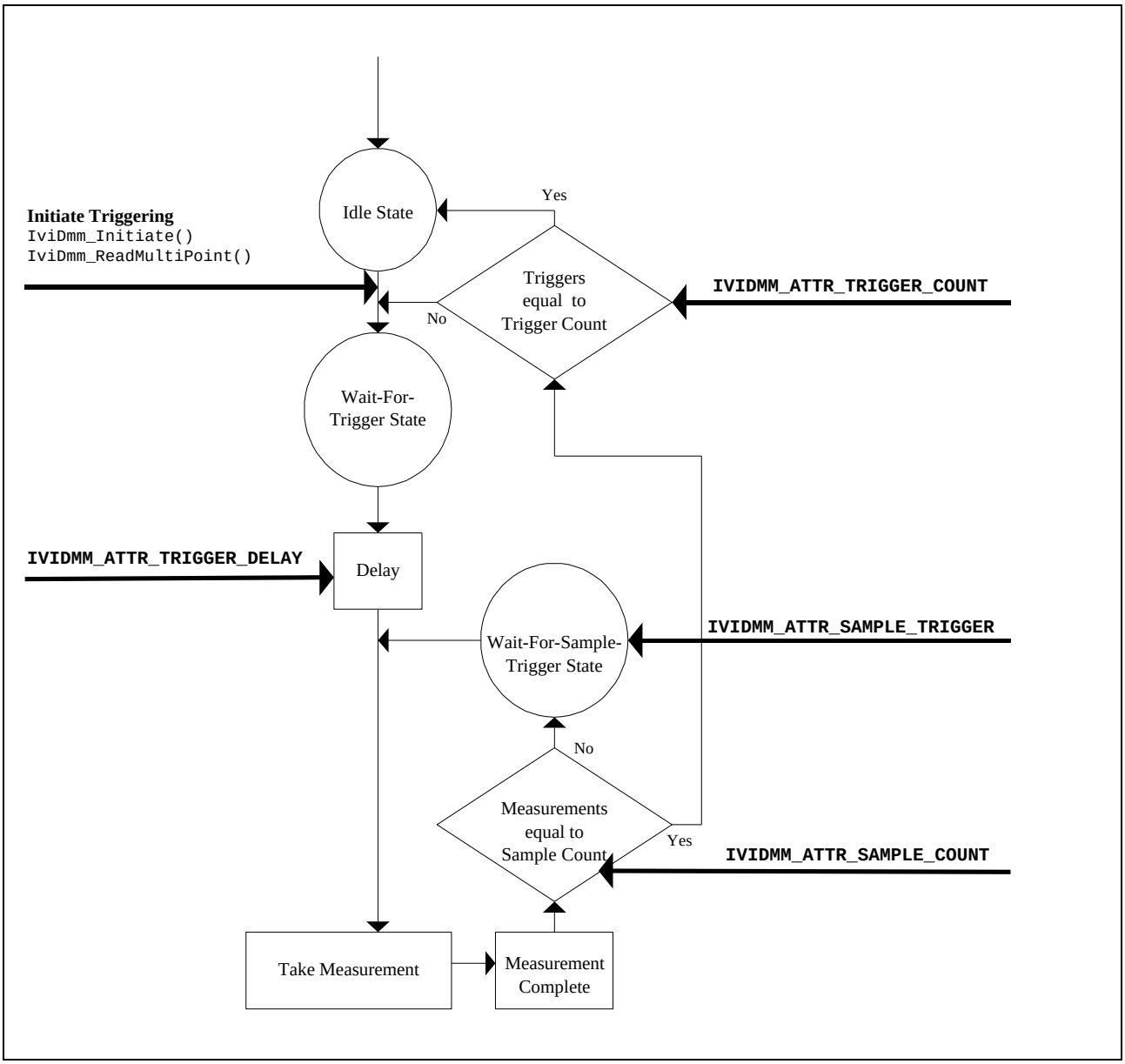


Figure 11-1. IviDmmMultiPoint Behavior Model

The IviDmmMultiPoint behavior model builds upon the fundamental IviDmm behavior model and only documents additional items introduced by the IviDmmMultiPoint extension group. The main state is the Idle state. Typically, the user configures the IviDmmMultiPoint attributes while DMM is in the Idle state. IviDmmMultiPoint attributes can be configured individually with the `IviDmm_SetAttribute` function or with the high-level `IviDmm_ConfigureMultiPoint` function.

The `IviDmm_Initiate` and `IviDmm_ReadMultiPoint` functions cause the DMM to leave the Idle state and transition to the Wait-For-Trigger state. The `IviDmm_ReadMultiPoint` function does not return until

the measurement process is complete and the DMM returns to the Idle state. The `IviDmm_Initiate` function returns as soon as the DMM leaves the Idle state.

The IviDmmMultiPoint extension group does not add additional capabilities to the Wait-For-Trigger state.

After the DMM leaves the Wait-For-Trigger state, it then executes a delay. The length of the delay is specified by the attribute `IVIDMM_ATTR_TRIGGER_DELAY`. After the measurement is taken, the DMM then, if it is capable of doing so, generates the Measurement Complete signal.

The DMM then compares the sample count with the number of measurements taken since the last trigger event. The sample count is specified by the attribute `IVIDMM_ATTR_SAMPLE_COUNT`. If the number measurements is not equal to the sample count the DMM moves to the Wait-For-Sample-Trigger state. The DMM remains in the Wait-For-Sample-Trigger state until the event specified by the attribute `IVIDMM_ATTR_SAMPLE_TRIGGER` occurs. Then it takes another measurement.

Once the number of measurements taken is equal to the sample count, the DMM then compares the number trigger count with the number of trigger events that have occurred since either the `IviDmm_Initiate` or `IviDmm_ReadMultiPoint` function was called. The trigger count is specified by the attribute `IVIDMM_ATTR_TRIGGER_COUNT`. If the number of trigger events is not equal to the trigger count, the DMM returns to the Wait-For-Trigger state.

Once the number of trigger events is equal to the trigger count, the DMM returns to the Idle state. The `IviDmm_FetchMultiPoint` function is used to retrieve measured data from measurements initiated by the `IviDmm_Initiate` function. The measurement data returned from the `IviDmm_ReadMultiPoint` and `IviDmm_FetchMultiPoint` functions is acquired after the the DMM has left the Wait-For-Trigger state.

12. IviDmmTriggerSlope Extension Group

12.1 IviDmmTriggerSlope Extension Group Overview

The IviDmmTriggerSlope extension group supports DMMs that can specify the polarity of the external trigger signal. It defines an attribute and a function to configure this polarity.

Special Note To Users

Typically the specific driver disables extension groups that the application program does not explicitly use or enable. The IviDmmTriggerSlope extension capability group affects the behavior of the instrument regardless of the value of the `IVIDMM_ATTR_TRIGGER_SLOPE` attribute. It is not possible for the specific driver to disable this extension capability group.

Therefore, it is the responsibility of the user to ensure that the slope of the trigger signal is correct for their application. Most DMMs do not have a programmable trigger slope and do not implement the IviDmmTriggerSlope extension capability group. Users should avoid using the IviDmmTriggerSlope extension capability in their test program source code so that they can maximize the set of instruments that they can use interchangeably.

For instrument drivers that implement the IviDmmTriggerSlope extension, the user can set the value of the `IVIDMM_ATTR_TRIGGER_SLOPE` attribute in the IVI configuration file. For instruments that do not implement the IviDmmTriggerSlope extension group, the user must ensure that trigger signal that their instrument receives has the correct polarity. This can be done with external circuitry.

12.2 IviDmmTriggerSlope Attributes

The IviDmmTriggerSlope extension group defines the following attribute:

- `IVIDMM_ATTR_TRIGGER_SLOPE`

This section describes the behavior and requirements of this attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

12.2.1 IVIDMM_ATTR_TRIGGER_SLOPE

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureTriggerSlope

Description

Specifies the polarity of the external trigger slope. The DMM triggers on either the rising or the falling edge of the external trigger source depending on the value of this attribute.

Defined Values

Value	Description
IVIDMM_VAL_POSITIVE	Sets the trigger event to occur on the rising edge of the trigger pulse.
IVIDMM_VAL_NEGATIVE	Sets the trigger event to occur on the falling edge of the trigger pulse.

Compliance Notes

1. If a specific driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_SLOPE_SPECIFIC_EXT_BASE.

2. If a class driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_TRIGGER_SLOPE_CLASS_EXT_BASE and less than IVIDMM_VAL_TRIGGER_SLOPE_SPECIFIC_EXT_BASE.

12.3 IviDmmTriggerSlope Functions

The IviDmmTriggerSlope extension group defines the following function:

- `IviDmm_ConfigureTriggerSlope`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

12.3.1 IviDmm_ConfigureTriggerSlope

Description

This function configures the polarity of the external trigger source of the DMM.

C Prototype

```
ViStatus IviDmm_ConfigureTriggerSlope (ViSession vi,
                                       ViInt32 polarity);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`polarity`	Specifies the trigger slope. The driver uses this value to set the `IVIDMM_ATTR_TRIGGER_SLOPE` attribute. See the attribute description for more details.	`ViInt32`

Return Values

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

12.4 IviDmmTriggerSlope Behavior Model

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

13. IviDmmSoftwareTrigger Extension Group

13.1 IviDmmSoftwareTrigger Extension Group Overview

The IviDmmSoftwareTrigger extension group supports DMMs that can initiate a measurement based on a software trigger signal. The user can send a software trigger to cause the DMM to initiate a measurement.

13.2 IviDmmSoftwareTrigger Functions

The IviDmmSoftwareTrigger extension group defines the following function:

- `IviDmm_SendSoftwareTrigger`~~`IviDmm_SendSWTrigger`~~

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

13.2.1 IviDmm_SendSoftwareTrigger

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

~~Inputs~~	~~Description~~	~~Base Type~~
~~vi~~	~~Instrument handle~~	~~ViSession~~

~~Return Values~~

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

~~Completion Codes~~	~~Description~~
~~IVIDMM_ERROR_TRIGGER_NOT_SOFTWARE~~	~~The trigger source is not set to a software trigger source.~~

13.3 IviDmmSoftwareTrigger Behavior Model

The behavior model of the IviDmmSoftwareTrigger follows the behavior model of the IviDmmBase capability group as described in Section 4.4, *IviDmmBase Behavior Model* and the IviDmmMultiPoint extension group described in Section 11.4, *IviDmmMultiPoint Behavior Model*. Furthermore, it defines an additional trigger event for the trigger source.

The DMM leaves the Wait-For-Trigger state when it receives a trigger event specified by the `IVIDMM_ATTR_TRIGGER_SOURCE` attribute. The DMM leaves the Wait-For-Sample-Trigger state when it receieves a trigger event specified by the `IVIDMM_ATTR_SAMPLE_TRIGGER` attribute When the trigger source or sample trigger is set to `IVIDMM_VAL_SOFTWARE_TRIG`~~`IVIDMM_VAL_SW_TRIG_FUNC`~~, the `IviDmm_SendSoftwareTrigger` function is used to generate the trigger event . Calling this function causes the DMM to take a measurement.

13.4 IviDmmSoftwareTrigger Compliance Notes

1. Specific drivers that implement this extension group must implement the IVIDMM_VAL_SOFTWARE_TRIG ~~IVIDMM_VAL_SW_TRIG_FUNC~~ value for the IVIDMM_ATTR_TRIGGER_SOURCE attribute in the IviDmmBase capability group.

14. IviDmmDeviceInfo Extension Group

14.1 IviDmmDeviceInfo Extension Group Overview

The IviDmmDeviceInfo extension group defines a set of read-only attributes for DMMs that can be queried to determine how they are presently configured. These attributes are the aperture time and the aperture time units.

14.2 IviDmmDeviceInfo Attributes

The IviDmmDeviceInfo extension group defines the following attributes:

- `IVIDMM_ATTR_APERTURE_TIME`
- `IVIDMM_ATTR_APERTURE_TIME_UNITS`

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

14.2.1 IVIDMM_ATTR_APERTURE_TIME

Data Type	Access	Channel-Based	Coercion	High Level Functions
`ViReal64`	`RO`	No	None	`IviDmm_GetApertureTimeInfo`

Description

Returns the measurement aperture time based on the present configuration. The units for this attribute are specified by attribute `IVIDMM_ATTR_APERTURE_TIME_UNITS`.

The aperture time is also known as the integration time.

14.2.2 IVIDMM_ATTR_APERTURE_TIME_UNITS

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	RO	No	None	IviDmm_GetApertureTimeInfo

Description

Returns the units for the IVIDMM_ATTR_APERTURE_TIME attribute.

Defined Values

Value	Description
IVIDMM_VAL_SECONDS	Reports that the units for the value returned by IVIDMM_ATTR_APERTURE_TIME are seconds.
IVIDMM_VAL_POWER_LINE_CYCLES	Reports that the units for the value returned by IVIDMM_ATTR_APERTURE_TIME are Power Line Cycles.

14.3 IviDmmDeviceInfo Functions

The IviDmmDeviceInfo extension group defines the following function:

- `IviDmm_GetApertureTimeInfo`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

14.3.1 IviDmm_GetApertureTimeInfo

Description

This function returns additional information about the state of the instrument. Specifically, it returns the aperture time and the aperture time units.

C Prototype

```
ViStatus IviDmm_GetApertureTimeInfo (ViSession vi
                                     ViReal64 *apertureTime,
                                     ViInt32 *apertureTimeUnits);
```

Parameters

Inputs	Description	Base Type
vi	Instrument handle	ViSession

Outputs	Description	Base Type
apertureTime	Returns the value of the IVIDMM_ATTR_APERTURE_TIME attribute. See the attribute description for more details.	ViReal64
apertureTimeUnits	Returns the value of the IVIDMM_ATTR_APERTURE_TIME_UNITS attribute. See the attribute description for more details.	ViInt32

Return Values

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

14.4 IviDmmDeviceInfo Behavior Model

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

15. IviDmmAutoRangeValue Extension Group

15.1 IviDmmAutoRangeValue Extension Group Overview

The IviDmmAutoRangeValue extension supports DMMs with the capability to return the actual range value when auto ranging.

15.2 IviDmmAutoRangeValue Attributes

The IviDmmAutoRangeValue extension group defines the following attribute:

- `IVIDMM_ATTR_AUTO_RANGE_VALUE`

This section describes the behavior and requirements of this attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

15.2.1 IVIDMM_ATTR_AUTO_RANGE_VALUE

Data Type	Access	Channel-Based	Coercion	High Level Functions
`ViReal64`	`RO`	No	None	`IviDmm_GetAutoRangeValue`

Description

Returns the actual range that the DMM is currently using, even while it is auto-ranging.

15.3 IviDmmAutoRangeValue Functions

The IviDmmAutoRangeValue extension group defines the following function:

- `IviDmm_GetAutoRangeValue`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in *Section 22, IviDmm Function Hierarchy*.

15.3.1 IviDmm_GetAutoRangeValue

Description

This function returns the actual range the DMM is currently using, even while it is auto-ranging.

C Prototype

```
ViStatus IviDmm_GetAutoRangeValue (ViSession vi
                                   ViReal64 *autoRangeValue;
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`

Outputs	Description	Base Type
`autoRangeValue`	Returns the value of the `IVIDMM_AUTO_RANGE_VALUE` attribute. See the attribute description for more details.	`ViReal64`

Return Values

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

15.4 IviDmmAutoRangeValue Behavior Model

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

15.5 IviDmmAutoRangeValue Compliance Notes

1. If the specific driver implements this extension group, then it must also implement the `IVIDMM_ATTR_AUTO_RANGE_ON` value for the `IVIDMM_ATTR_RANGE` attribute in the IviDmmBase capability group.

16. IviDmmAutoZero Extension Group

16.1 IviDmmAutoZero Extension Group Overview

The IviDmmAutoZero extension supports DMMs with the capability to take an auto zero reading. In general, the auto-zero capability of a DMM normalizes all measurements based on a Zero Reading.

16.2 IviDmmAutoZero Attributes

The IviDmmAutoZero extension group defines the following attribute:

- `IVIDMM_ATTR_AUTO_ZERO`

This section describes the behavior and requirements of this attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

16.2.1 IVIDMM_ATTR_AUTO_ZERO

Data Type	Access	Channel-Based	Coercion	High Level Functions
ViInt32	R/W	No	None	IviDmm_ConfigureAutoZeroMode

Description

Specifies the auto-zero mode. When the auto-zero mode is enabled, the DMM internally disconnects the input signal and takes a Zero Reading. The DMM then subtracts the Zero Reading from the measurement. This prevents offset voltages present in the instrument's input circuitry from affecting measurement accuracy.

Defined Values

Value	Description
IVIDMM_VAL_AUTO_ZERO_OFF	Disables the auto-zero feature.
IVIDMM_VAL_AUTO_ZERO_ON	Configures the DMM to take a Zero Reading for each measurement. The DMM subtracts the Zero Reading from the value it measures.
IVIDMM_VAL_AUTO_ZERO_ONCE	Configures the DMM to take a Zero Reading immediately. The DMM then subtracts this Zero Reading from all subsequent values it measures.

Compliance Notes

1. If a specific driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_AUTO_ZERO_SPECIFIC_EXT_BASE.

2. If a class driver defines additional values for this attribute, the magnitude of the actual values must be greater than or equal to IVIDMM_VAL_AUTO_ZERO_CLASS_EXT_BASE and less than IVIDMM_VAL_AUTO_ZERO_SPECIFIC_EXT_BASE.

16.3 IviDmmAutoZero Functions

The IviDmmAutoZero extension group defines the following function:

- `IviDmm_ConfigureAutoZeroMode`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in *Section 22, IviDmm Function Hierarchy*.

16.3.1 IviDmm_ConfigureAutoZeroMode

Description

This function configures the auto zero mode of the DMM.

C Prototype

```
ViStatus IviDmm_ConfigureAutoZeroMode (ViSession vi
                                       ViInt32 autoZeroMode);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`autoZeroMode`	Specifies the auto-zero mode The driver uses this value to set the `IVIDMM_ATTR_AUTO_ZERO` attribute. See the attribute description for more details.	`ViInt32`

Return Values

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

16.4 IviDmmAutoZero Behavior Model

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

17. IviDmmPowerLineFrequency Extension Group

17.1 IviDmmPowerLineFrequency Extension Group Overview

The IviDmmPowerLineFrequency extension supports DMMs with the capability to specify the power line frequency.

Special Note To Users

Typically the specific driver disables extension groups that the application program does not explicitly use or enable. The IviDmmPowerLineFrequency extension capability group affects the behavior of the instrument regardless of the value of the `IVIDMM_ATTR_POWERLINE_FREQ` attribute. It is not possible for the specific driver to disable this extension capability group.

Therefore, it is the responsibility of the user to ensure that the power line frequency is correct for their application. Most DMMs do not have a programmable power line frequency and do not implement the IviDmmPowerLineFrequency extension capability group. Users should avoid using the IviDmmPowerLineFrequency extension group in their test program source code so that they can maximize the set of instruments that they can use interchangeably.

For instrument drivers that implement the IviDmmPowerLineFrequency extension, the user can set the value of the `IVIDMM_ATTR_POWERLINE_FREQ` attribute in the IVI configuration file. For instruments that do not implement the IviDmmPowerLineFrequency extension group, the user must ensure that their instrument is set to use the correct power line frequency. Users can manually change the power line frequency setting on most DMMs by means of a switch on the instrument's back panel.

17.2 IviDmmPowerLineFrequency Attributes

The IviDmmPowerLineFrequency extension group defines the following attribute:

- `IVIDMM_ATTR_POWERLINE_FREQ`

This section describes the behavior and requirements of this attribute. The actual value for each attribute ID is defined in Section 18, *IviDmm Attribute ID Definitions.*

17.2.1 IVIDMM_ATTR_POWERLINE_FREQ

Data Type	Access	Channel-Based	Coercion	High Level Functions
`ViReal64`	`R/W`	No	None	`IviDmm_ConfigurePowerLineFrequency`

Description

Specifies the power line frequency in Hertz.

17.3 IviDmmPowerLineFrequency Functions

The IviDmmPowerLineFrequency extension group defines the following function:

- `IviDmm_ConfigurePowerLineFrequency`

This section describes the behavior and requirements of this function.
The required function hierarchy for this function as well as for other IviDmm functions, the IVI inherent functions, and VXI*plug&play* required functions is shown in Section 22, *IviDmm Function Hierarchy*.

17.3.1 IviDmm_ConfigurePowerLineFrequency

Description

This function configures the power line frequency of the DMM.

C Prototype

```
ViStatus IviDmm_ConfigurePowerLineFrequency (ViSession vi
                                             ViReal64 powerLineFreq);
```

Parameters

Inputs	Description	Base Type
`vi`	Instrument handle	`ViSession`
`powerLineFreq`	Specifies the power line frequency The driver uses this value to set the `IVIDMM_ATTR_POWERLINE_FREQ` attribute. See the attribute description for more details.	`ViReal64`

Return Values

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

17.4 IviDmmPowerLineFrequency Behavior Model

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

18. IviDmm Attribute ID Definitions

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

Table 18-1. IviDmm Attributes ID Values

Attribute Name	ID Definition
IVIDMM_ATTR_FUNCTION	IVI_CLASS_PUBLIC_ATTR_BASE + 1
IVIDMM_ATTR_RANGE	IVI_CLASS_PUBLIC_ATTR_BASE + 2
IVIDMM_ATTR_RESOLUTION_ABSOLUTE	IVI_CLASS_PUBLIC_ATTR_BASE + 8
IVIDMM_ATTR_TRIGGER_SOURCE	IVI_CLASS_PUBLIC_ATTR_BASE + 4
IVIDMM_ATTR_TRIGGER_DELAY	IVI_CLASS_PUBLIC_ATTR_BASE + 5
IVIDMM_ATTR_AC_MIN_FREQ	IVI_CLASS_PUBLIC_ATTR_BASE + 6
IVIDMM_ATTR_AC_MAX_FREQ	IVI_CLASS_PUBLIC_ATTR_BASE + 7
IVIDMM_ATTR_FREQ_VOLTAGE_RANGE	IVI_CLASS_PUBLIC_ATTR_BASE + 101
IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE	IVI_CLASS_PUBLIC_ATTR_BASE + 201
IVIDMM_ATTR_TEMP_TC_TYPE	IVI_CLASS_PUBLIC_ATTR_BASE + 231
IVIDMM_ATTR_TEMP_TC_REF_JUNC_TYPE	IVI_CLASS_PUBLIC_ATTR_BASE + 232
IVIDMM_ATTR_TEMP_TC_FIXED_REF_JUNC	IVI_CLASS_PUBLIC_ATTR_BASE + 233
IVIDMM_ATTR_TEMP_RTD_ALPHA	IVI_CLASS_PUBLIC_ATTR_BASE + 241
IVIDMM_ATTR_TEMP_RTD_RES	IVI_CLASS_PUBLIC_ATTR_BASE + 242
IVIDMM_ATTR_TEMP_THERMISTOR_RES	IVI_CLASS_PUBLIC_ATTR_BASE + 251
IVIDMM_ATTR_SAMPLE_COUNT	IVI_CLASS_PUBLIC_ATTR_BASE + 301
IVIDMM_ATTR_SAMPLE_TRIGGER	IVI_CLASS_PUBLIC_ATTR_BASE + 302
IVIDMM_ATTR_SAMPLE_INTERVAL	IVI_CLASS_PUBLIC_ATTR_BASE + 303
IVIDMM_ATTR_TRIGGER_COUNT	IVI_CLASS_PUBLIC_ATTR_BASE + 304
IVIDMM_ATTR_MEAS_COMPLETE_DEST	IVI_CLASS_PUBLIC_ATTR_BASE + 305
IVIDMM_ATTR_APERTURE_TIME	IVI_CLASS_PUBLIC_ATTR_BASE + 321
IVIDMM_ATTR_APERTURE_TIME_UNITS	IVI_CLASS_PUBLIC_ATTR_BASE + 322
IVIDMM_ATTR_AUTO_RANGE_VALUE	IVI_CLASS_PUBLIC_ATTR_BASE + 331
IVIDMM_ATTR_AUTO_ZERO	IVI_CLASS_PUBLIC_ATTR_BASE + 332
IVIDMM_ATTR_POWERLINE_FREQ	IVI_CLASS_PUBLIC_ATTR_BASE + 333
IVIDMM_ATTR_TRIGGER_SLOPE	IVI_CLASS_PUBLIC_ATTR_BASE + 334

18.1 IviDmm Obsolete Attribute Names

The following attribute names are reserved by the IviDmm specification 1.0. Future versions of this specification cannot use these names:

- `IVIDMM_ATTR_RESOLUTION`

18.2 IviDmm Obsolete Attribute ID Values

The following attribute ID values are reserved by the IviDmm specification 1.0. Future versions of this specification cannot use these values:

- `IVI_CLASS_PUBLIC_ATTR_BASE + 3`

19. IviDmm Attribute Value Definitions

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

IVIDMM_ATTR_APERTURE_TIME_UNITS

Value Name	Actual Value
IVIDMM_VAL_SECONDS	0
IVIDMM_VAL_POWER_LINE_CYCLES	1

IVIDMM_ATTR_AUTO_ZERO

Value Name	Actual Value
IVIDMM_VAL_AUTO_ZERO_OFF	0
IVIDMM_VAL_AUTO_ZERO_ON	1
IVIDMM_VAL_AUTO_ZERO_ONCE	2
IVIDMM_VAL_AUTO_ZERO_CLASS_EXT_BASE	100
IVIDMM_VAL_AUTO_ZERO_SPECIFIC_EXT_BASE	1000

IVIDMM_ATTR_FREQ_VOLTAGE_RANGE

Value Name	Actual Value
IVIDMM_VAL_AUTO_RANGE_ON	-1.0
IVIDMM_VAL_AUTO_RANGE_OFF	-2.0
IVIDMM_VAL_FREQ_VOLT_RANGE_CLASS_EXT_BASE	-100.0
IVIDMM_VAL_FREQ_VOLT_RANGE_SPECIFIC_EXT_BASE	-1000.0

IVIDMM_ATTR_FUNCTION

Value Name	Actual Value
IVIDMM_VAL_DC_VOLTS	1
IVIDMM_VAL_AC_VOLTS	2
IVIDMM_VAL_DC_CURRENT	3
IVIDMM_VAL_AC_CURRENT	4
IVIDMM_VAL_2_WIRE_RES	5
IVIDMM_VAL_4_WIRE_RES	101
IVIDMM_VAL_AC_PLUS_DC_VOLTS	106
IVIDMM_VAL_AC_PLUS_DC_CURRENT	107

Value Name	Actual Value
IVIDMM_VAL_FREQ	104
IVIDMM_VAL_PERIOD	105
IVIDMM_VAL_TEMPERATURE	108
IVIDMM_VAL_FUNC_CLASS_EXT_BASE	500
IVIDMM_VAL_FUNC_SPECIFIC_EXT_BASE	1000

The following values are reserved by the IviDmm specification 1.0 for the IVIDMM_ATTR_FUNCTION attribute. Future versions of this specification cannot use these values for this attribute:

- 102
- 103
- 109
- 110
- 111

IVIDMM_ATTR_MEAS_COMPLETE_DEST

Value Name	Actual Value
IVIDMM_VAL_NONE	-1
IVIDMM_VAL_EXTERNAL	2
IVIDMM_VAL_TTL0	111
IVIDMM_VAL_TTL1	112
IVIDMM_VAL_TTL2	113
IVIDMM_VAL_TTL3	114
IVIDMM_VAL_TTL4	115
IVIDMM_VAL_TTL5	116
IVIDMM_VAL_TTL6	117
IVIDMM_VAL_TTL7	118
IVIDMM_VAL_ECL0	119
IVIDMM_VAL_ECL1	120
IVIDMM_VAL_PXI_STAR	131
IVIDMM_VAL_RTSI_0	140
IVIDMM_VAL_RTSI_1	141
IVIDMM_VAL_RTSI_2	142
IVIDMM_VAL_RTSI_3	143
IVIDMM_VAL_RTSI_4	144
IVIDMM_VAL_RTSI_5	145
IVIDMM_VAL_RTSI_6	146
IVIDMM_VAL_TRIGGER_SOURCE_CLASS_EXT_BASE	500

Value Name	Actual Value
IVIDMM_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE	1000

IVIDMM_ATTR_RANGE

Value Name	Actual Value
IVIDMM_VAL_AUTO_RANGE_ON	-1.0
IVIDMM_VAL_AUTO_RANGE_OFF	-2.0
IVIDMM_VAL_AUTO_RANGE_ONCE	-3.0
IVIDMM_VAL_RANGE_CLASS_EXT_BASE	-100.0
IVIDMM_VAL_RANGE_SPECIFIC_EXT_BASE	-1000.0

IVIDMM_ATTR_SAMPLE_TRIGGER

Value Name	Actual Value
IVIDMM_VAL_IMMEDIATE	1
IVIDMM_VAL_EXTERNAL	2
IVIDMM_VAL_SOFTWARE_TRIG~~IVIDMM_VAL_SW_TRIG_FUNC~~	3
IVIDMM_VAL_TTL0	111
IVIDMM_VAL_TTL1	112
IVIDMM_VAL_TTL2	113
IVIDMM_VAL_TTL3	114
IVIDMM_VAL_TTL4	115
IVIDMM_VAL_TTL5	116
IVIDMM_VAL_TTL6	117
IVIDMM_VAL_TTL7	118
IVIDMM_VAL_ECL0	119
IVIDMM_VAL_ECL1	120
IVIDMM_VAL_PXI_STAR	131
IVIDMM_VAL_RTSI_0	140
IVIDMM_VAL_RTSI_1	141
IVIDMM_VAL_RTSI_2	142
IVIDMM_VAL_RTSI_3	143
IVIDMM_VAL_RTSI_4	144
IVIDMM_VAL_RTSI_5	145

Value Name	Actual Value
IVIDMM_VAL_RTSI_6	146
IVIDMM_VAL_INTERVAL	10
IVIDMM_VAL_TRIGGER_SOURCE_CLASS_EXT_BASE	500
IVIDMM_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE	1000

The following values are reserved by the IviDmm specification 1.0 for the IVIDMM_ATTR_SAMPLE_TRIGGER attribute. Future versions of this specification cannot use these values:

- 101

IVIDMM_ATTR_TEMP_TC_REF_JUNC_TYPE

Value Name	Actual Value
IVIDMM_VAL_TEMP_REF_JUNC_INTERNAL	1
IVIDMM_VAL_TEMP_REF_JUNC_FIXED	2
IVIDMM_VAL_TEMP_REF_JUNC_CLASS_EXT_BASE	100
IVIDMM_VAL_TEMP_REF_JUNC_SPECIFIC_EXT_BASE	1000

IVIDMM_ATTR_TEMP_TC_TYPE

Value Name	Actual Value
IVIDMM_VAL_TEMP_TC_B	1
IVIDMM_VAL_TEMP_TC_C	2
IVIDMM_VAL_TEMP_TC_D	3
IVIDMM_VAL_TEMP_TC_E	4
IVIDMM_VAL_TEMP_TC_G	5
IVIDMM_VAL_TEMP_TC_J	6
IVIDMM_VAL_TEMP_TC_K	7
IVIDMM_VAL_TEMP_TC_N	8
IVIDMM_VAL_TEMP_TC_R	9
IVIDMM_VAL_TEMP_TC_S	10
IVIDMM_VAL_TEMP_TC_T	11
IVIDMM_VAL_TEMP_TC_U	12
IVIDMM_VAL_TEMP_TC_V	13
IVIDMM_VAL_TEMP_TC_TYPE_CLASS_EXT_BASE	100
IVIDMM_VAL_TEMP_TC_TYPE_SPECIFIC_EXT_BASE	1000

IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE

Value Name	Actual Value
IVIDMM_VAL_THERMOCOUPLE	1
IVIDMM_VAL_THERMISTOR	2
IVIDMM_VAL_2_WIRE_RTD	3
IVIDMM_VAL_4_WIRE_RTD	4
IVIDMM_VAL_TRANSDUCER_CLASS_EXT_BASE	100
IVIDMM_VAL_TRANSDUCER_SPECIFIC_EXT_BASE	1000

IVIDMM_ATTR_TRIGGER_DELAY

Value Name	Actual Value
IVIDMM_VAL_AUTO_DELAY_ON	-1.0
IVIDMM_VAL_AUTO_DELAY_OFF	-2.0
IVIDMM_VAL_TRIGGER_DELAY_CLASS_EXT_BASE	-100.0
IVIDMM_VAL_TRIGGER_DELAY_SPECIFIC_EXT_BASE	-1000.0

IVIDMM_ATTR_TRIGGER_SLOPE

Value Name	Actual Value
IVIDMM_VAL_POSITIVE	0
IVIDMM_VAL_NEGATIVE	1
IVIDMM_VAL_TRIGGER_SLOPE_CLASS_EXT_BASE	100
IVIDMM_VAL_TRIGGER_SLOPE_SPECIFIC_EXT_BASE	1000

IVIDMM_ATTR_TRIGGER_SOURCE

Value Name	Actual Value
IVIDMM_VAL_IMMEDIATE	1
IVIDMM_VAL_EXTERNAL	2
IVIDMM_VAL_SOFTWARE_TRIG~~IVIDMM_VAL_SW_TRIG_FUNC~~	3
IVIDMM_VAL_TTL0	111
IVIDMM_VAL_TTL1	112
IVIDMM_VAL_TTL2	113
IVIDMM_VAL_TTL3	114
IVIDMM_VAL_TTL4	115

Value Name	Actual Value
IVIDMM_VAL_TTL5	116
IVIDMM_VAL_TTL6	117
IVIDMM_VAL_TTL7	118
IVIDMM_VAL_ECL0	119
IVIDMM_VAL_ECL1	120
IVIDMM_VAL_PXI_STAR	131
IVIDMM_VAL_RTSI_0	140
IVIDMM_VAL_RTSI_1	141
IVIDMM_VAL_RTSI_2	142
IVIDMM_VAL_RTSI_3	143
IVIDMM_VAL_RTSI_4	144
IVIDMM_VAL_RTSI_5	145
IVIDMM_VAL_RTSI_6	146
IVIDMM_VAL_TRIGGER_SOURCE_CLASS_EXT_BASE	500
IVIDMM_VAL_TRIGGER_SOURCE_SPECIFIC_EXT_BASE	1000

The following values are reserved by the IviDmm specification 1.0 for the IVIDMM_ATTR_TRIGGER_SOURCE attribute. Future versions of this specification cannot use these values:

- 101

19.1 IviDmm Obsolete Attribute Value Names

The following attribute value names are reserved by the IviDmm specification 1.0. Future versions of this specification cannot use these names:

- IVIDMM_VAL_DIODE
- IVIDMM_VAL_CONTINUITY
- IVIDMM_VAL_TEMP_C
- IVIDMM_VAL_TEMP_F
- IVIDMM_VAL_SIEMENS
- IVIDMM_VAL_COULOMBS
- IVIDMM_VAL_3_5_DIGITS
- IVIDMM_VAL_4_DIGITS
- IVIDMM_VAL_4_5_DIGITS
- IVIDMM_VAL_5_DIGITS
- IVIDMM_VAL_5_5_DIGITS
- IVIDMM_VAL_6_DIGITS
- IVIDMM_VAL_6_5_DIGITS
- IVIDMM_VAL_7_DIGITS
- IVIDMM_VAL_7_5_DIGITS
- IVIDMM_VAL_50_HERTZ
- IVIDMM_VAL_60_HERTZ
- IVIDMM_VAL_400_HERTZ
- IVIDMM_VAL_GPIB_GET
- IVIDMM_VAL_SW_TRIG_FUNC

20. IviDmm Function Parameter Value Definitions

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

IviDmm_Read

Parameter: maxTime

Value Name	Actual Value
IVIDMM_VAL_MAX_TIME_IMMEDIATE	0
IVIDMM_VAL_MAX_TIME_INFINITE	0xFFFFFFFF

IviDmm_Fetch

Parameter: maxTime

Same as defined for the maxTime parameter of the IviDmm_Read function.

IviDmm_ReadMultiPoint

Parameter: maxTime

Same as defined for the maxTime parameter of the IviDmm_Read function.

IviDmm_FetchMultiPoint

Parameter: maxTime

Same as defined for the maxTime parameter of the IviDmm_Read function.

21. IviDmm Error and Completion Code Value Definitions

The table below specifies the actual value for each status code that the IviDmm class specification defines.

Table 21-1. IviDmm Completion Codes

Completion Code	Actual Value
IVIDMM_ERROR_MAX_TIME_EXCEEDED	IVI_CLASS_ERROR_BASE + 3
IVIDMM_WARN_OVER_RANGE	IVI_CLASS_WARN_BASE + 1

21.1 IviDmm Obsolete Error and Completion Code Names

The following error and completion codes names are reserved by the IviDmm specification 1.0. Future versions of this specification cannot use these names:

- IVIDMM_ERROR_ACCURACY_UNKNOWN
- IVIDMM_ERROR_ACCURACY_UNKNOWN_WHILE_AUTORANGING

21.2 IviDmm Obsolete Error and Completion Code Values

The following error and completion codes values are reserved by the IviDmm specification 1.0. Future versions of this specification cannot use these values:

- IVI_CLASS_ERROR_BASE + 1
- IVI_CLASS_ERROR_BASE + 2

22. IviDmm Function Hierarchy

The IviDmm class function hierarchy, including IVI and VXI*plug&play* required functions, is shown in the following table.

☞ **Note:** ***To reduce complexity, the individual Set and Get attribute functions required by IVI are not shown in the following table.***

Table 22-1. IviDmm Function Hierarchy

Name or Class	Function Name	Required By
Initialize	`IviDmm_init`	VPP
Initialize with Options	`IviDmm_InitWithOptions`	IVI
Configuration…		
Configure Measurement	`IviDmm_ConfigureMeasurement`	Base
Specific Measurements…		
Configure AC Bandwidth	`IviDmm_ConfigureACBandwidth`	AC
Configure Frequency Voltage Range	`IviDmm_ConfigureFrequencyVoltageRange`	FRQ
Temperature …		
Configure Transducer Type	`IviDmm_ConfigureTransducerType`	TMP
Configure Thermocouple	`IviDmm_ConfigureThermocouple`	TC
Configure Fixed Reference Junction	`IviDmm_ConfigureFixedRefJunction`	TC
Configure RTD	`IviDmm_ConfigureRTD`	RTD
Configure Thermistor	`IviDmm_ConfigureThermistor`	THM
Trigger…		
Configure Trigger	`IviDmm_ConfigureTrigger`	Base
Configure Trigger Slope	`IviDmm_ConfigureTriggerSlope`	TS
MultiPoint …		
Configure Multi Point	`IviDmm_ConfigureMultiPoint`	MP
Configure Measurement Complete Destination	`IviDmm_ConfigureMeasCompleteDest`	MP
Measurement Operation Options…		
Configure Auto Zero Mode	`IviDmm_ConfigureAutoZeroMode`	AZ
Configure Power Line Frequency	`IviDmm_ConfigurePowerLineFrequency`	PLF
Configuration Information…		
Get Auto Range Value	`IviDmm_GetAutoRangeValue`	ARV
Get Aperture Time Info	`IviDmm_GetApertureTimeInfo`	DI
Set/Get Attribute…		
Set Attribute…	`IviDmm_SetAttribute<type>`	IVI
Get Attribute…	`IviDmm_GetAttribute<type>`	IVI

Name or Class	Function Name	Required By
Measurement…		
Read	IviDmm_Read	Base
Read Multi Point	IviDmm_ReadMultiPoint	MP
Low Level Measurement…		
Initiate	IviDmm_Initiate	Base
Send Software Trigger	IviDmm_SendSoftwareTrigger	SWT
Fetch	IviDmm_Fetch	Base
Fetch Multi Point	IviDmm_FetchMultiPoint	MP
Abort	IviDmm_Abort	Base
Is Overrange	IviDmm_IsOverRange	Base
Utility…		
Reset	IviDmm_reset	VPP
Self-Test	IviDmm_self_test	VPP
Revision Query	IviDmm_revision_query	VPP
Error-Query	IviDmm_error_query	VPP
Error Message	IviDmm_error_message	VPP
Error Info…		
Get Error Info	IviDmm_GetErrorInfo	IVI
Clear Error Info	IviDmm_ClearErrorInfo	IVI
Locking…		
Lock Session	IviDmm_LockSession	IVI
Unlock Session	IviDmm_UnlockSession	IVI
Close	IviDmm_close	VPP

22.1 IviDmm Obsolete Function Names

The following function names are reserved by the IviDmm specification 1.0. Future versions of this specification cannot use these function names:

- IviDmm_Configure
- IviDmm_CalculateAccuracy
- IviDmm_SendSWTrigger

23. Appendix A, Specific Driver Development Guidelines

23.1 Introduction

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

23.2 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 must take appropriate action so that the instrument settings that correspond to the unused extension capability group do not affect the behavior of the instrument when the application program performs an operation that might be affected by those settings.

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

Disabling the IviDmm Measurement Extension Capability Groups

Some measurements that the user selects with the IVIDMM_ATTR_FUNCTION require an extension group to further configure the measurement. The values for the IVIDMM_ATTR_FUNCTION that require additional extension capability groups are shown in the following table.

- IviDmmACMeasurement
- IviDmmFrequencyMeasurement
- IviDmmTemperatureMeasurement
- IviDmmThermocouple
- IviDmmResistanceTemperatureDevice
- IviDmmThermistor

When the IVIDMM_ATTR_FUNCTION is set to one of these values, the corresponding extension capability group affects the behavior of the instruments. Otherwise, the extension capability group does *not* affect the behavior of the instrument and is effectively disabled. Therefore, this section does not recommend how to disable these extension capability groups.

Disabling the IviDmmMultiPoint Extension Group

Attribute values that effectively disable the IviDmmMultiPoint extension group are shown in the following table.

Table 23-1. Values for Disabling the IviDmmMultiPoint Extension Group

Attribute	Value
IVIDMM_ATTR_SAMPLE_COUNT	1
IVIDMM_ATTR_TRIGGER_COUNT	1

Disabling the IviDmmAutoZero Extension Group

Attribute values that effectively disable the IviDmmAutoZero extension group are shown in the following table.

Table 23-2. Values for Disabling the IviDmmAutoZero Extension Group

Attribute	Value
IVIDMM_ATTR_AUTO_ZERO	IVIDMM_VAL_AUTO_ZERO_OFF

Disabling the IviDmmTriggerSlope Extension Group

The purpose of disabling an extension capability group is to make instrument drivers that implement the capability group behave like instrument drivers that do not implement the capability group in cases were it is not used by the application program. The IviDmmTriggerSlope extension group affects the behavior of the instrument regardless of the value of the IVIDMM_ATTR_TRIGGER_SLOPE attribute. Therefore, this section does not define any values that can disable the IviDmmTriggerSlope extension group.

Refer to *Special Notes for Users* in Section 12.1, *IviDmmTriggerSlope Extension group Overview* for further details.

Disabling the IviDmmPowerLineFrequency Extension Group

The purpose of disabling an extension capability group is to make instrument drivers that implement the capability group behave like instrument drivers that do not implement the capability group in cases were it is not used by the application program. The IviDmmPowerLineFrequency extension group affects the behavior of the instrument regardless of the value of the IVIDMM_ATTR_POWER_LINE_FREQ attribute. Therefore, this section does not define any values that can disable the IviDmmPowerLineFrequency extension group.

Refer to *Special Notes for Users* in Section 17.1, *IviDmmPowerLineFrequency Extension group Overview* for further details.

23.3 Query Instrument Status

Based on the value of IVIDMM_ATTR_QUERY_INSTR_STATUS, the instrument may be queried by the specific driver to determine if it has encountered an error. In specific driver functions, the status check should not occur in the lowest-level signal generation functions Prefix_Initiate, Prefix_Abort, and Prefix_Fetch, Prefix_FetchMultiPoint,and Prefix_SendSWTrigger. 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.

23.4 Special Considerations for IVIDMM_ATTR_SAMPLE_TRIGGER

Some of the simpler DMMs on the market implement in hardware a simplified version of the IviDmmMultiPoint state model. These DMMs still have the ability to specify IVIDMM_ATTR_TRIGGER_COUNT and IVIDMM_ATTR_SAMPLE_COUNT. However, they do not implement IVIDMM_ATTR_SAMPLE_TRIGGER and IVIDMM_ATTR_SAMPLE_INTERVAL. When IVIDMM_ATTR_SAMPLE_COUNT is greater than 1, these DMMs typically execute the trigger delay for each sample. Therefore, the behavior between simple and sophisticated DMMs can vary greatly when performing multipoint scanning.

If you implement the IviDmmMultiPoint extension group on instruments that do not have a IVIDMM_ATTR_SAMPLE_TRIGGER, you should do the following to be interchangeable with DMMs that fully support the extension:

1. Implement IVIDMM_ATTR_SAMPLE_TRIGGER with the only supported value IVIDMM_VAL_SAMPLE_INTERVAL.
2. Implement IVIDMM_ATTR_SAMPLE_INTERVAL where the only possible value is the present value for IVIDMM_ATTR_TRIGGER_DELAY.
3. Set the IVIDMM_ATTR_TRIGGER_DELAY attribute to invalidate the IVIDMM_ATTR_SAMPLE_INTERVAL attribute.

By following these guidelines, you will maximize interchangeable behavior between all DMMs.

23.5 Special Considerations for IVIDMM_ATTR_AUTO_RANGE_VALUE

The purpose of the attribute IVIDMM_ATTR_AUTO_RANGE_VALUE is to return the range that the instrument has auto-ranged to when the attribute IVIDMM_ATTR_RANGE is set to IVIDMM_VAL_AUTO_RANGE_ON. Since the value of IVIDMM_ATTR_AUTO_RANGE is likely to change as the input signal changes, drivers that may cache attributes should never cache this attribute.

24. Appendix B, Interchangeability Checking Guidelines

24.1 Introduction

IVI drivers have 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.

24.2 When to Perform Interchangeability Checking

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

- The `IVIDMM_ATTR_INTERCHANGE_CHECK` attribute is set to `VI_TRUE`
- The user calls one of the following functions.
  - `IviDmm_Initiate`
  - `IviDmm_Read`
  - `IviDmm_ReadMultiPoint`

24.3 Interchangeability Checking Rules

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

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

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

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

24.3.1 IviDmmBase Capability Group

If the `IVIDMM_ATTR_FUNCTION` attribute is set to `IVIDMM_VAL_TEMPERATURE`, the `IVIDMM_ATTR_RESOLUTION_ABSOLUTE` attribute is not required to be in a user specified state.

24.3.2 IviDmmACMeasurement Extension Group

If the IVIDMM_ATTR_FUNCTION attribute is not set to IVIDMM_VAL_AC_VOLTS, IVIDMM_VAL_AC_CURRENT, IVIDMM_VAL_AC_PLUS_DC_VOLTS, or IVIDMM_VAL_AC_PLUS_DC_CURRENT, then the following attributes are not required to be in a user specified state:

- IVIDMM_ATTR_AC_MIN_FREQ
- IVIDMM_ATTR_AC_MAX_FREQ

24.3.3 IviDmmFrequencyMeasurement Extension Group

If the IVIDMM_ATTR_FUNCTION attribute is not set to IVIDMM_VAL_FREQ or IVIDMM_VAL_PERIOD, then the IVIDMM_ATTR_FREQ_VOLTAGE_RANGE attribute is not required to be in a user specified state.

24.3.4 IviDmmTemperatureMeasurement Extension Group

If the IVIDMM_ATTR_FUNCTION attribute is not set to IVIDMM_VAL_TEMPERATURE, the IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE attribute is not required to be in a user specified state.

24.3.5 IviDmmThermocouple Extension Group

If the IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE attribute is not set to IVIDMM_VAL_THERMOCOUPLE, then the following attributes are not required to be in a user specified state:

- IVIDMM_ATTR_TEMP_TC_TYPE
- IVIDMM_ATTR_TEMP_TC_REF_JUNC_TYPE
- IVIDMM_ATTR_TEMP_TC_FIXED_REF_JUNC

24.3.6 IviDmmResistanceTemperatureDevice Extension Group

If the IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE attribute is not set to IVIDMM_VAL_2_WIRE_RTD or IVIDMM_VAL_4_WIRE_RTD, then the following attributes are not required to be in a user specified state:

- IVIDMM_ATTR_TEMP_RTD_ALPHA
- IVIDMM_ATTR_TEMP_RTD_RES

24.3.7 IviDmmThermistor Extension Group

If the IVIDMM_ATTR_TEMP_TRANSDUCER_TYPE attribute is not set to IVIDMM_VAL_THERMISTOR, the IVIDMM_ATTR_TEMP_THERMISTOR_RES attribute is not required to be in a user specified state.

24.3.8 IviDmmMultiPoint Extension Group

1. If the IVIDMM_ATTR_SAMPLE_COUNT attribute is set to 1, then the following attributes are not required to be in a user specified state:

   - IVIDMM_ATTR_SAMPLE_TRIGGER
   - IVIDMM_ATTR_SAMPLE_INTERVAL

2. If the IVIDMM_ATTR_SAMPLE_COUNT attribute is set 1 and the IVIDMM_ATTR_SAMPLE_TRIGGER attribute is set to a value other than IVIDMM_VAL_INTERVAL, then the IVIDMM_ATTR_SAMPLE_INTERVAL attribute is not required to be in a user specified state.

24.3.9 IviDmmTriggerSlope Extension Group

No additional interchangeability rules or exceptions are defined for the IviDmmTriggerSlope extension group.

24.3.10 IviDmmSoftwareTrigger Extension Group

No additional interchangeability rules or exceptions are defined for the IviDmmSoftwareTrigger extension group.

24.3.11 IviDmmDeviceInfo Extension Group

No additional interchangeability rules or exceptions are defined for the IviDmmDeviceInfo extension group.

24.3.12 IviDmmAutoRangeValue Extension Group

No additional interchangeability rules or exceptions are defined for the IviDmmAutoRangeValue extension group.

24.3.13 IviDmmAutoZero Extension Group

No additional interchangeability rules or exceptions are defined for the IviDmmAutoZero extension group.

24.3.14 IviDmmPowerLineFrequency Extension Group

No additional interchangeability rules or exceptions are defined for the IviDmmPowerLineFrequency extension group.