# IVI

## Interchangeable Virtual Instruments

# IVI-4.13: IviDownconverter Class Specification

June 9, 2010 Edition
Revision 2.0

# Important Information

The IviDownconverter Class Specification (IVI-4.13) is authored by the IVI Foundation member companies. For a vendor membership roster list, please visit the IVI Foundation web site at `www.ivifoundation.org`.

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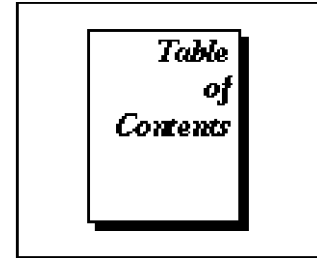
**Warranty**

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

**Trademarks**

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

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

# IviDownconverter Class Specification

## IviDownconverter Revision History

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

**Table 1-1-1** IviDownconverter Class Specification Revisions

| Revision Number | Date of Revision | Revision Notes |
|---|---|---|
| Revision 1.0 | October 21, 2009 | Version 1.0 of Specification Approved |
| Revision 2.0 | June 9, 2010 | Incorporated IVI.NET |

**API Versions**

| Architecture | Drivers that comply with version 2.0 comply with all of the versions below. |
|---|---|
| C | 1.0, 2.0 |
| COM | 1.0, 2.0 |
| .NET | 2.0 |

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

# 1. Overview of the IviDownconverter Specification

## 1.1 Introduction

This specification defines the IVI class for frequency downconverters. The IviDownconverter class is designed to support the typical downconverter as well as common extended functionality found in more complex instruments. This section summarizes the *IviDownconverter Class Specification* 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:

- IviDownconverter class overview

- The definitions of terms and acronyms

- References

## 1.2 IviDownconverter Class Overview

This specification defines the IVI class for downconverters. The IviDownconverter class is designed to support devices that convert RF input frequency signals to IF output frequency signals suitable for processing by a downstream system component, such as a digitizer.

The IviDownconverter class is divided into a base capability group and several extension groups. The base capability group is used to configure basic input and output filtering and gain/attenuation stages. The IviDownconverter base capability group is described in Section 4, *IviDownconverterBase Capability Group*.

Extension groups are provided for more advanced capabilities. Several extension groups are provided to support different styles of sweeping the RF input. Other extension groups provide more precise control of signal routing through the various downconverter stages.

## 1.3 References

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

- IVI-3.1: Driver Architecture Specification

- IVI-3.2: Inherent Capabilities Specification

- IVI-3.3: Standard Cross Class Capabilities Specification

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

- IVI- 5.0: Glossary

## 1.4 Definitions of Terms and Acronyms

This section defines terms and acronyms that are specific to the IviDownconverter class.

| | |
|---|---|
| Local Oscillator (LO) | An oscillator mixed with the RF input signal to produce the IF output. |
| Radio Frequency (RF) | Refers to the input frequency to be translated by the downconverter |
| Intermediate Frequency (IF) | Refers to the frequency output by the downconverter |

Refer to *IVI–5.0: Glossary* for a description of more terms used in this specification.

# 2. IviDownconverter Class Capabilities

## 2.1 Introduction

The IviDownconverter specification divides generic downconverter 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 IviDownconverter Group Names

The capability group names for the IviDownconverter 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 Class Group Capabilities attribute.

Table **2-1.** IviDownconverter Group Names

| Group Name | Description |
|---|---|
| IviDownconverterBase | Base Capabilities of the IviDownconverter specification.  This group includes the ability to set the RF input frequency, IF output frequency, input attenuation, output gain, and other control parameters. |
| IviDownconverterBypass | Extension:  IviDownconverter with the ability to have the RF input completely bypass the downconverter. |
| IviDownconverterExternalMixer | Extension:  IviDownconverter with the ability to use an external mixer. |
| IviDownconverterFrequencyStep | Extension:  IviDownconverter with the ability to sweep the RF input frequency in discrete steps. |
| IviDownconverterFrequencySweep | Extension:  IviDownconverter with the ability to sweep the RF input frequency. |
| IviDownconverterFrequencySweepList | Extension:  IviDownconverter with the ability to sweep the RF input frequency using a list of values. |
| IviDownconverterBandCrossingInformation | Extension:  IviDownconverter with the ability to return frequency band information for frequency sweeps. |
| IviDownconverterSoftwareTrigger | Extension: IviDownconverter with the ability to sweep the RF input based on a software trigger. |
| IviDownconverterIFFilter | Extension:  IviDownconverter with the ability to specify the IF output filter bandwidth. |
| IviDownconverterPreselector | Extension:  IviDownconverter with the ability to bypass the preselection filter. |
| IviDownconverterVideoDetectorBandwidth | Extension:  IviDownconverter with the ability to employ a video detection capability at the IF output. |
| IviDownconverterCalibration | Extension: IviDownconverter with the ability to perform self-calibration. |
| IviDownconverterReferenceOscillator | Extension: IviDownconverter with the ability to |

| Group Name | Description |
|---|---|
|  | use an external frequency reference. |

## *2.3 Repeated Capability Names*

The IviDownconverter Class Specification defines two repeated capabilities.  Refer to the sections of IVI-3.1: *Driver Architecture Specification* that deal with repeated capabilities.  They are Section 2.7, *Repeated Capabilities*, Section 4.1.9, *Repeated Capabilities*, Section 4.2.5, *Repeated Capabilities*, Section 4.3.9, *Repeated Capabilities*,  and Section 5.9, *Repeated Capability Identifiers and Selectors*.

- RFInput

- IFOutput

### 2.3.1 RFInput

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

All RF Input related attributes and functions operate exclusively on the active RF Input, controlled by setting the Active RF Input attribute prior to using any RF Input related attributes and functions.

### 2.3.2 IFOutput

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

All IF Output related attributes and functions operate exclusively on the active IF Output, controlled by setting the Active IF Output attribute prior to using any IF Output related attributes and functions.

## *2.4 Boolean Attribute and Parameter Values*

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

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

## *2.5 .NET Namespace*

The .NET namespace for the IviDownconverter class is Ivi.Downconverter.

## *2.6 .NET IviDownconverter Session Factory*

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

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

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

**.NET Method Prototype**

```
IIviDownconverter IviDownconverter.Create(String name);
IIviDownconverter IviDownconverter.Create(String name,
                                   Boolean idQuery,
                                   Boolean reset);
IIviDownconverter IviDownconverter.Create(String name,
                                   Boolean idQuery,
                                   Boolean reset,
                                   String options);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| name | A session name or a logical name that points to a session that uses an IVI.NET IviDownconverter class-compliant driver. | String |
| idQuery | Specifies whether to verify the ID of the instrument. The default is False. | Boolean |
| reset | Specifies whether to reset the instrument. The default is False. | Boolean |
| options | A string that allows the user to specify the initial values of certain inherent attributes. The default is an empty string. | String |

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

**.NET Exceptions**

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

**Usage**

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

```
IIviDownconverter downconverter = IviDownconverter.Create("MyLogicalName");
```

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

# 3. General Requirements

This section describes the general requirements a specific 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 IviDownconverter Class Specification, an IVI specific driver shall conform to all of the requirements for an IVI class-compliant specific driver as specified in *IVI-3.1: Driver Architecture Specification,* implement the inherent capabilities that *IVI-3.2: Inherent IVI Capabilities Specification* defines, and implements the IviDownconverterBase capability group.

### 3.1.1 Disable

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

The Disable function shall cause the downconverter to apply the minimum amount of power possible at the output terminals. Setting the IF output gain to a value close to zero or physically disconnecting the function generator from the output terminals meets this requirement. Other techniques are also allowed.

## 3.2 Capability Group Compliance

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

## 3.3 Initialize

In addition to the standard `OptionsString` attributes specified in Section 6.16, *Initialize* of *IVI-3.2: Inherent IVI Capabilities Specification*, the IviDownconverter Class Specification adds the following valid `OptionsString` attributes:

**Table 3-1.** IVI Downconverter Attribute Initial Values and Options String Name

| Attribute | Default Initial Value | Options String Name |
|---|---|---|
| External LO Enabled | False | ExternalLOEnabled |

If a driver does not support setting the External LO Enabled attribute to True, it should return the Invalid Value error.

# 4. IviDownconverterBase Capability Group

## 4.1 Overview

The IviDownconverterBase Capability Group supports basic downconverter operation. The IviDownconverterBase Capability Group defines attributes and functions for configuring and/or accessing RF input frequency, and attenuation as well as IF output frequency and gain. This Capability Group also defines functions for using an external local oscillator (LO).

This specification defines an RF Input repeated capability to accommodate devices that offer multiple RF inputs. Many devices, however, offer only a single RF input. The block diagram below presents the basic aspects of a single-input downconverter.
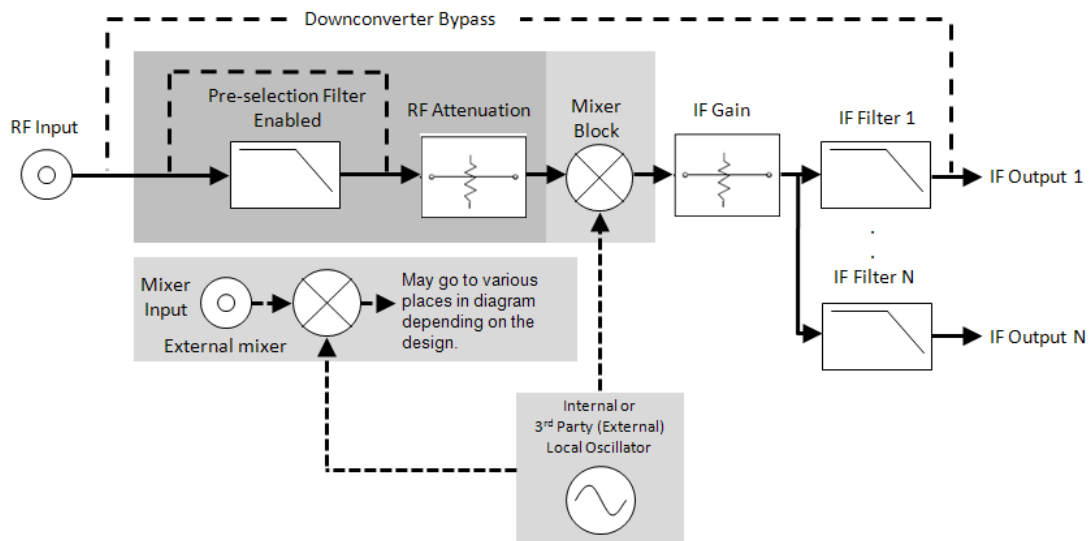


**Figure 4-1.** Single-Input IviDownconverter Block Diagram

The diagram above illustrates a downconverter that has a single RF input feeding the front-end elements of the device. Figure 4-1 shows a preselection filter followed by an RF attenuation stage. Depending upon the design of the downconverter, the order in which these elements appears may vary.

Some downconverters may offer multiple RF inputs along with multiple IF outputs. This specification does not dictate whether the IF outputs are dedicated to specific RF inputs or whether they are shared amongst the multiple RF inputs. Vendors are encouraged to use a repeated capability naming scheme that indicates whether or not IF outputs are associated with specific RF inputs.

The block diagram below shows one possible arrangement for a multi-input downconverter. In the diagram below, the RF inputs each feed dedicated sets of IF outputs. An alternate arrangement might consist of multiple inputs routed through an RF multiplexer into a set of shared IF outputs.
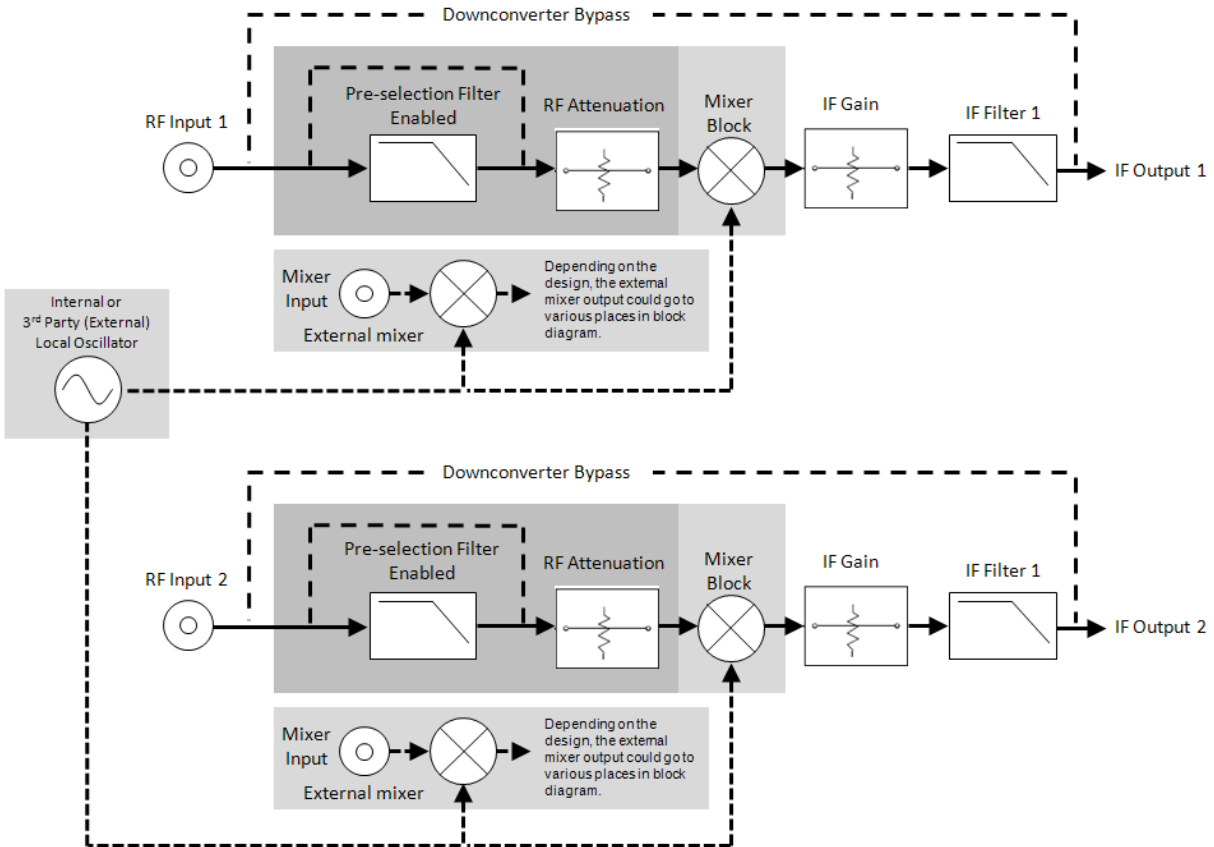
**Figure 4-2.** Multi-Input IviDownconverter Block Diagram

## 4.2 IviDownconverterBase Attributes

The IviDownconverterBase capability group defines the following attributes:

- Active IF Output
- Active RF Input
- External LO Enabled
- External LO Frequency
- IF Output Count
- IF Output Enabled
- IF Output Frequency
- IF Output Gain
- IF Output Name (IVI-COM Only)

- Is Settled

- RF Input Attenuation

- RF Input Corrections Enabled

- RF Input Count

- RF Input Coupling

- RF Input Frequency

- RF Input Name (IVI-COM Only)

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions.*

## 4.2.1 Active IF Output

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViString | R/W | N/A | None | Set Active IF Output (IVI-C Only) |

**.NET Property Name**

      IFOutput.ActiveIFOutput

**COM Property Name**

      IFOutput.ActiveIFOutput

**C Constant Name**

      IVIDOWNCONVERTER_ATTR_ACTIVE_IF_OUTPUT

**Description**

Specifies the IF output that is currently active. Subsequent calls to functions and attributes that are based on the IF Output repeated capability will be applied to the Active IF Output specified here. The values for this attribute correspond to the allowed repeated capability names for the IF Output repeated capability. If the driver defines a qualified IF Output name, this attribute returns the qualified name. Use the IF Output Name attribute (for IVI-COM) or the Get IF Output Name function (for IVI-C) to read the allowed values for this attribute.

Note that the Active IF Output attribute does not enable the specified output. This attribute only controls the IF Output repeated capability instance to which other functions and attributes apply. Use the IF Output Enabled attribute to route the IF signal to a specific output.

**.NET Exceptions**

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

## 4.2.2 Active RF Input

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|---------------------|
| ViString | R/W | N/A | None | Set Active RF Input (IVI-C Only) |

**.NET Property Name**

       RFInput.ActiveRFInput

**COM Property Name**

       RFInput.ActiveRFInput

**C Constant Name**

       IVIDOWNCONVERTER_ATTR_ACTIVE_RF_INPUT

**Description**

Specifies the RF input that is currently active. Subsequent calls to functions and attributes that are based on the RF Input repeated capability will be applied to the Active RF Input specified here. The values for this attribute correspond to the allowed repeated capability names for the RFInput repeated capability. If the driver defines a qualified RFInput name, this attribute returns the qualified name. Use the RF Input Name attribute (for IVI-COM) or the Get RF Input Name function (for IVI-C and IVI.NET) to read the allowed values for this attribute.

**.NET Exceptions**

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

## 4.2.3 External LO Enabled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | N/A | None | N/A |

**.NET Property Name**

> ExternalLO.Enabled

**COM Property Name**

> ExternalLO.Enabled

**C Constant Name**

> IVIDOWNCONVERTER_ATTR_EXTERNAL_LO_ENABLED

**Description**

> If True, the external LO is enabled.  If False, the external LO is disabled.  If the driver does not support an external LO and this attribute is set to True, then the driver returns the Invalid Value error.  As specified in Section 3.3, *Initialize* of this specification, the External LO Enabled attribute can also be set via the OptionsString parameter.

**.NET Exceptions**

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

## 4.2.4 External LO Frequency

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | N/A | None | N/A |

**.NET Property Name**

ExternalLO.Frequency

**COM Property Name**

ExternalLO.Frequency

**C Constant Name**

IVIDOWNCONVERTER_ATTR_EXTERNAL_LO_FREQUENCY

**Description**

Specifies the frequency of the external LO.  The units are Hertz.

External LO Frequency will normally be read-only so that the driver can supply the correct external LO frequency for a given requested RF Input frequency.  Some downconverters can support multiple LO frequencies for a single RF frequency to help minimize images.  For these cases, the External LO Frequency is writeable.

**Compliance Notes**

1.  It is acceptable to return an error if the External LO Frequency is set to a value not compatible with the requested RF Input frequency.

**.NET Exceptions**

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

## 4.2.5 IF Output Count

| Data Type | Access | Applies To | Coercion | High Level Functions |
|---|---|---|---|---|
| ViInt32 | RO | N/A | None | N/A |

**.NET Property Name**

IFOutput.Count

**COM Property Name**

IFOutput.Count

**C Constant Name**

IVIDOWNCONVERTER_ATTR_IF_OUTPUT_COUNT

**Description**

Returns the number of IF Outputs available on the device.

**.NET Exceptions**

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

## 4.2.6 IF Output Enabled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | IFOutput | None | Configure IF Output Enabled (IVI-C only) |

**.NET Property Name**

IFOutput.Enabled

**COM Property Name**

IFOutput.Enabled

**C Constant Name**

IVIDOWNCONVERTER_ATTR_IF_OUTPUT_ENABLED

**Description**

If True, the active IF output is enabled.  If False, the active IF output is disabled.  Only a single IF Output can be enabled at a time.  Thus, when this attribute is set to True for a particular IF output, all other IF outputs are disabled.

**.NET Exceptions**

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

## 4.2.7 IF Output Frequency

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | RO | IFOutput | None | N/A |

**.NET Property Name**

       IFOutput.Frequency

**COM Property Name**

       IFOutput.Frequency

**C Constant Name**

       IVIDOWNCONVERTER_ATTR_IF_OUTPUT_FREQUENCY

**Description**

       Returns the frequency of the active IF output.  The units are Hertz.

**.NET Exceptions**

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

## 4.2.8 IF Output Gain

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | IFOutput | Down | Configure IF Output Gain (IVI-C only) |

**.NET Property Name**

      IFOutput.Gain

**COM Property Name**

      IFOutput.Gain

**C Constant Name**

      IVIDOWNCONVERTER_ATTR_IF_OUTPUT_GAIN

**Description**

Specifies the amount of gain (or attenuation) to apply to the active IF output of the downconverter. The units are dB.

Positive values for this attribute represent signal gain while negative values represent attenuation.

**.NET Exceptions**

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

### 4.2.9 IF Output Name (IVI-COM Only)

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViString | RO | IFOutput | None | N/A |

**.NET Property Name**

    N/A

(Use the Get IF Output Name method.)

**COM Property Name**

    HRESULT IFOutput.Name ([in] LONG Index,
                            [out,retval] BSTR* Name);

**C Constant Name**

    N/A

(Use the Get IF Output Name function.)

**Description**

Returns the IF Output identifier that corresponds to the one-based index that the user specifies. If the driver defines a qualified IF Output name, this function returns the qualified name. If the value that the user passes for the Index parameter is less than one or greater than the value of the IF Output Count attribute, the property returns an empty string in the Name parameter and returns the Invalid Value error.

## 4.2.10 Is Settled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | RO | IFOutput | None | N/A |

**.NET Property Name**

      IFOutput.IsSettled

**COM Property Name**

      IFOutput.IsSettled

**C Constant Name**

      IVIDOWNCONVERTER_ATTR_IS_SETTLED

**Description**

Indicates whether the downconverter has settled from changes to either the RF input signal or changes to device control parameters, such as IF Output Gain, IF Frequency, or RF Attenuation.  This attribute indicates whether or not the active IF output is valid for processing by another downstream system component, such as a digitizer.

**.NET Exceptions**

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

## 4.2.11 RF Input Attenuation

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | RFInput | Up | Configure RF Input Attenuation (IVI-C Only) |

**.NET Property Name**

      RFInput.Attenuation

**COM Property Name**

      RFInput.Attenuation

**C Constant Name**

      IVIDOWNCONVERTER_ATTR_RF_INPUT_ATTENUATION

**Description**

Specifies the amount of attenuation (or gain) to apply to the active RF input of the downconverter. The units are dB.

Positive values for this attribute represent attenuation while negative values represent gain.

**.NET Exceptions**

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

## 4.2.12 RF Input Corrections Enabled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | RFInput | None | N/A |

**.NET Property Name**

RFInput.CorrectionsEnabled

**COM Property Name**

RFInput.CorrectionsEnabled

**C Constant Name**

IVIDOWNCONVERTER_ATTR_RF_INPUT_CORRECTIONS_ENABLED

**Description**

If True, the automatic global corrections on the device is enabled.  If False, the automatic global corrections on the device is disabled.

**.NET Exceptions**

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

## 4.2.13 RF Input Coupling

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViInt32 | R/W | RFInput | None | N/A |

**.NET Property Name**

RFInput.Coupling

**.NET Enumeration Name**

InputCoupling

**COM Property Name**

RFInput.Coupling

**COM Enumeration Name**

IviDownconverterInputCouplingEnum

**C Constant Name**

IVIDOWNCONVERTER_ATTR_RF_INPUT_COUPLING

**Description**

Specifies the coupling applied to active RF input.

**Defined Values**

| Name | Description | | |
|------|-------------|----------|------------|
| | | *Language* | *Identifier* |
| AC | The downconverter AC couples the RF input signal. | | |
| | | .NET | AC |
| | | C | IVIDOWNCONVERTER_VAL_INPUT_COUPLING_AC |
| | | COM | IviDownconverterInputCouplingAC |
| DC | The downconverter DC couples the RF input signal. | | |
| | | .NET | DC |
| | | C | IVIDOWNCONVERTER_VAL_INPUT_COUPLING_DC |
| | | COM | IviDownconverterInputCouplingDC |

**.NET Exceptions**

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

## 4.2.14 RF Input Count

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViInt32 | RO | N/A | None | N/A |

**.NET Property Name**

RFInput.Count

**COM Property Name**

RFInput.Count

**C Constant Name**

IVIDOWNCONVERTER_ATTR_RF_INPUT_COUNT

**Description**

Returns the number of RF Inputs available on the device.

**.NET Exceptions**

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

## 4.2.15 RF Input Frequency

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|-----------|----------|----------------------|
| ViReal64 | R/W | RFInput | None | Configure RF Input Frequency (IVI-C Only) |

**.NET Property Name**

      RFInput.Frequency

**COM Property Name**

      RFInput.Frequency

**C Constant Name**

      IVIDOWNCONVERTER_ATTR_RF_INPUT_FREQUENCY

**Description**

      Specifies the frequency of the active RF input.  The units are Hertz.

**.NET Exceptions**

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

## 4.2.16 RF Input Name (IVI-COM Only)

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViString  | RO     | RFInput    | None     | N/A                  |

**.NET Property Name**

N/A

(Use the Get RF Input Name function.)

**COM Property Name**

```
HRESULT RFInput.Name ([in] LONG Index,
                      [out,retval] BSTR* Name);
```

**C Constant Name**

N/A

(Use the Get RF Input Name function.)

**Description**

Returns the RF Input identifier that corresponds to the one-based index that the user specifies. If the driver defines a qualified RF Input name, this function returns the qualified name. If the value that the user pass for the Index parameter is less than one or greater than the value of the RF Input Count attribute, the property returns and empty string in the Name parameter and returns the Invalid Value error.

**.NET Exceptions**

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

## 4.3 IviDownconverterBase Functions

The IviDownconverterBase capability group defines the following functions:

- Configure IF Output Gain (IVI-C Only)
- Configure IF Output Enabled (IVI-C Only)
- Configure Reference Oscillator
- Configure RF Input Attenuation (IVI-C Only)
- Configure RF Input Frequency (IVI-C Only)
- Get IF Output Name (IVI-C & IVI.NET Only)
- Get RF Input Name (IVI-C & IVI.NET Only)
- Set Active IF Output (IVI-C Only)
- Set Active RF Input (IVI-C Only)
- Wait Until Settled

This section describes the behavior and requirements of each function.

## 4.3.1 Configure IF Output Gain (IVI-C Only)

**Description**

Configures the active IF output gain.

**.NET Method Prototype**

```
N/A
(use the IFOutput.Gain property)
```

**COM Method Prototype**

```
N/A
(use the IFOutput.Gain property)
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureIFOutputGain (ViSession Vi,
                                                 ViReal64 Gain);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Gain | Specifies the gain applied to the active IF output. The driver uses this value to set the IF Output Gain attribute. See the attribute description for more details. | ViReal64 |

**Return Values**

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

### 4.3.2 Configure IF Output Enabled (IVI-C Only)

**Description**

Configures whether or not the active IF output is enabled.

**.NET Method Prototype**

```
N/A
(use the IFOutput.Enabled property)
```

**COM Method Prototype**

```
N/A
(use the IFOutput.Enabled property)
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureIFOutputEnabled (ViSession Vi,
                                            ViBoolean Enabled);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Enabled | Enables or disables the active IF output.  The driver uses this value to set the IF Output Enabled attribute. See the attribute description for more details. | ViBoolean |

**Return Values**

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

### 4.3.3 Configure RF Input Attenuation (IVI-C Only)

**Description**

Configures the amount of attenuation applied to the active RF input.

**.NET Method Prototype**

```
N/A
(use the RFInput.Attenuation property)
```

**COM Method Prototype**

```
N/A
(use the RFInput.Attenuation property)
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureRFInputAttenuation (ViSession Vi,
                                              ViReal64 Attenuation);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Attenuation | Specifies the amount of attenuation to apply to the active RF Input. The driver uses this value to set the RF Input Attenuation 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 Configure RF Input Frequency (IVI-C Only)

**Description**

Configures the frequency of the active RF input.

**.NET Method Prototype**

```
N/A
(use the RFInput.Frequency property)
```

**COM Method Prototype**

```
N/A
(use the RFInput.Frequency property)
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureRFInputFrequency (ViSession Vi,
                                                ViReal64 Frequency);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Frequency | Specifies the frequency of the active RF input. The driver use this value to set the RF Input Frequency attribute. See the attribute description for more details. | ViReal64 |

**Return Values**

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

## 4.3.5 Get IF Output Name (IVI-C & IVI.NET Only)

**Description**

This function returns the specific driver defined IF Output name that corresponds to the one-based index that the user specifies.  If the driver defines a qualified IF Output name, this function returns the qualified name.  If the value that the user passes for the Index parameter is less than one or greater than the value of the IF Output Count, the function returns an empty string in the Name parameter and returns the Invalid Value error.

**.NET Method Prototype**

```
String IFOutput.GetName (Int32 index);
```

**COM Method Prototype**

```
N/A
(use the IFOutput.Name property)
```

**C Prototype**

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

**Parameters**

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

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

**Return Values (C)**

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

**.NET Exceptions**

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

## 4.3.6 Get RF Input Name (IVI-C & IVI.NET Only)

**Description**

This function returns the specific driver defined RF Input name that corresponds to the one-based index that the user specifies. If the driver defines a qualified RF Input name, this function returns the qualified name. If the value that the user passes for the `Index` parameter is less than one or greater than the value of the RF Input Count, the function returns an empty string in the `Name` parameter and returns the Invalid Value error.

**.NET Method Prototype**

**String RFInput.GetName (Int32 index);COM Method Prototype**

```
N/A
(use the RFInput.Name property)
```

**C Prototype**

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

**Parameters**

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

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

**Return Values (C)**

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

**.NET Exceptions**

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

### 4.3.7 Set Active IF Output (IVI-C Only)

**Description**

This function selects one of the available IF outputs, and makes it the active IF output.

**.NET Method Prototype**

```
N/A
(use the IFOutput.ActiveIFOutput property)
```

**COM Method Prototype**

```
N/A
(use the IFOutput.ActiveIFOutput property)
```

**C Prototype**

```
ViStatus IviDownconverter_SetActiveIFOutput (ViSession Vi,
                                             ViConstString Name);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Name | IF Output to be selected. The driver uses this value to set the Active IF Output attribute. See the attribute description for more details. | ViConstString |

**Return Values**

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

### 4.3.8 Set Active RF Input (IVI-C Only)

**Description**

This function selects one of the available RF inputs, and makes it the active RF input.

**.NET Method Prototype**

```
N/A
(use the RFInput.ActiveRFInput property)
```

**COM Method Prototype**

```
N/A
(use the RFInput.ActiveRFInput property)
```

**C Prototype**

```
ViStatus IviDownconverter_SetActiveRFInput (ViSession Vi,
                                            ViConstString Name);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Name | RF Input to be selected. The driver uses this value to set the Active RF Input attribute.  See the attribute description for more details. | ViConstString |

**Return Values**

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

### 4.3.9 Wait Until Settled

**Description**

This function waits until all of the signals flowing through the downconverter have settled. If the signals did not settle within the time period the user specified with the MaxTimeMilliseconds parameter, the function returns the Max Time Exceeded error.

**.NET Method Prototype**

```
void IFOutput.WaitUntilSettled (PrecisionTimeSpan maxTime);
```

**COM Method Prototype**

```
HRESULT IFOutput.WaitUntilSettled ([in] long MaxTimeMilliseconds);
```

**C Prototype**

```
ViStatus IviDownconverter_WaitUntilSettled (ViSession Vi,
                                   ViInt32 MaxTimeMilliseconds);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| MaxTimeMilliseconds (C/COM) | Specifies the maximum time the end-user allows for this function to complete. The units are milliseconds. Defined values: Max Time Immediate - The function returns immediately. If the downconverter has not settled the function returns an error. Max Time Infinite - The function waits indefinitely for the downconverter to settle. | ViInt32 |
| maxTime (.NET) | Specifies the maximum time the end-user allows for this function to complete. The units are implicit in the PrecisionTimeSpan type. Defined values: PrecisionTimeSpan.Zero - The function returns immediately. If the downconverter has not settled, the function returns an error. PrecisionTimeSpan.MaxValue - The function waits indefinitely for the downconverter to settle. | PrecisionTimeSpan |

**Defined Values for the MaxTimeMilliseconds Parameter (C/COM)**

| Name | Description | | |
|------|-------------|------|----------|
| | | Language | Identifier |
| Max Time Immediate | The function returns immediately. | | |
| | | C | IVIDOWNCONVERTER_VAL_MAX_TIME_IMMEDIATE |
| | | COM | IviDownconverterTimeOutImmediate |
| Max Time Infinite | The function waits indefinitely for the downconverter to settle. | | |

| | C | IVIDOWNCONVERTER_VAL_MAX_TIME_INFINITE |
|---|---|---|
| | COM | IviDownconverterTimeOutInfinite |

**Defined Values for the maxTime Parameter (.NET)**

| *Name* | *Description* | | |
|---|---|---|---|
| | | *Language* | *Identifier* |
| Zero | Sets timeout to immediate. The function returns immediately. | | |
| | | .NET | TimeSpan.Zero |
| MaxValue | Sets timeout to infinite. The function waits indefinitely for the settling to complete. | | |
| | | .NET | TimeSpan.MaxValue |

**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 |
|---|---|
| Max Time Exceeded | Error: Maximum time exceeded before the operation completed. |

**.NET Exceptions**

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method. The table below specifies additional class-defined exceptions for this method.

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

**Compliance Notes**

The specific instrument driver is not required to support any of the defined values for the MaxTimeMilliseconds parameter.

## 4.4 IviDownconverterBase Behavior Model

The following state diagram shows the basic operation of the IviDownconverter.
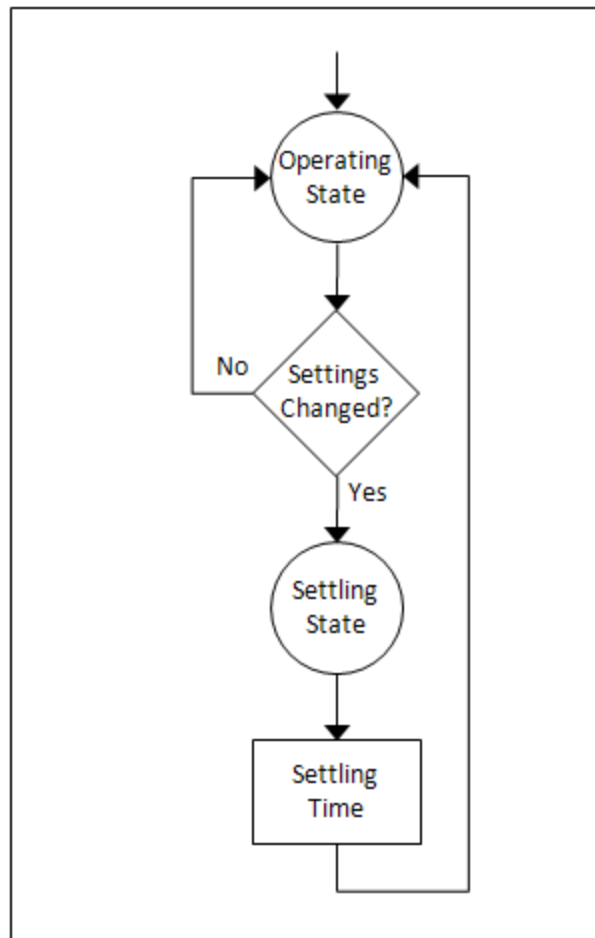


**Figure 4-3:** IviDownconverterBase Behavior Model

The behavior model for the IviDownconverterBase capability group is simple. The device is always operating (assuming power is applied) with an IF signal appearing at the output in response to an RF signal applied to the input. This specification defines a transient Settling state to account for the fact that the IF output signal may need to settle to a stable state after some operational parameters have been changed. Thus, in order to perform more accurate measurements, users will typically want to wait for a certain amount of time after changing some parameters to allow the signal to settle. A number of factors might require settling time, including (but not limited to) the following attributes defined by this specification:

- RF Input Attenuation

- RF Input Frequency

- IF Output Gain

# 5. IviDownconverterBypass Extension Group

## 5.1 IviDownconverterBypass Overview

The IviDownconverterBypass extension group supports downconverters with the ability to route the RF input completely around the downconverter. It defines a single attribute.

## 5.2 IviDownconverterBypass Attributes

The IviDownconverterBypass extension group defines the following attributes:

- Bypass

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions.*

## 5.2.1 Bypass

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | RFInput | None | Configure Bypass (IVI-C Only) |

**.NET Property Name**

       RFInput.Bypass

**COM Property Name**

       RFInput.Bypass

**C Constant Name**

       IVIDOWNCONVERTER_ATTR_BYPASS

**Description**

Specifies whether or not the RF input signal bypass the entire downconverter.

When set to True, the RF input signal is routed directly to the IF output indicated by the value of the Active IF Output attribute, and completely bypasses all stages of the downconverter. When set to False, the RF input signal does not bypass the downconverter - it is routed into the front end of the downconverter and follows the normal signal path, as dictated by other downstream path control attributes, such as Preselector Enabled.

**.NET Exceptions**

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

## 5.3 IviDownconverterBypass Functions

The IviDownconverterBypass extension group defines the following function:

- Configure Bypass (IVI-C Only)

This section describes the behavior and requirements of this function.

### 5.3.1 Configure Bypass (IVI-C Only)

**Description**

Configures whether or not the RF input signal completely bypasses the downconverter.

**.NET Method Prototype**

```
N/A
(use the RFInput.Bypass property)
```

**COM Method Prototype**

```
N/A
(use the RFInput.Bypass property)
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureBypass (ViSession Vi,
                                           ViBoolean Bypass);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Bypass | Enables or disables the downconverter bypass. The driver uses this value to set the Bypass attribute. See the attribute description for more details. | ViBoolean |

**Return Values**

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

## 5.4 IviDownconverterBypass Behavior Model

The IviDownconverterBypass extension group follows the same behavior model as the IviDownconverterBase group described in Section 4.4, *IviDownconverterBase Behavior Model*.

## 5.5 IviDownconverterBypass Compliance Notes

For a specific driver to comply with the IviDownconverterBypass extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 6. IviDownconverterExternalMixer Extension Group

## 6.1 IviDownconverterExternalMixer Overview
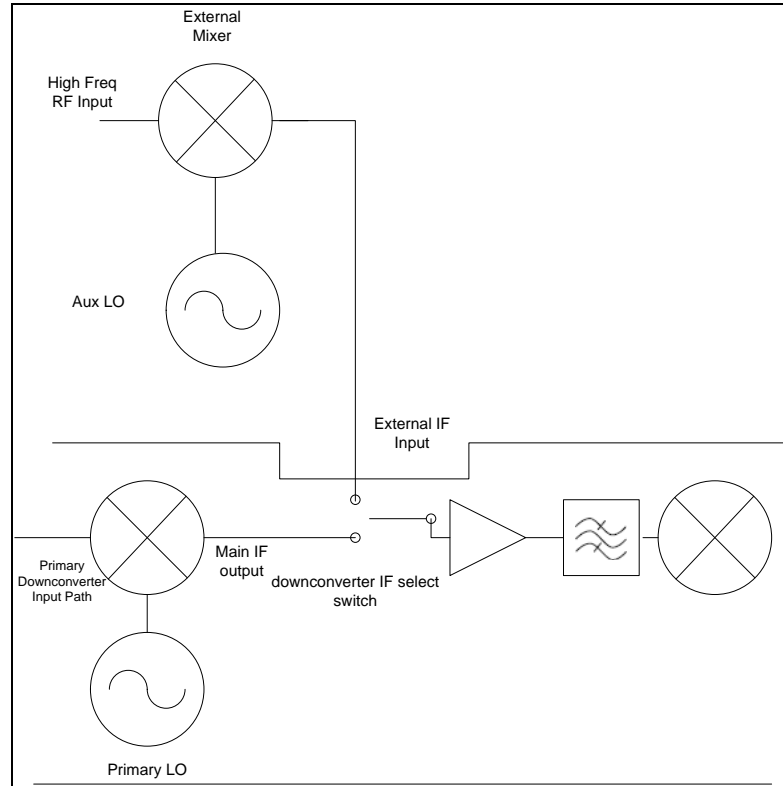


**Figure 6-4.  External mixer used with a downconverter's IF input.**

The IviDownconverterExternalMixer extension group supports using an external mixer. It defines attributes that allow you to enable/disable the mixer as well as control the harmonic and the bias level and limit. This extension group also defines functions that configure these attributes.

When using an external mixer, many of the settings of the downconverter have to be carefully converted to allow the user to know what is meant by the values read.  Specifically, the frequency, the harmonic number, mixer configuration, and conversion loss must be configured carefully to be able to use external mixing successfully.

The frequency of the input signal can be expressed as a function of the local oscillator (LO) frequency and the selected harmonic of the 1st LO is as follows:

$f_{in} = n * f_{LO} +/- f_{IF}$

Where: $f_{in}$    frequency of input signal

$n$    order of harmonic used for conversion

$f_{LO}$    frequency of 1st LO

$f_{IF}$    intermediate frequency

The Harmonic number defines the order n of the harmonic used for conversion.  Although both even and odd harmonics can be used, the harmonic is generally odd in balanced mixers.  Even order terms are naturally attenuated.  The selected harmonic, together with the setting range of the 1st LO, determines the limits of the settable frequency range.  The following applies:

Lower frequency limit:     $f_{min} = n * f_{LO,min} - f_{IF}$

Upper frequency limit:     $f_{max} = n * f_{LO,max} + f_{IF}$

Where:  $f_{LO,min}$    lower frequency limit of LO

$f_{LO,max}$    upper frequency limit of LO

The following sections describe the mixer configuration and the conversion loss table configuration.

## 6.1.1 Mixer Configuration

The external mixers are typically configured either as two-port or three port devices. Single-diode mixers generally require a DC voltage which is applied via the LO line. This DC voltage is to be tuned to the minimum conversion loss versus frequency. Some instruments can define a limit for the BIAS current.

The two-port mixer connects the 'LO OUT / IF IN' output of the analyzer to the LO/IF port of the external mixer. The diplexer is contained in the analyzer and the IF signal can be tapped from the line which is used to feed the LO signal to the mixer. The signal to be measured is fed to the RF input of the external mixer.

On the other hand, the three-port mixer connects the 'LO OUT / IF IN' output of the analyzer to the LO port of the external mixer. The 'IF IN' input of the analyzer is connected to the IF port of the external mixer. The signal to be measured is fed to the RF input of the external mixer.

## 6.1.2 Conversion Loss

The maximum input level depends on the external mixer's conversion loss, the LO drive level to the mixer, and the mixer configuration. Conversion loss is defined as the difference between a test signal directly connected to the RF input of the downconverter (or the IF input) to the signal strength at the output of the external mixer's IF output port for the same drive level. For example, let the desired IF signal be -20dBm at 250MHz. If an external mixer downconverted a -20dBm 30GHz signal to 250MHz but at -44dBm, it would have a conversion loss of 24dB.

In general the maximum input signal to an external mixer's input, the RF port, should be 15dB lower than the mixer's LO drive level to insure proper, linear operation. An external mixer with a +13dBm LO drive level should not have an input above -2dBm, for example. External amplifiers and attenuators could affect this level. Operating well below the LO drive level eliminates RF input compression on the main signal. Although the system will usually have filtering to minimize unwanted mixing products,  and will still downconvert larger signals, overdriven input signals can be unfaithfully reproduced at the IF output.

Some instruments allow the definition of conversion loss tables. The Conversion loss table allows the conversion loss of the mixer in the selected band to be taken into account as a function of frequency. Correction values for frequencies between the individual reference values are obtained by interpolation (Linear interpolation). Outside the frequency range covered by the table the conversion loss is assumed to be the same as that for the reference value marking the table limit.

## 6.2 IviDownconverterExternalMixer Attributes

The IviDownconverterExternalMixer extension group defines the following attributes:

- External Mixer Bias Enabled
- External Mixer Bias Level
- External Mixer Bias Limit
- External Mixer Enabled
- External Mixer Harmonic
- External Mixer Number Of Ports

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions*.

## 6.2.1 External Mixer Bias Enabled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | N/A | None | N/A |

**.NET Property Name**

ExternalMixer.Bias.Enabled

**COM Property Name**

ExternalMixer.Bias.Enabled

**C Constant Name**

IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_BIAS_ENABLED

**Description**

If true, the external mixer's bias is enabled.  If false, the external mixer's bias is disabled.

**.NET Exceptions**

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

## 6.2.2 External Mixer Bias Level

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | N/A | None | Configure External Mixer Bias |

**.NET Property Name**

ExternalMixer.Bias.Level

**COM Property Name**

ExternalMixer.Bias.Level

**C Constant Name**

IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_BIAS_LEVEL

**Description**

Specifies the external mixer bias current.  The units are Amps.

**.NET Exceptions**

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

### 6.2.3 External Mixer Bias Limit

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|---------------------|
| ViReal64 | R/W | N/A | None | Configure External Mixer Bias |

**.NET Property Name**

     ExternalMixer.Bias.Limit

**COM Property Name**

     ExternalMixer.Bias.Limit

**C Constant Name**

     IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_BIAS_LIMIT

**Description**

     Specifies the external mixer bias current limit.  The units are Amps.

**.NET Exceptions**

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

## 6.2.4 External Mixer Enabled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | N/A | None | N/A |

**.NET Property Name**

> ExternalMixer.Enabled

**COM Property Name**

> ExternalMixer.Enabled

**C Constant Name**

> IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_ENABLED

**Description**

> If true, the external mixer is enabled.  If false, the external mixer is disabled.

**.NET Exceptions**

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

## 6.2.5 External Mixer Harmonic

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

**.NET Property Name**

  ExternalMixer.Harmonic

**COM Property Name**

  ExternalMixer.Harmonic

**C Constant Name**

  IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_HARMONIC

**Description**

  Specifies the order n of the harmonic used for conversion.

**.NET Exceptions**

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

## 6.2.6 External Mixer Number Of Ports

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

**.NET Property Name**

>     ExternalMixer.NumberOfPorts

**COM Property Name**

>     ExternalMixer.NumberOfPorts

**C Constant Name**

>     IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_NUMBER_OF_PORTS

**Description**

>     Specifies the number of ports.

**.NET Exceptions**

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

## 6.3 IviDownconverterExternalMixer Functions

The IviDownconverterExternalMixer extension group defines the following function:

- Configure External Mixer Bias

This section describes the behavior and requirements of this function.

### 6.3.1 Configure External Mixer Bias

**Description**

    This function configures the external mixer bias and the external mixer bias limit.

**.NET Method Prototype**

```
void ExternalMixer.Bias.Configure (Double bias,
                                   Double biasLimit);
```

**COM Method Prototype**

```
HRESULT ExternalMixer.Bias.Configure ([in] double Bias,
                                      [in] double BiasLimit);
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureExternalMixerBias (ViSession Vi,
                                                      ViReal64 Bias,
                                                      ViReal64 BiasLimit);
```

**Parameters**

| Inputs | Description | Base Type |
|---|---|---|
| Vi | Instrument handle. | ViSession |
| Bias | Specifies the bias current. The driver uses this value to set the External Mixer Bias Level attribute.  See the attribute description for more details. | ViReal64 |
| BiasLimit | Specifies the bias current limit. The driver uses this value to set the External Mixer Bias Limit attribute.  See the attribute description for more details. | ViReal64 |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

## 6.4 IviDownconverterExternalMixer Behavior Model

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

## 6.5 IviDownconverterExternalMixer Compliance Notes

For a specific driver to comply with the IviDownconverterExternalMixer extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 7. IviDownconverterFrequencyStep Extension Group

## 7.1 IviDownconverterFrequencyStep Overview

The IviDownconverterFrequencyStep Extension Group supports downconverters that can vary (sweep) the frequency of the RF input signal in steps. The user can specify the start, stop and step frequency and set linear or logarithmic spacing. Setting single step and dwell time are also included.
This extension group requires the Frequency Sweep Extension Group. Frequency stepping is enabled by setting the Frequency Sweep Mode to Frequency Step in the IviDownconverterFrequencySweep Extension Group.

## 7.2 IviDownconverterFrequencyStep Attributes

The IviDownconverterFrequencyStep extension group defines the following attributes:

- Frequency Step Dwell
- Frequency Step Scaling
- Frequency Step Single Step Enabled
- Frequency Step Size
- Frequency Step Start
- Frequency Step Stop

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions.*

## 7.2.1 Frequency Step Dwell

| Data Type | Access | Applies To | Coercion | High Level Functions |
|---|---|---|---|---|
| ViReal64 (C/COM) PrecisionTimeSpan (.NET) | R/W | N/A | None | Configure Frequency Step Dwell |

**.NET Property Name**

RFInput.FrequencySweep.Step.Dwell

**COM Property Name**

RFInput.FrequencySweep.Step.Dwell

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_DWELL

**Description**

Specifies the duration of one step.  For C and COM, the units are seconds.  For .NET, the units are implicit in the PrecisionTimeSpan class.

The dwell time starts immediately at the start of each step. No settling time is added. This attribute is ignored if the Frequency Step Single Step Enabled attribute is set to True.

**.NET Exceptions**

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

## 7.2.2 Frequency Step Scaling

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|-----------|----------|----------------------|
| ViInt32 | R/W | N/A | None | Configure Frequency Step Start Stop |

**.NET Property Name**

RFInput.FrequencySweep.Step.Scaling

**.NET Enumeration Name**

FrequencyStepScaling

**COM Property Name**

RFInput.FrequencySweep.Step.Scaling

**COM Enumeration Name**

IviDownconverterFrequencyStepScalingEnum

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_SCALING

**Description**

Specifies the spacing of the steps.

**Defined Values**

| Name | Description | | |
|------|-------------|---|---|
| | | *Language* | *Identifier* |
| Linear | Enables linear scaling for step sizes in stepped sweeps. | | |
| | | .NET | Linear |
| | | C | IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_LINEAR |
| | | COM | IviDownconverterFrequencyStepScalingLinear |
| Logarithmic | Enables logarithmic scaling for step sizes in stepped sweeps. | | |
| | | .NET | Logarithmic |
| | | C | IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_LOGARITHMIC |
| | | COM | IviDownconverterFrequencyStepScalingLogarithmic |

**.NET Exceptions**

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

### 7.2.3 Frequency Step Single Step Enabled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | N/A | None | Configure Frequency Step Dwell |

**.NET Property Name**

RFInput.FrequencySweep.Step.SingleStepEnabled

**COM Property Name**

RFInput.FrequencySweep.Step.SingleStepEnabled

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_SINGLE_STEP_ENABLED

**Description**

If true, single step mode is enabled.  If false, single step mode is disabled.

**.NET Exceptions**

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

## 7.2.4 Frequency Step Size

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | N/A | None | Configure Frequency Step Start Stop |

**.NET Property Name**

RFInput.FrequencySweep.Step.Size

**COM Property Name**

RFInput.FrequencySweep.Step.Size

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_SIZE

**Description**

Specifies the step size. The units are Hertz if the Frequency Step Scaling attribute is set to Linear and unitless (factor) if the Frequency Step Scaling attribute is set to Logarithmic.

**.NET Exceptions**

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

## 7.2.5 Frequency Step Start

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | N/A | None | Configure Frequency Step Start Stop |

**.NET Property Name**

RFInput.FrequencySweep.Step.Start

**COM Property Name**

RFInput.FrequencySweep.Step.Start

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_START

**Description**

Specifies the start frequency of the stepped sweep. If the stop frequency is less than the start frequency, the frequency decreases during the sweep. The units are Hertz.

**.NET Exceptions**

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

## 7.2.6 Frequency Step Stop

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64  | R/W    | N/A        | None     | Configure Frequency Step Start Stop |

**.NET Property Name**

> RFInput.FrequencySweep.Step.Stop

**COM Property Name**

> RFInput.FrequencySweep.Step.Stop

**C Constant Name**

> IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_STOP

**Description**

> Specifies the stop frequency of the stepped sweep.
>
> If the stop frequency is less than the start frequency, the frequency decreases during the sweep. The units are Hertz.

**.NET Exceptions**

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

## 7.3 IviDownconverterFrequencyStep Functions

The IviDownconverterFrequencyStep extension group defines the following function:

- Configure Frequency Step Dwell

- Configure Frequency Step Start Stop

- Reset Frequency Step

This section describes the behavior and requirements of this function.

### 7.3.1 Configure Frequency Step Dwell

**Description**

Configures the attributes that control frequency stepping dwell.

**.NET Method Prototype**

```
void RFInput.FrequencySweep.Step.ConfigureDwell (Boolean singleStepEnabled,
                                                  Double dwell);
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.Step.ConfigureDwell (
                                        [in] VARIANT_BOOL SingleStepEnabled,
                                        [in] double Dwell);
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureFrequencyStepDwell ( ViSession Vi,
                                        ViBoolean SingleStepEnabled,
                                        ViReal64 Dwell);
```

**Parameters**

| Inputs | Description | Base Type |
|---|---|---|
| Vi | Instrument handle. | ViSession |
| SingleStepEnabled | Specifies whether the trigger initiates the next step. The driver uses this value to set the Frequency Step Single Step Enabled attribute. See the attribute description for more details. | ViBoolean |
| Dwell | Specifies the duration of one frequency step. The driver uses this value to set the Frequency Step Dwell attribute. See the attribute description for more details. | ViReal64 |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

## 7.3.2 Configure Frequency Step Start Stop

**Description**

Configures the attributes that control the step frequencies of the downconverter's input frequency.  These attributes are start and stop frequency, step size and lin/log scaling. If the stop frequency is less than the start frequency, the frequency decreases during the sweep.

**.NET Method Prototype**

```
void RFInput.FrequencySweep.Step.ConfigureStartStop (
                                    Double start,
                                    Double stop,
                                    FrequencyStepScaling scaling,
                                    Double stepSize);
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.Step.ConfigureStartStop ( [in] double Start,
    [in] double Stop,
    [in] IviDownconverterFrequencyStepScalingEnum Scaling,
    [in] double StepSize);
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureFrequencyStepStartStop ( ViSession Vi,
                                    ViReal64 Start,
                                    ViReal64 Stop,
                                    ViInt32 Scaling,
                                    ViReal64 StepSize);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Start | Specifies the start frequency of the sweep. The driver uses this value to set the Frequency Step Start attribute. See the attribute description for more details. | ViReal64 |
| Stop | Specifies the stop frequency of the sweep. The driver uses this value to set the Frequency Step Stop attribute. See the attribute description for more details. | ViReal64 |
| Scaling | Specifies the scaling of the step sweep. The driver uses this value to set the Frequency Step Scaling attribute. See the attribute description for more details. | ViInt32 |
| StepSize | Specifies the size of one step. The driver uses this value to set the Frequency Step Size attribute. See the attribute description for more details. | ViReal64 |

**Defined Values for the Scaling Parameter**

| *Name* | *Description* | | |
|---|---|---|---|
| | | *Language* | *Identifier* |
| Linear | Enables linear scaling for step sizes in stepped sweeps. | | |
| | | .NET | Linear |
| | | C | IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_LINEAR |
| | | COM | IviDownconverterFrequencyStepScalingLinear |
| Logarithmic | Enables logarithmic scaling for step sizes in stepped sweeps. | | |
| | | .NET | Logarithmic |
| | | C | IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_LOGARITHMIC |
| | | COM | IviDownconverterFrequencyStepScalingLogarithmic |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

### 7.3.3 Reset Frequency Step

**Description**

Resets the current frequency step to the frequency step start value

**.NET Method Prototype**

```
void RFInput.FrequencySweep.Step.Reset ();
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.Step.Reset ();
```

**C Prototype**

```
ViStatus IviDownconverter_ResetFrequencyStep (ViSession Vi);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

## 7.4 IviDownconverterFrequencyStep Behavior Model

The following state diagram shows the behavior of the IviDownconverterFrequencyStep extension group.
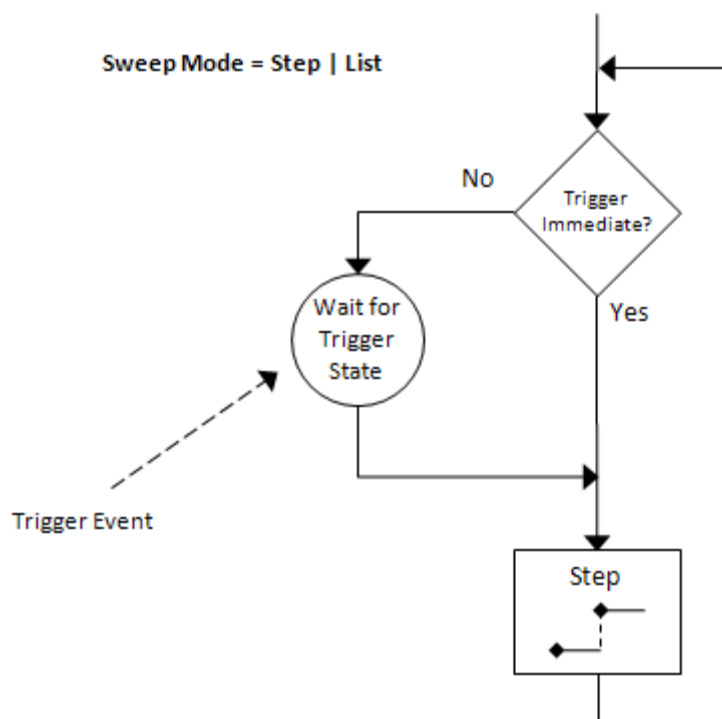


**Figure 7-1.** IviDownconverterFrequencyStep Behavior Model

Setting the Frequency Sweep Trigger Source attribute to "Immediate" will continuously generate frequency steps from start to stop with fixed increments (steps). The duration of one step is defined with Frequency Step Dwell Time. Setting the Frequency Sweep Trigger Source attribute to "External" or "Software" will delay the start of the next step until the specified trigger event occurs. This is shown in Figure 7-2 below.
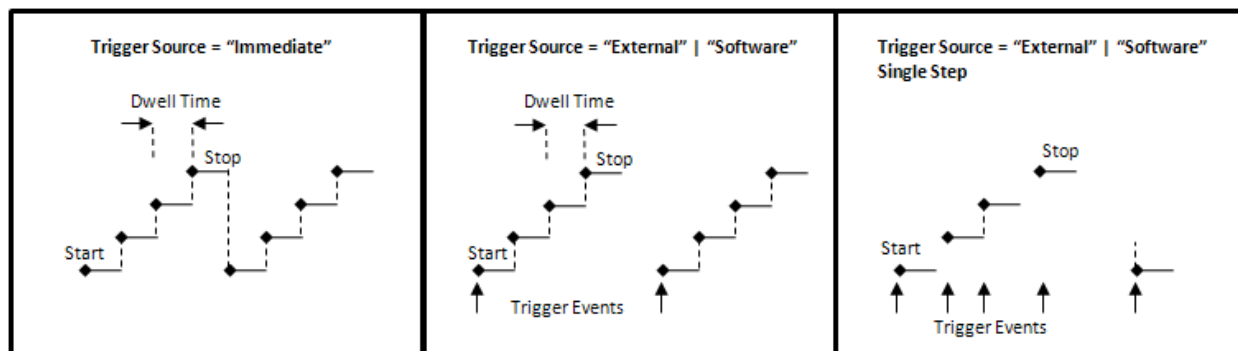
**Figure 7-2.** Frequency StepTiming

In addition the following rules apply:

1. When a call to Configure RF Input Frequency of the base capability group is made, the Frequency Sweep Mode attribute is set to None, hence stopping any sweeps that may have been in progress.

2. When the Frequency Sweep Mode attribute is set to Frequency Step, setting the RF Input Frequency attribute of the base capability group will set the Frequency Sweep Mode attribute to None.

3. When the Frequency Sweep Mode attribute is changed from any sweep mode to None, the downconverter will use the previously set value of the RF Input Frequency attribute of the base capability group.

## 7.5 IviDownconverterFrequencyStep Compliance Notes

For a specific driver to comply with the IviDownconverterFrequencyStep extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 8. IviDownconverterFrequencySweep Extension Group

## 8.1 IviDownconverterFrequencySweep Overview

The IviDownconverterFrequencySweep Extension Group supports downconverters that can apply a sweep to the RF input frequency.  The user may configure the sweep with start and stop frequencies and may also set the sweep time.

## 8.2 IviDownconverterFrequencySweep Attributes

The IviDownconverterFrequencySweep extension group defines the following attributes:

- Frequency Sweep Mode

- Frequency Sweep Start

- Frequency Sweep Stop

- Frequency Sweep Time

- Frequency Sweep Trigger Source

- Is Sweeping

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions.*

## 8.2.1 Frequency Sweep Mode

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|-----------|----------|---------------------|
| ViInt32 | R/W | N/A | None | Configure Frequency Sweep |

**.NET Property Name**

RFInput.FrequencySweep.Mode

**.NET Enumeration Name**

FrequencySweepMode

**COM Property Name**

RFInput.FrequencySweep.Mode

**COM Enumeration Name**

IviDownconverterFrequencySweepModeEnum

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_MODE

**Description**

Specifies the sweep mode of the RF input signal.

**Defined Values**

| Name | Description | |
|------|-------------|---|
| | *Language* | *Identifier* |
| None | The RF input of the downconverter is a non-swept signal (continuous wave). Frequency settings from the base capability group are used. | |
| | .NET | None |
| | C | IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_NONE |
| | COM | IviDownconverterFrequencySweepModeNone |
| Sweep | The downconverter sweeps the RF input signal frequency in analog form (non-stepped). Refer to IviDownconverterAnalogyFrequencySweep extension group. | |
| | .NET | Sweep |
| | C | IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_SWEEP |
| | COM | IviDownconverterFrequencySweepModeSweep |
| Step | The downconverter sweeps the RF input signal frequency in steps. Refer to IviDownconverterFrequencyStep extension group. | |
| | .NET | Step |
| | C | IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_STEP |

| | COM | IviDownconverterFrequencySweepModeStep |
|---|---|---|
| List | The downconverter uses a list to sweep the RF input signal frequency. Refer to IviDownconverterFrequencySweepList extension group. | |
| | .NET | List |
| | C | IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_LIST |
| | COM | IviDownconverterFrequencySweepModeList |

**Compliance Notes**

1. The driver shall implement the None value for this attribute.

2. If an IVI Class-Compliant specific driver implements any of the defined values in the following table, it shall also implement the corresponding capability group:

| Value | Required Capability Group |
|---|---|
| Sweep | IviDownconverterFrequencySweep |
| Step | IviDownconverterFrequencyStep |
| List | IviDownconverterFrequencyList |

**.NET Exceptions**

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

## 8.2.2 Frequency Sweep Start

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64  | R/W    | N/A        | None     | Configure Frequency Sweep Start Stop |

**.NET Property Name**

RFInput.FrequencySweep.Analog.Start

**COM Property Name**

RFInput.FrequencySweep.Analog.Start

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_START

**Description**

Specifies the start frequency of the LO sweep. If the stop frequency is less than the start frequency, the frequency decreases during the sweep. The units are Hertz

**.NET Exceptions**

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

### 8.2.3 Frequency Sweep Stop

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | N/A | None | Configure Frequency Sweep Start Stop |

**.NET Property Name**

> RFInput.FrequencySweep.Analog.Stop

**COM Property Name**

> RFInput.FrequencySweep.Analog.Stop

**C Constant Name**

> IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_STOP

**Description**

> Specifies the stop frequency of the LO sweep. If the stop frequency is less than the start frequency, the frequency decreases during the sweep. The units are Hertz.

**.NET Exceptions**

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

## 8.2.4 Frequency Sweep Time

| Data Type | Access | Applies To | Coercion | High Level Functions |
|---|---|---|---|---|
| ViReal64 (C/COM) PrecisionTimeSpan (.NET) | R/W | N/A | None | Configure Frequency Sweep Time (IVI-C Only) |

**.NET Property Name**

RFInput.FrequencySweep.Analog.Time

**COM Property Name**

RFInput.FrequencySweep.Analog.Time

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_TIME

**Description**

Specifies the duration of one LO sweep from start to stop frequency. For C and COM, the units are seconds. For IVI.NET, the units are implicit in the PrecisionTimeSpan class.

**.NET Exceptions**

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

## 8.2.5 Frequency Sweep Trigger Source

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|-----------|----------|----------------------|
| ViString | R/W | N/A | None | Configure Frequency Sweep |

**.NET Property Name**

RFInput.FrequencySweep.TriggerSource

**COM Property Name**

RFInput.FrequencySweep.TriggerSource

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_TRIGGER_SOURCE

**Description**

Specifies the trigger used to start an LO sweep operation.

**Defined Values**

In IVI.NET the trigger source is a string. If an IVI driver supports a trigger source and the trigger source is listed in IVI-3.3 *Cross Class Capabilities Specification*, Section 3 then the IVI driver shall accept the standard string for that trigger source. This attribute is case insensitive, but case preserving. That is the setting is case insensitive but when reading it back the programmed case is returned. IVI specific drivers may define new trigger source strings for trigger sources that are not defined by IVI-3.3 *Cross Class Capabilities Specification* if needed.

**Compliance Notes**

1. The specific driver need not support all of the trigger source values defined in *Chapter 18: IviDownconverter Attribute Value Definitions.*

**.NET Exceptions**

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

## 8.2.6 Is Sweeping

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | RO | N/A | None | N/A |

**.NET Property Name**

RFInput.FrequencySweep.IsSweeping

**COM Property Name**

RFInput.FrequencySweep.IsSweeping

**C Constant Name**

IVIDOWNCONVERTER_ATTR_IS_SWEEPING

**Description**

Indicates if the downconverter is currently sweeping the RF input frequency.

**.NET Exceptions**

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

## 8.3 IviDownconverterFrequencySweep Functions

The IviDownconverterFrequencySweep extension group defines the following function:

- Configure Frequency Sweep
- Configure Frequency Sweep Start Stop
- Configure Frequency Sweep Time (IVI-C Only)
- Wait Until Frequency Sweep Complete

This section describes the behavior and requirements of this function.

## 8.3.1 Configure Frequency Sweep

**Description**

Configures the whether the downconverter's RF input frequency is fixed, swept, or stepped.

**.NET Method Prototype**

```
void RFInput.FrequencySweep.Configure (FrequencySweepMode mode,
                                       String triggerSource);
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.Configure (
                       [in] IviDownconverterFrequencySweepModeEnum Mode,
                       [in] BSTR TriggerSource);
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureFrequencySweep (ViSession Vi,
                                       ViInt32 Mode,
                                       ViConstString TriggerSource);
```

**Parameters**

| Inputs | Description | Base Type |
|---|---|---|
| Vi | Instrument handle. | ViSession |
| Mode | Specifies the frequency sweep mode of the downconverter. The driver uses this value to set the Frequency Sweep Mode attribute. See the attribute description for more details. | ViInt32 |
| TriggerSource | Specifies the way to trigger source used to start a sweep or whether the sweep should run continuously. The driver uses this value to set the Frequency Sweep Trigger Source attribute. See the attribute description for more details. | ViConstString |

**Defined Values for the Mode Parameter**

| Name | Description | |
|---|---|---|
| | *Language* | *Identifier* |
| None | The RF input of the downconverter is a non-swept signal (continuous wave). Frequency settings from the base capability group are used. | |
| | .NET | None |
| | C | IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_NONE |
| | COM | IviDownconverterFrequencySweepModeNone |
| Sweep | The downconverter sweeps the RF input signal frequency in analog form (non-stepped). Refer to IviDownconverterAnalogyFrequencySweep extension group. | |
| | .NET | Sweep |
| | C | IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_SWEEP |
| | COM | IviDownconverterFrequencySweepModeSweep |

| Step | The downconverter sweeps the RF input signal frequency in steps. Refer to IviDownconverterFrequencyStep extension group. | | |
|---|---|---|---|
| | .NET | `Step` | |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_STEP` | |
| | COM | `IviDownconverterFrequencySweepModeStep` | |
| List | The downconverter uses a list to sweep the RF input signal frequency. Refer to IviDownconverterFrequencySweepList extension group. | | |
| | .NET | `List` | |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_LIST` | |
| | COM | `IviDownconverterFrequencySweepModeList` | |

**Defined Values for the TriggerSource Parameter**

The defined values for this parameter are listed with the Frequency Sweep Trigger Source attribute.

**Return Values (C/COM)**

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

**.NET Exceptions**

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

### 8.3.2 Configure Frequency Sweep Start Stop

**Description**

Configures the start and stop frequency attributes that control the frequency sweep of the RF input signal. If the stop frequency is less than the start frequency, the frequency decreases during the sweep.

**.NET Method Prototype**

```
void RFInput.FrequencySweep.Analog.ConfigureStartStop (Double start,
                                                        Double stop);
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.Analog.ConfigureStartStop ([in] double Start,
                                        [in] double Stop);
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureFrequencySweepStartStop (ViSession Vi,
                                        ViReal64 Start,
                                        ViReal64 Stop);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Start | Specifies the start frequency of the sweep. The driver uses this value to set the Frequency Sweep Start attribute. See the attribute description for more details. | ViReal64 |
| Stop | Specifies the stop frequency of the sweep. The driver uses this value to set the Frequency Sweep Stop attribute. See the attribute description for more details. | ViReal64 |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

### 8.3.3 Configure Frequency Sweep Time (IVI-C Only)

**Description**

Sets the duration of one frequency sweep.

**.NET Method Prototype**

```
N/A
```
(use the RFInput.FrequencySweep.Analog.Time property)

**COM Method Prototype**

```
N/A
```
(use the RFInput.FrequencySweep.Analog.Time property)

**C Prototype**

```
ViStatus IviDownconverter_ConfigureFrequencySweepTime (ViSession Vi,
                                        ViReal64 SweepTime);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| SweepTime | Specifies the duration of the frequency sweep. The driver uses this value to set the Frequency Sweep Time 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.

## 8.3.4 Wait Until Frequency Sweep Complete

**Description**

Waits until the configured frequency sweep is complete.  If no frequency sweep is currently running, this function returns immediately.   If the sweep does not complete  within the time period the user specified with the MaxTimeMilliseconds parameter, the function returns the Max Time Exceeded error.

**.NET Method Prototype**

```
void RFInput.FrequencySweep.WaitUntilComplete (PrecisionTimeSpan maxTime);
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.WaitUntilComplete (
                                    [in] long MaxTimeMilliseconds);
```

**C Prototype**

```
ViStatus IviDownconverter_WaitUntilFrequencySweepComplete (ViSession Vi,
                                    ViInt32 MaxTimeMilliseconds);
```

**Parameters**

| Inputs | Description | Base Type |
|---|---|---|
| Vi | Instrument handle. | ViSession |
| MaxTimeMilliseconds | Specifies the maximum time the end-user allows for this function to complete.  The units are milliseconds. Defined values: Max Time Immediate - The function returns immediately. If the sweep has not completed, the function returns an error. Max Time Infinite - The function waits indefinitely for the frequency sweep to complete. | ViInt32 |
| maxTime | Specifies the maximum time the end-user allows for this function to complete.  The units are implicit in the PrecisionTimeSpan type.  Defined values: PrecisionTimeSpan.Zero - The function returns immediately. If the sweep has not completed, the function returns an error. PrecisionTimeSpan.MaxValue - The function waits indefinitely for the frequency sweep to complete. | PrecisionTimeSpan |

**Defined Values for the MaxTimeMilliseconds Parameter (C/COM)**

| Name | Description | |
|---|---|---|
| | *Language* | *Identifier* |
| Max Time Immediate | The function returns immediately. | |
| | C | IVIDOWNCONVERTER_VAL_MAX_TIME_IMMEDIATE |
| | COM | IviDownconverterTimeOutImmediate |

| Max Time Infinite | The function waits indefinitely for the frequency sweep to complete. | |
| | C | IVIDOWNCONVERTER_VAL_MAX_TIME_INFINITE |
| | COM | IviDownconverterTimeOutInfinite |

**Defined Values for the maxTime Parameter (.NET)**

| *Name* | *Description* | |
| --- | --- | --- |
| | *Language* | *Identifier* |
| Zero | Sets timeout to immediate. The function returns immediately. | |
| | .NET | TimeSpan.Zero |
| MaxValue | Sets timeout to infinite. The function waits indefinitely for the frequency sweep to complete. | |
| | .NET | TimeSpan.MaxValue |

**Return Values (C/COM)**

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

| **Completion Codes** | **Description** |
| --- | --- |
| Max Time Exceeded | Error: Maximum time exceeded before the operation completed. |

**.NET Exceptions**

The *IVI-3.2: Inherent Capabilities Specification* defines general exceptions that may be thrown, and warning events that may be raised, by this method. The table below specifies additional class-defined exceptions for this method.

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

**Compliance Notes**

The specific instrument driver is not required to support any of the defined values for the MaxTimeMilliseconds parameter.

## 8.4 IviDownconverterFrequencySweep Behavior Model

The following state diagram shows the behavior of the IviDownconverterFrequencySweep extension group.
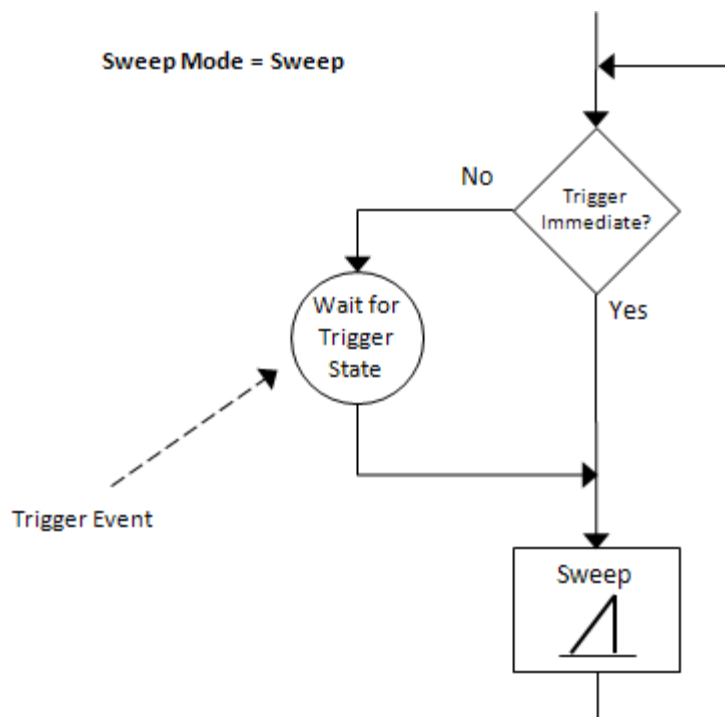


**Figure 8-1.** Frequency Sweep Behavior Model

Setting the Frequency Sweep Trigger Source attribute to "Immediate" will continuously generate sweeps The duration of one sweep from start to stop is defined by the Frequency Sweep Time attribute. Setting the Frequency Sweep Trigger Source attribute to "External" or "Software" will delay the start of the next sweep until the specified trigger event occurs. This is shown below in Figure 8-2 below.
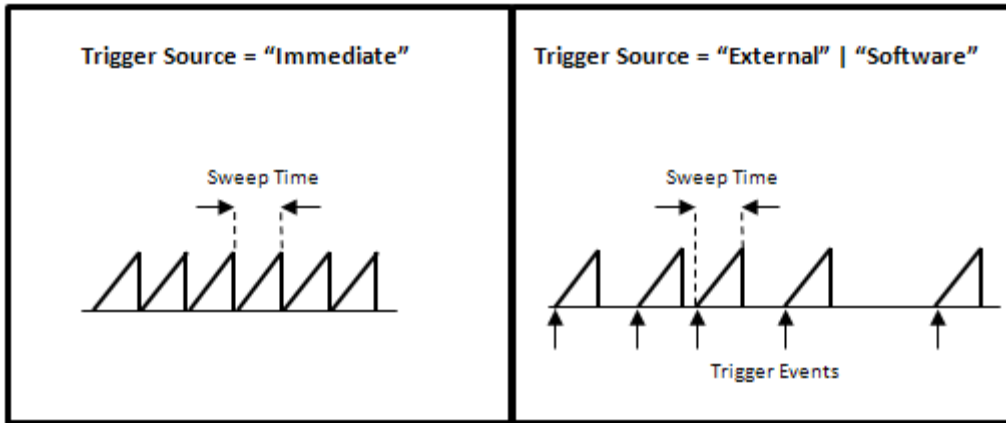
**Figure 8-2.** Frequency Sweep Timing

In addition the following rules apply:

1. When a call to Configure RF Input Frequency of the base capability group is made, the Frequency Sweep Mode attribute is set to None, hence stopping any sweeps that may have been in progress.

2. When the Frequency Sweep Mode attribute is set to Frequency Step, setting the RF Input Frequency attribute of the base capability group will set the Frequency Sweep Mode attribute to None.

3. When the Frequency Sweep Mode attribute is changed from any sweep mode to None, the downconverter will use the previously set value of the RF Input Frequency attribute of the base capability group.

## 8.5 IviDownconverterFrequencySweep Compliance Notes

For a specific driver to comply with the IviDownconverterFrequencySweep extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 9. IviDownconverterFrequencySweepList Extension Group

## 9.1 IviDownconverterFrequencySweepList Overview

The IviDownconverterFrequencySweepList Extension Group supports signal generators that can set the frequency of the RF input signal to values given as a list of values. The user can enable or disable stepping the frequency, specify the name of the frequency list and set its values. The active list can be selected using the list name. Setting single step and dwell time are also included.

This extension group requires the IviDownconverterFrequencySweep Extension Group. List stepping is enabled by setting the Frequency Sweep Mode to List in the IviDownconverterFrequencySweep Extension Group.

## 9.2 IviDownconverterFrequencySweepList Attributes

The IviDownconverterFrequencySweepList extension group defines the following attributes:

- Frequency Sweep List Dwell

- Frequency Sweep List Selected Name

- Frequency Sweep List Single Step Enabled

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions*.

### 9.2.1 Frequency Sweep List Dwell

| Data Type | Access | Applies To | Coercion | High Level Functions |
|---|---|---|---|---|
| ViReal64 (C/COM) PrecisionTimeSpan (.NET) | R/W | N/A | None | Configure Frequency Sweep List Dwell |

**.NET Property Name**

RFInput.FrequencySweep.List.Dwell

**COM Property Name**

RFInput.FrequencySweep.List.Dwell

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_LIST_DWELL

**Description**

Specifies the duration of one step. For C and COM, the units are seconds. For .NET, the units are implicit in the PrecisionTimeSpan type. This attribute is ignored if the Frequency Sweep List Single Step Enabled attribute is set to True.

**.NET Exceptions**

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

## 9.2.2 Frequency Sweep List Selected Name

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViString  | R/W    | N/A        | None     | N/A                  |

**.NET Property Name**

RFInput.FrequencySweep.List.SelectedName

**COM Property Name**

RFInput.FrequencySweep.List.SelectedName

**C Constant Name**

IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_LIST_SELECTED_NAME

**Description**

Specifies the name of the selected list to become active.  The name shall be one of the lists created .

**.NET Exceptions**

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

### 9.2.3 Frequency Sweep List Single Step Enabled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | N/A | None | Configure Frequency Sweep List Dwell |

**.NET Property Name**

> RFInput.FrequencySweep.List.SingleStepEnabled

**COM Property Name**

> RFInput.FrequencySweep.List.SingleStepEnabled

**C Constant Name**

> IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_LIST_SINGLE_STEP_ENABLED

**Description**

> Enables or disables single step mode.

**.NET Exceptions**

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

## 9.3 IviDownconverterFrequencySweepList Functions

The IviDownconverterFrequencySweepList extension group defines the following function:

- Clear All Frequency Sweep Lists

- Configure Frequency Sweep List Dwell

- Create Frequency Sweep List

- Reset Frequency Sweep List

This section describes the behavior and requirements of this function.

### 9.3.1 Clear All Frequency Sweep Lists

**Description**

Deletes all lists from the pool of defined lists.

**.NET Method Prototype**

```
void RFInput.FrequencySweep.List.ClearAll ();
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.List.ClearAll ();
```

**C Prototype**

```
ViStatus IviDownconverter_ClearAllFrequencySweepLists (ViSession Vi);
```

**Parameters**

| Inputs | Description | Base Type |
|---|---|---|
| Vi | Instrument handle. | ViSession |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

### 9.3.2 Configure Frequency Sweep List Dwell

**Description**

Configures the attributes that control frequency list stepping.

**.NET Method Prototype**

```
void RFInput.FrequencySweep.List.ConfigureDwell (
                               Boolean singleStepEnabled,
                               PrecisionTimeSpan dwell);
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.List.ConfigureDwell (
                               [in] VARIANT_BOOL SingleStepEnabled,
                               [in] double Dwell);
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureFrequencySweepListDwell (ViSession Vi,
                               ViBoolean SingleStepEnabled,
                               ViReal64 Dwell);
```

**Parameters**

| Inputs | Description | Base Type |
|---|---|---|
| Vi | Instrument handle. | ViSession |
| SingleStepEnabled | Specifies whether the trigger initiates the next step. The driver uses this value to set the Frequency Sweep List Single Step Enabled attribute. See the attribute description for more details. | ViBoolean |
| Dwell | Specifies the duration of one frequency step. The driver uses this value to set the Frequency Sweep List Dwell attribute. See the attribute description for more details. | ViReal64 |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

### 9.3.3 Create Frequency Sweep List

**Description**

Creates a named list of frequency values.

**.NET Method Prototype**

```
void RFInput.FrequencySweep.List.CreateList (String name,
                                             Double[] frequencyList);
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.List.CreateList (
                                    [in] BSTR Name,
                                    [in] SAFEARRAY(double)* FrequencyList);
```

**C Prototype**

```
ViStatus IviDownconverter_CreateFrequencySweepList (ViSession Vi,
                                    ViConstString Name,
                                    ViInt32 FrequencyListBufferSize,
                                    ViReal64 FrequencyList[]);
```

**Parameters**

| Inputs | Description | Base Type |
|---|---|---|
| Vi | Instrument handle. | ViSession |
| Name | Specifies the name of the frequency list to be created. | ViConstString |
| FrequencyListBuf ferSize | Specifies the number of elements in the frequency list, | ViInt32 |
| FrequencyList | Specifies the array of frequency values to become elements of the list. The array must have at least as many elements as the value in the FrequencyListBufferSize parameter. The units are Hertz. | ViReal64[] |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

### 9.3.4 Reset Frequency Sweep List

**Description**

Resets the current list to the first entry value.

**.NET Method Prototype**

```
void RFInput.FrequencySweep.List.Reset ();
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.List.Reset ();
```

**C Prototype**

```
ViStatus IviDownconverter_ResetFrequencySweepList (ViSession Vi);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

## 9.4 IviDownconverterFrequencySweepList Behavior Model

The IviDownconverterFrequencySweepList extension group follows the same behavior model as the IviDownconverterFrequencyStep capability group described in Section 7.4, *IviDownconverterFrequencyStep Behavior Model*. The only difference between the two behavior models is that instead of the stepped sweep proceeding from a Start frequency to a Stop frequency, the steps in the IviDownconverterFrequencySweepList Extension group are generated from a list of frequencies.

In addition the following rules apply:

1. When a call to Configure RF Input Frequency of the base capability group is made, the Frequency Sweep Mode attribute is set to None, hence stopping any sweeps that may have been in progress.

2. When the Frequency Sweep Mode attribute is set to List Mode, setting the RF Input Frequency attribute of the base capability group will set the Frequency Sweep Mode attribute to None.

3. When the Frequency Sweep Mode attribute is changed from any sweep mode to None, the downconverter will use the previously set value of the RF Input Frequency attribute of the base capability group.

## 9.5 IviDownconverterFrequencySweepList Compliance Notes

For a specific driver to comply with the IviDownconverterFrequencySweepList extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 10. IviDownconverterBandCrossingInformation Extension Group

## 10.1 IviDownconverterBandCrossingInformation Overview

The IviDownconverterBandCrossingInformation Extension Group supports downconverters with the ability to return frequency band information for frequency sweeps.

## 10.2 IviDownconverterBandCrossingInformation Attributes

The IviDownconverterBandCrossingInformation extension group defines the following attributes:

- Number of Band Crossings

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions.*

## 10.2.1 Number of Bands

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViInt32 | RO | N/A | None | N/A |

**.NET Property Name**

     RFInput.FrequencySweep.NumberOfBands

**COM Property Name**

     RFInput.FrequencySweep.NumberOfBands

**C Constant Name**

     IVIDOWNCONVERTER_ATTR_NUM_BANDS

**Description**

Returns the number of frequency bands that will be returned from a call to the Get Band Crossing Info function. The count returned here indicates the number of start-stop frequency pairs.

**.NET Exceptions**

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

## 10.3 IviDownconverterBandCrossingInformation Functions

The IviDownconverterBandCrossingInformation extension group defines the following function:

- Get Band Crossing Info

This section describes the behavior and requirements of this function.

## 10.3.1 Get Band Crossing Info

**Description**

This function returns the band crossing information for sweeps.  Sweep timing is influenced by points in the sweep where frequency bands are crossed.  This function returns pairs of start/stop frequencies over which the sweep timing is constant.  Sweep timing between different pairs of start/stop frequencies is variable.  Thus, users may choose to  use the Wait Until Settled function between sweeps across bands.  The bands are returned in ascending order of frequency.  The Number of Bands attribute can be used to determine how many pairs of frequencies will be returned by this function.

**.NET Method Prototype**

```
struct Band
{
    public Band (Double startFrequency, Double stopFrequency);
    public Double StartFrequency { get; }
    public Double StopFrequency { get; }
}


Band[] RFInput.FrequencySweep.GetBandInformation ();
```

**COM Method Prototype**

```
HRESULT RFInput.FrequencySweep.GetBandCrossingInfo (
                        [in,out] SAFEARRAY(DOUBLE)* StartFrequencies,
                        [in,out] SAFEARRAY(DOUBLE)* StopFrequencies);
```

**C Prototype**

```
ViStatus IviDownconverter_GetBandCrossingInfo (ViSession Vi
                                    ViInt32 BufferSize,
                                    ViReal64 StartFrequencies[],
                                    ViReal64 StopFrequencies[],
                                    ViInt32 *ActualNumFrequencies);
```

**Parameters**

| Inputs | Description | Base Type |
|---|---|---|
| Vi | Instrument handle | ViSession |
| BufferSize | Specifies the number of elements in the StartFrequencies and StopFrequencies arrays. | ViInt32 |

| Outputs | Description | Base Type |
|---|---|---|
| StartFrequencies (C/COM) | A user-allocated (for IVI-C) or driver allocated (for IVI-COM) buffer into which the start frequencies for each band are stored.  Each value is paired with the corresponding element in the StopFrequencies output array.  The frequencies are returned in ascending order.  The units are Hertz. | ViReal64[] |

| | | |
|---|---|---|
| StopFrequencies (C/COM) | A user-allocated (for IVI-C) or driver allocated (for IVI-COM) buffer into which the stop frequencies for each band are stored. Each value is paired with the corresponding element in the StartFrequencies output array. The frequencies are returned in ascending order. The units are Hertz. | ViReal64[] |
| ActualNumFrequencies | Number of frequency values actually returned in each frequency output array. | ViInt32 |
| Return Value (.NET) | An array of start and stop frequency pairs over which the sweep timing is constant. The frequencies are returned in ascending order. The units are Hertz. | Band[] |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

## 10.4 IviDownconverterBandCrossingInformation Behavior Model

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


## 10.5 IviDownconverterBandCrossingInformation Compliance Notes

For a specific driver to comply with the IviDownconverterBandCrossingInformation extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 11. IviDownconverterSoftwareTrigger Extension Group

## 11.1 IviDownconverterSoftwareTrigger Overview

The IviDownconverterSoftwareTrigger Extension Group supports downconverters that can generate output based on a software trigger signal. The user can send a software trigger to start a frequency sweep or to initiate the next frequency step

This extension affects instrument behavior when the Frequency Sweep Trigger Source attribute is set to Software Trigger.

## 11.2 IviDownconverterSoftwareTrigger Functions

The IviDownconverterSoftwareTrigger extension group defines the following function:

- Send Software Trigger

This section describes the behavior and requirements of this function.

## 11.2.1 Send Software Trigger

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

## 11.3 IviDownconverterSoftwareTrigger Behavior Model

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

## 11.4 IviDownconverterSoftwareTrigger Compliance Notes

For a specific driver to comply with the IviDownconverterSoftwareTrigger extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

In addition, the following rules apply:

1.  If a specific driver implements the IviDownconverterSoftwareTrigger Extension Group, it shall also implement the IviDownconverterFrequencySweep extension group.

2.  If a specific driver implements the IviDownconverterSoftwareTrigger Extension, it shall implement the value Software Trigger for the Frequency Sweep Trigger Source attribute.

# 12. IviDownconverterIFFilter Extension Group

## 12.1 IviDownconverterIFFilter Overview

The IviDownconverterIFFilter extension group supports downconverters with the ability to configure a filter on the IF output. It defines a single attribute and an associated configure function.

## 12.2 IviDownconverterIFFilter Attributes

The IviDownconverterIFFilter extension group defines the following attributes:

- IF Output Filter Bandwidth

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions.*

## 12.2.1 IF Output Filter Bandwidth

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | IFOutput | Up | Configure IF Output Filter Bandwidth (IVI-C Only) |

**.NET Property Name**

IFOutput.FilterBandwidth

**COM Property Name**

IFOutput.FilterBandwidth

**C Constant Name**

IVIDOWNCONVERTER_ATTR_IF_OUTPUT_FILTER_BANDWIDTH

**Description**

Specifies the maximum effective IF signal bandwidth that the downconverter's active IF Output can present to the digitizer. The units are Hertz.

This value is a measure of the spectral width between two points for which the amplitude profile is 3 dB below a peak close to mid band.

**.NET Exceptions**

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

## 12.3 IviDownconverterIFFilter Functions

The IviDownconverterIFFilter extension group defines the following function:

- Configure IF Output Filter Bandwidth

This section describes the behavior and requirements of this function.

## 12.3.1 Configure IF Output Filter Bandwidth (IVI-C Only)

**Description**

Configures the IF output filter bandwidth.

**.NET Method Prototype**

```
N/A
(use the IFOutput.FilterBandwidth property)
```

**COM Method Prototype**

```
N/A
(use the IFOutput.FilterBandwidth property)
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureIFOutputFilterBandwidth (ViSession Vi,
                                         ViReal64 Bandwidth);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Bandwidth | Specifies the maximum effective IF signal bandwidth that the downconverter's active IF Output can present to the digitizer. The driver uses this value to set the IF Output Filter Bandwidth 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.

## 12.4 IviDownconverterIFFilter Behavior Model

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

## 12.5 IviDownconverterIFFilter Compliance Notes

For a specific driver to comply with the IviDownconverterIFFilter extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 13. IviDownconverterPreselector Extension Group

## 13.1 IviDownconverterPreselector Overview

The IviDownconverterPreselector extension group supports downconverters with the ability to bypass the preselection filter. It defines a single attribute and an associated configure function.

## 13.2 IviDownconverterPreselector Attributes

The IviDownconverterPreselector extension group defines the following attributes:

- Preselector Enabled

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions*.

## 13.2.1 Preselector Enabled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | RFInput | None | Configure Preselector Enabled (IVI-C Only) |

**.NET Property Prototype**

RFInput.PreselectorEnabled

**COM Property Name**

RFInput.PreselectorEnabled

**C Constant Name**

IVIDOWNCONVERTER_ATTR_PRESELECTOR_ENABLED

**Description**

If True, enables bypassing the downconverter's pre-selection filter for the active RF Input.  If False, disables bypassing the downconverter's pre-selection filter for the active RF Input.

**.NET Exceptions**

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

## 13.3 IviDownconverterPreselector Functions

The IviDownconverterPreselector extension group defines the following function:

- Configure Preselector Enabled

This section describes the behavior and requirements of this function.

### 13.3.1 Configure Preselector Enabled (IVI-C Only)

**Description**

Configures whether or not to bypass the preselection filter for the active RF Input.

**.NET Method Prototype**

```
N/A
(use the RFInput.PreselectorEnabled property)
```

**COM Method Prototype**

```
N/A
(use the RFInput.PreselectorEnabled property)
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigurePreselectorEnabled (ViSession Vi,
                                                       ViBoolean Enabled);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Enabled | Specifies whether or not to bypass the preselection filter. The driver uses this value to set the Preselector Enabled attribute. See the attribute description for more details. | ViBoolean |

**Return Values**

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

## 13.4 IviDownconverterPreselector Behavior Model

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

## 13.5 IviDownconverterPreselector Compliance Notes

For a specific driver to comply with the IviDownconverterPreselector extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 14. IviDownconverterVideoDetectorBandwidth Extension Group

## 14.1 IviDownconverterVideoDetectorBandwidth Overview

The IviDownconverterVideoDetectorBandwidth extension group supports downconverters with the ability to apply video bandwidth detection at the IF output. It defines a single attribute and an associated configure function.

## 14.2 IviDownconverterVideoDetectorBandwidth Attributes

The IviDownconverterVideoDetectorBandwidth extension group defines the following attributes:

- IF Output Video Detector Bandwidth

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 13, *IviDownconverter Attribute ID Definitions.*

## 14.2.1 IF Output Video Detector Bandwidth

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | IFOutput | Up | Configure IF Output Video Detector Bandwidth (IVI-C Only) |

**.NET Property Name**

       IFOutput.VideoDetectorBandwidth

**COM Property Name**

       IFOutput.VideoDetectorBandwidth

**C Constant Name**

       IVIDOWNCONVERTER_ATTR_IF_OUTPUT_VIDEO_DETECTOR_BANDWIDTH

**Description**

       Specifies the 3 dB bandwidth of the active IF output video detection filter.  The units are Hertz.

**.NET Exceptions**

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

## 14.3 IviDownconverterVideoDetectorBandwidth Functions

The IviDownconverterVideoDetectorBandwidth extension group defines the following function:

- Video Detector Bandwidth

This section describes the behavior and requirements of this function.

## 14.3.1 Configure IF Output Video Detector Bandwidth (IVI-C Only)

**Description**

Configures the bandwidth of the active IF output video detector.

**.NET Method Prototype**

```
N/A
```
(use the `IFOutput.VideoDetectorBandwidth` property)

**COM Method Prototype**

```
N/A
```
(use the `IFOutput.VideoDetectorBandwidth` property)

**C Prototype**

```
ViStatus IviDownconverter_ConfigureIFOutputVideoDetectorBandwidth (
                                    ViSession Vi,
                                    ViReal64 Bandwidth);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Bandwidth | Specifies the bandwidth of the IF output video detection filter.  The driver uses this value to set the IF Output Video Detector Bandwidth 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.

## 14.4 IviDownconverterVideoDetectorBandwidth Behavior Model

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

## 14.5 IviDownconverterVideoDetectorBandwidth Compliance Notes

For a specific driver to comply with the IviDownconverterVideoDetectorBandwidth extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 15. IviDownconverterCalibration Extension Group

## 15.1 IviDownconverterCalibration Overview

The IviDownconverterCalibration extension capability group supports downconverters that can perform self-calibration. The IviDownconverterCalibration capability group defines functions to perform the calibration.

## 15.2 IviDownconverterCalibration Functions

The IviDownconverterCalibration extension group defines the following function:

- Calibrate
- Is Calibration Complete
- IsCalibrated

This section describes the behavior and requirements of this function.

### 15.2.1 Calibrate

**Description**

This function performs calibration on the entire device. This call can be blocking or can be non-blocking, depending on the instrument implementation. If it is non-blocking, the user may use the Is Calibration Complete function to determine when the calibration is complete.

For IVI.NET, this method throws an exception if the instrument does not support programmatic calibration operations.

**.NET Method Prototype**

```
void Calibration.Calibrate();
```

**COM Method Prototype**

```
HRESULT RFInput.Calibrate();
```

**C Prototype**

```
ViStatus IviDownconverter_Calibrate (ViSession Vi);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

### 15.2.2 Is Calibration Complete

**Description**

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

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

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

**.NET Method Prototype**

```
CalibrationStatus Calibration.GetCalibrationStatus();
```

**COM Method Prototype**

```
HRESULT RFInput.IsCalibrationComplete(IviDownconverterCalibrationStatusEnum*
Status);
```

**C Prototype**

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

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |

| Outputs | Description | Base Type |
|---------|-------------|-----------|
| Status (C/COM) | Returns the status of the calibration operation. | ViInt32 |
| Return Value (.NET) | Returns the status of the calibration operation. | ViInt32 |

**Defined Values for the Status Parameter**

| Name | Description | |
|------|-------------|--|
| | *Language* | *Identifier* |
| Calibration Complete | The downconverter has completed the calibration. | |
| | .NET | Complete |

| | C | IVIDOWNCONVERTER_VAL_CALIBRATION_COMPLETE |
|---|---|---|
| | COM | IviDownconverterCalibrationStatusComplete |
| Calibration In Progress | The downconverter is still performing the calibration. | |
| | .NET | InProgress |
| | C | IVIDOWNCONVERTER_VAL_CALIBRATION_IN_PROGRESS |
| | COM | IviDownconverterCalibrationStatusInProgress |
| Calibration Status Unknown | The downconverter cannot determine the status of the calibration. | |
| | .NET | Unknown |
| | C | IVIDOWNCONVERTER_VAL_CALIBRATION_STATUS_UNKNOWN |
| | COM | IviDownconverterCalibrationStatusUnknown |
| Calibration Failed | The downconverter calibration failed. | |
| | .NET | Failed |
| | C | IVIDOWNCONVERTER_VAL_CALIBRATION_FAILED |
| | COM | IviDownconverterCalibrationStatusFailed |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

## 15.2.3 Is Calibrated

**Description**

This function queries the instrument to determine the whether the instrument is currently in a valid self-calibrated state or whether it needs to be calibrated.  This function returns the Calibrated value in the Status parameter when the device does not need further self-calibration.

If it does need self-calibration, the driver returns the Uncalibrated value. If the driver cannot query the instrument to determine its state, the driver returns the Calibrated Status Unknown value.

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

**.NET Method Prototype**

```
CalibratedStatus Calibration.GetCalibratedStatus();
```

**COM Method Prototype**

```
HRESULT RFInput.IsCalibrated(IviDownconverterCalibratedStatusEnum* Status);
```

**C Prototype**

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

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |

| Outputs | Description | Base Type |
|---------|-------------|-----------|
| Status (C/COM) | Returns the status of the calibration operation. | ViInt32 |
| Return Value (.NET) | Returns the status of the calibration operation. | ViInt32 |

**Defined Values for the Status Parameter**

| Name | Description | | |
|------|-------------|--|--|
| | | Language | Identifier |
| Calibrated | The downconverter is calibrated. | | |
| | | .NET | Calibrated |
| | | C | IVIDOWNCONVERTER_VAL_CALIBRATED |
| | | COM | IviDownconverterCalibratedStatusCalibrated |

| Uncalibrated | The downconverter requires further calibration. | |
|---|---|---|
| | .NET | Uncalibrated |
| | C | IVIDOWNCONVERTER_VAL_UNCALIBRATED |
| | COM | IviDownconverterCalibratedStatusUncalibrated |
| Calibrated Status Unknown | The downconverter cannot determine the status of the calibration. | |
| | .NET | Unknown |
| | C | IVIDOWNCONVERTER_VAL_CALIBRATED_STATUS_UNKNOWN |
| | COM | IviDownconverterCalibratedStatusUnknown |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

## 15.3 IviDownconverterCalibration Behavior Model

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

## 15.4 IviDownconverterCalibration Compliance Notes

For a specific driver to comply with the IviDownconverterCalibration extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 16. IviDownconverterReferenceOscillator Extension Group

## 16.1 IviDownconverterReferenceOscillator Overview

The IviDownconverterReferenceOscillator extension group supports downconverters with a configurable frequency reference.

## 16.2 IviDownconverterReferenceOscillator Attributes

The IviDownconverterReferenceOscillator extension group defines the following attributes:

- Reference Oscillator External Frequency

- Reference Oscillator Source

- Reference Oscillator Output Enabled

This section describes the behavior and requirements of each attribute. The actual value for each attribute ID is defined in Section 17, *IviDownconverter Attribute ID Definitions.*

## 16.2.1 Reference Oscillator External Frequency

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViReal64 | R/W | N/A | None | Configure Reference Oscillator |

**.NET Property Name**

ReferenceOscillator.ExternalFrequency

**COM Property Name**

ReferenceOscillator.ExternalFrequency

**C Constant Name**

IVIDOWNCONVERTER_ATTR_REFERENCE_OSCILLATOR_EXTERNAL_FREQUENCY

**Description**

Specifies the frequency of the external signal that is used as reference for internal RF frequency generation. This value is used only if Reference Oscillator Source is set to External. The units are Hertz.

**.NET Exceptions**

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

## 16.2.2 Reference Oscillator Source

| Data Type | Access | Applies To | Coercion | High Level Functions |
|---|---|---|---|---|
| ViInt32 (C/COM) ViString (.NET) | R/W | N/A | None | Configure Reference Oscillator |

**.NET Property Name**

      ReferenceOscillator.Source

**.NET Enumeration Name**

      ReferenceOscillatorSource

**COM Property Name**

      ReferenceOscillator.Source

**COM Enumeration Name**

      IviDownconverterReferenceOscillatorSourceEnum

**C Constant Name**

      IVIDOWNCONVERTER_ATTR_REFERENCE_OSCILLATOR_SOURCE

**Description**

      Specifies the reference oscillator source used to generate the precise RF output frequency.

**Defined Values**

In IVI.NET the reference oscillator source is a string. If an IVI driver supports a reference oscillator source and the reference oscillator source is listed in IVI-3.3 *Cross Class Capabilities Specification,* Section 3, then the IVI driver shall accept the standard string for that reference oscillator source. This attribute is case insensitive, but case preserving. That is, the setting is case insensitive but when reading it back the programmed case is returned. IVI specific drivers may define new reference oscillator source strings for reference oscillator sources that are not defined by IVI-3.3 *Cross Class Capabilities Specification* if needed.

| Name | Description | |
|---|---|---|
| | *Language* | *Identifier* |
| Internal | The downconverter uses the internal reference oscillator. | |
| | C | IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_INTERNAL |
| | COM | IviDownconverterReferenceOscillatorSourceInternal |
| External | The downconverter uses an external reference oscillator. | |
| | C | IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_EXTERNAL |
| | COM | IviDownconverterReferenceOscillatorSourceExternal |

**Compliance Notes**

1. If an IVI-C class driver defines additional values for this attribute, the actual values shall be greater than or equal to IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_CLASS_EXT_BASE and less than IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SPECIFIC_EXT_BASE.

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

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

See Section Attribute Value Definitions, for the definitions of Reference Oscillator Specific Ext Base, IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SPECIFIC_EXT_BASE and IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_CLASS_EXT_BASE.

**.NET Exceptions**

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

### 16.2.3 Reference Oscillator Output Enabled

| Data Type | Access | Applies To | Coercion | High Level Functions |
|-----------|--------|------------|----------|----------------------|
| ViBoolean | R/W | N/A | None | Configure Reference Oscillator Output Enabled (IVI-C Only) |

**.NET Property Name**

> ReferenceOscillator.OutputEnabled

**COM Property Name**

> ReferenceOscillator.OutputEnabled

**C Constant Name**

> IVIDOWNCONVERTER_ATTR_REFERENCE_OSCILLATOR_OUTPUT_ENABLED

**Description**

> If True, the Reference output is enabled.  If False, the Reference output is disabled.

**.NET Exceptions**

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

## 16.3 IviDownconverterReferenceOscillator Functions

The IviDownconverterReferenceOscillator extension group defines the following function:

- Configure Reference Oscillator
- Configure Reference Oscillator Output Enabled

This section describes the behavior and requirements of this function.

### 16.3.1 Configure Reference Oscillator

**Description**

Configures the downconverter's reference oscillator.

**.NET Method Prototype**

```
void ReferenceOscillator.Configure (String source,
                                    Double frequency);
```

**COM Method Prototype**

```
HRESULT ReferenceOscillator.Configure ([in]
IviDOwnconverterReferenceOscillatorSourceEnum Source,
                                       [in] double Frequency);
```

**C Prototype**

```
ViStatus IviDownconverter_ConfigureReferenceOscillator (ViSession Vi,
                                            ViInt32 Source,
                                            ViReal64 Frequency);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Source | Specifies the source of the reference frequency signal. The driver uses this value to set the Reference Oscillator Source attribute. See the attribute description for more details. | ViConstString (C/COM) ViString (.NET) |
| Frequency | Specifies the frequency of the external reference oscillator. This parameter is only used if the Source is set to External. The driver uses this value to set the Reference Oscillator External Frequency attribute. See the attribute description for more details. | ViReal64 |

**Defined Values for the Source Parameter**

| *Name* | *Description* | |
|--------|---------------|--|
| | *Language* | *Identifier* |
| Internal | The downconverter uses the internal reference oscillator. | |
| | C | IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_INTERNAL |
| | COM | IviDownconverterReferenceOscillatorSourceInternal |
| External | The downconverter uses an external reference oscillator. | |
| | C | IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_EXTERNAL |
| | COM | IviDownconverterReferenceOscillatorSourceExternal |

**Return Values (C/COM)**

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

**.NET Exceptions**

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

## 16.3.2 Configure Reference Oscillator Output Enabled (IVI-C Only)

**Description**

Configures the downconverter's reference oscillator output.  Many downconverters support the ability to output their frequency reference.  This function allows the user to enable or disable that output.

**.NET Method Prototype**

```
N/A
```
(use the `ReferenceOscillator.OutputEnabled` property)

**COM Method Prototype**

```
N/A
```
(use the `ReferenceOscillator.OutputEnabled` property)

**C Prototype**

```
ViStatus IviDownconverter_ConfigureReferenceOscillatorOutputEnabled (ViSession
Vi,
                                    ViBoolean Enabled);
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| Vi | Instrument handle. | ViSession |
| Enabled | Enables the frequency reference output. | ViBoolean |

**Return Values**

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

## 16.4 IviDownconverterReferenceOscillator Behavior Model

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

## 16.5 IviDownconverterReferenceOscillator Compliance Notes

For a specific driver to comply with the IviDownconverterReferenceOscillator extension, it shall be compliant with the IviDownconverterBase capability group and it shall implement all of the attributes and functions listed in this section.

# 17. IviDownconverter Attribute ID Definitions

<p style="text-align:center">**Table 17-1.** IviDownconverter Attributes ID Values</p>

| Attribute Name | ID Definition |
|---|---|
| Active IF Output | IVI_CLASS_ATTR_BASE + 0 |
| Active RF Input | IVI_CLASS_ATTR_BASE + 1 |
| External LO Enabled | IVI_CLASS_ATTR_BASE + 2 |
| External LO Frequency | IVI_CLASS_ATTR_BASE + 3 |
| IF Output Count | IVI_CLASS_ATTR_BASE + 4 |
| IF Output Enabled | IVI_CLASS_ATTR_BASE + 5 |
| IF Output Frequency | IVI_CLASS_ATTR_BASE + 6 |
| IF Output Gain | IVI_CLASS_ATTR_BASE + 7 |
| Is Settled | IVI_CLASS_ATTR_BASE + 8 |
| RF Input Attenuation | IVI_CLASS_ATTR_BASE + 9 |
| RF Input Coupling | IVI_CLASS_ATTR_BASE + 10 |
| RF Input Count | IVI_CLASS_ATTR_BASE + 11 |
| RF Input Frequency | IVI_CLASS_ATTR_BASE + 12 |
| Bypass | IVI_CLASS_ATTR_BASE + 100 |
| External Mixer Enabled | IVI_CLASS_ATTR_BASE + 110 |
| External Mixer Bias Level | IVI_CLASS_ATTR_BASE + 111 |
| External Mixer Bias Limit | IVI_CLASS_ATTR_BASE + 112 |
| External Mixer Bias Enabled | IVI_CLASS_ATTR_BASE + 113 |
| External Mixer Harmonic | IVI_CLASS_ATTR_BASE + 114 |
| External Mixer Number Of Ports | IVI_CLASS_ATTR_BASE + 115 |
| Frequency Step Dwell | IVI_CLASS_ATTR_BASE + 200 |
| Frequency Step Scaling | IVI_CLASS_ATTR_BASE + 201 |
| Frequency Step Single Step Enabled | IVI_CLASS_ATTR_BASE + 202 |
| Frequency Step Size | IVI_CLASS_ATTR_BASE + 203 |
| Frequency Step Start | IVI_CLASS_ATTR_BASE + 204 |
| Frequency Step Stop | IVI_CLASS_ATTR_BASE + 205 |
| Frequency Sweep Mode | IVI_CLASS_ATTR_BASE + 210 |
| Frequency Sweep Start | IVI_CLASS_ATTR_BASE + 211 |
| Frequency Sweep Stop | IVI_CLASS_ATTR_BASE + 212 |
| Frequency Sweep Time | IVI_CLASS_ATTR_BASE + 213 |
| Frequency Sweep Trigger Source | IVI_CLASS_ATTR_BASE + 214 |
| Is Sweeping | IVI_CLASS_ATTR_BASE + 215 |
| Frequency Sweep List Dwell | IVI_CLASS_ATTR_BASE + 220 |
| Frequency Sweep List Selected Name | IVI_CLASS_ATTR_BASE + 221 |

Table **17-1**. IviDownconverter Attributes ID Values

| Attribute Name | ID Definition |
|---|---|
| Frequency Sweep List Single Step Enabled | IVI_CLASS_ATTR_BASE + 222 |
| Number of Bands | IVI_CLASS_ATTR_BASE + 300 |
| IF Output Filter Bandwidth | IVI_CLASS_ATTR_BASE + 310 |
| Preselector Enabled | IVI_CLASS_ATTR_BASE + 320 |
| IF Output Video Detector Bandwidth | IVI_CLASS_ATTR_BASE + 330 |
| RF Input Corrections Enabled | IVI_CLASS_ATTR_BASE + 340 |
| Reference Oscillator External Frequency | IVI_CLASS_ATTR_BASE + 341 |
| Reference Oscillator Source | IVI_CLASS_ATTR_BASE + 342 |
| Reference Oscillator Output Enabled | IVI_CLASS_ATTR_BASE + 343 |

# 18. IviDownconverter Attribute Value Definitions

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

**Frequency Step Scaling**

| Value Name | Language | Identifier | Actual Value |
|---|---|---|---|
| Linear | .NET | `FrequencyStepScaling.Linear` | 0 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_LINEAR` | 0 |
| | COM | `IviDownconverterFrequencyStepScalingLinear` | 0 |
| Logarithmic | .NET | `FrequencyStepScaling.Logarithmic` | 1 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_LOGARITHMIC` | 1 |
| | COM | `IviDownconverterFrequencyStepScalingLogarithmic` | 1 |
| Frequency Step Scaling Class Ext Base | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_CLASS_EXT_BASE` | 100 |
| Frequency Step Scaling Specific Ext Base | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_SPECIFIC_EXT_BASE` | 1000 |
| | COM | `N/A` | 1000 |

**Frequency Sweep Mode**

| Value Name | Language | Identifier | Actual Value |
|---|---|---|---|
| None | .NET | `FrequencySweepMode.None` | 0 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_NONE` | 0 |
| | COM | `IviDownconverterFrequencySweepModeNone` | 0 |
| Sweep | .NET | `FrequencySweepMode.Sweep` | 1 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_SWEEP` | 1 |
| | COM | `IviDownconverterFrequencySweepModeSweep` | 1 |
| Step | .NET | `FrequencySweepMode.Step` | 2 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_STEP` | 2 |
| | COM | `IviDownconverterFrequencySweepModeStep` | 2 |
| List | .NET | `FrequencySweepMode.List` | 3 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_LIST` | 3 |
| | COM | `IviDownconverterFrequencySweepModeList` | 3 |
| Frequency Sweep Mode Class Ext Base | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_CLASS_EXT_BASE` | 100 |

| Value Name | Language | Identifier | Actual Value |
|---|---|---|---|
| Frequency Sweep Mode | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_SPECIFIC_EXT_BASE` | 1000 |
| Specific Ext Base | COM | `N/A` | 1000 |

**Reference Oscillator Source**

| Value Name | Language | Identifier | Actual Value |
|---|---|---|---|
| Internal | C | `IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_INTERNAL` | 0 |
| | COM | `IviDownconverterReferenceOscillatorSourceInternal` | 0 |
| External | C | `IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_EXTERNAL` | 1 |
| | COM | `IviDownconverterReferenceOscillatorSourceExternal` | 1 |
| Reference Oscillator Source Class Ext Base | C | `IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_CLASS_EXT_BASE` | 100 |
| Reference Oscillator Source Specific Ext Base | C | `IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_SPECIFIC_EXT_BASE` | 1000 |
| | COM | `N/A` | 1000 |

**RF Input Coupling**

| Value Name | Language | Identifier | Actual Value |
|---|---|---|---|
| AC | .NET | `InputCoupling.AC` | 0 |
| | C | `IVIDOWNCONVERTER_VAL_INPUT_COUPLING_AC` | 0 |
| | COM | `IviDownconverterInputCouplingAC` | 0 |
| DC | .NET | `InputCoupling.DC` | 1 |
| | C | `IVIDOWNCONVERTER_VAL_INPUT_COUPLING_DC` | 1 |
| | COM | `IviDownconverterInputCouplingDC` | 1 |
| RF Input Coupling Class Ext Base | C | `IVIDOWNCONVERTER_VAL_INPUT_COUPLING_CLASS_EXT_BASE` | 100 |
| RF Input Coupling Specific xt Base | C | `IVIDOWNCONVERTER_VAL_INPUT_COUPLING_SPECIFIC_EXT_BASE` | 1000 |
| | COM | `N/A` | 1000 |

# 19. IviDownconverter Function Parameter Value Definitions

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

**Configure Frequency Step Start Stop**

**Parameter:** `Scaling`

| *Value Name* | *Language* | *Identifier* | *Actual Value* |
|---|---|---|---|
| Linear | .NET | `FrequencyStepScaling.Linear` | 0 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_LINEAR` | 0 |
| | COM | `IviDownconverterFrequencyStepScalingLinear` | 0 |
| Logarithmic | .NET | `FrequencyStepScaling.Logarithmic` | 1 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_STEP_SCALING_LOGARITHMIC` | 1 |
| | COM | `IviDownconverterFrequencyStepScalingLogarithmic` | 1 |

**Configure Frequency Sweep**

**Parameter:** `Mode`

| *Value Name* | *Language* | *Identifier* | *Actual Value* |
|---|---|---|---|
| None | .NET | `FrequencySweepMode.None` | 0 |
| | C | `IVIDOWNCONVERTERVAL_VAL_FREQUENCY_SWEEP_MODE_NONE` | 0 |
| | COM | `IviDownconverterFrequencySweepModeNone` | 0 |
| Sweep | .NET | `FrequencySweepMode.Sweep` | 1 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_SWEEP` | 1 |
| | COM | `IviDownconverterFrequencySweepModeSweep` | 1 |
| Step | .NET | `FrequencySweepMode.Step` | 2 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_STEP` | 2 |
| | COM | `IviDownconverterFrequencySweepModeStep` | 2 |
| List | .NET | `FrequencySweepMode.List` | 3 |
| | C | `IVIDOWNCONVERTER_VAL_FREQUENCY_SWEEP_MODE_LIST` | 3 |
| | COM | `IviDownconverterFrequencySweepModeList` | 3 |

**Configure Reference Oscillator**

**Parameter:** `Source`

| *Value Name* | *Language* | *Identifier* | *Actual Value* |
|---|---|---|---|
| Internal | C | `IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_INTERNAL` | 0 |
| | COM | `IviDownconverterReferenceOscillatorSourceInternal` | 0 |
| External | C | `IVIDOWNCONVERTER_VAL_REFERENCE_OSCILLATOR_SOURCE_EXTERNAL` | 1 |
| | COM | `IviDownconverterReferenceOscillatorSourceExternal` | 1 |

**Is Calibrated**

**Parameter:** `Status`

| *Value Name* | *Language* | *Identifier* | *Actual Value* |
|---|---|---|---|
| Calibrated | .NET | `CalibratedStatus.Calibrated` | 0 |
| | C | `IVIDOWNCONVERTER_VAL_CALIBRATED` | 0 |
| | COM | `IviDownconverterCalibratedStatusCalibrated` | 0 |
| Uncalibrated | .NET | `CalibratedStatus.Uncalibrated` | 1 |
| | C | `IVIDOWNCONVERTER_VAL_UNCALIBRATED` | 1 |
| | COM | `IviDownconverterCalibratedStatusUncalibrated` | 1 |
| Calibrated Status Unknown | .NET | `CalibratedStatus.Unknown` | 2 |
| | C | `IVIDOWNCONVERTER_VAL_CALIBRATED_STATUS_UNKNOWN` | 2 |
| | COM | `IviDownconverterCalibratedStatusUnknown` | 2 |

**Is Calibration Complete**

**Parameter:** `Status`

| *Value Name* | *Language* | *Identifier* | *Actual Value* |
|---|---|---|---|
| Calibration Complete | .NET | `CalibrationStatus.Complete` | 0 |
| | C | `IVIDOWNCONVERTER_VAL_CALIBRATION_COMPLETE` | 0 |
| | COM | `IviDownconverterCalibrationStatusComplete` | 0 |
| Calibration In Progress | .NET | `CalibrationStatus.InProgress` | 1 |

**Parameter:** `Status`

| *Value Name* | *Language* | *Identifier* | *Actual Value* |
|---|---|---|---|
| | C | `IVIDOWNCONVERTER_VAL_CALIBRATION_IN_PROGRESS` | 1 |
| | COM | `IviDownconverterCalibrationStatusInProgress` | 1 |
| Calibration Status Unknown | .NET | `CalibrationStatus.Unknown` | 2 |
| | C | `IVIDOWNCONVERTER_VAL_CALIBRATION_STATUS_UNKNOWN` | 2 |
| | COM | `IviDownconverterCalibrationStatusUnknown` | 2 |
| Calibration Failed | .NET | `CalibrationStatus.Failed` | 3 |
| | C | `IVIDOWNCONVERTER_VAL_CALIBRATION_FAILED` | 3 |
| | COM | `IviDownconverterCalibrationStatusFailed` | 3 |

## Wait Until Frequency Sweep Complete

**Parameter:** `MaxTimeMilliseconds`

| Value Name | Language | Identifier | Actual Value |
|---|---|---|---|
| Max Time Immediate | C | `IVIDOWNCONVERTER_VAL_MAX_TIME_IMMEDIATE` | 0x0 |
| | COM | `IviDownconverterTimeOutImmediate` | 0x0 |
| Max Time Infinite | C | `IVIDOWNCONVERTER_VAL_MAX_TIME_INFINITE` | 0xFFFFFFFFUL |
| | COM | `IviDownconverterTimeOutInfinite` | 0xFFFFFFFFUL |

## Wait Until Settled

**Parameter:** `MaxTimeMilliseconds`

| Value Name | Language | Identifier | Actual Value |
|---|---|---|---|
| Max Time Immediate | C | `IVIDOWNCONVERTER_VAL_MAX_TIME_IMMEDIATE` | 0x0 |
| | COM | `IviDownconverterTimeOutImmediate` | 0x0 |
| Max Time Infinite | C | `IVIDOWNCONVERTER_VAL_MAX_TIME_INFINITE` | 0xFFFFFFFFUL |
| | COM | `IviDownconverterTimeOutInfinite` | 0xFFFFFFFFUL |

# 20. IviDownconverter Error and Completion Code Value Definitions

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

**Table 20-1. IviDownconverter Error and Completion Codes**

| *Error Name* | *Description* | | |
|---|---|---|---|
| | *Language* | *Identifier* | *Value(hex)* |
| Trigger Not Software | The trigger source is not set to software trigger. | | |
| | .NET | `Ivi.Driver.TriggerNotSoftwareException` | IVI Defined Exception (See IVI-3.2) |
| | C | `IVIDOWNCONVERTER`<br>`_ERROR_TRIGGER_NOT_SOFTWARE` | 0xBFFA1001 |
| | COM | `E_ IVIDOWNCONVERTER`<br>`_TRIGGER_NOT_SOFTWARE` | 0x80041001 |
| Frequency List Unknown | The selected frequency list is not defined. | | |
| | .NET | `FrequencyListUnknownException` | |
| | C | `IVIDOWNCONVERTER_ERROR_FREQUENCY_LIST_`<br>`UNKNOWN` | 0xBFFA2001 |
| | COM | `E_IVIDOWNCONVERTER_FREQUENCY_LIST_UNKN`<br>`OWN` | 0x80042001 |
| Max Time Exceeded | Maximum time exceeded before the operation completed. | | |
| | .NET | `Ivi.Driver.MaxTimeExceededException` | IVI Defined Exception (See IVI-3.2) |
| | C | `IVIDOWNCONVERTER_ERROR_MAX_TIME_EXCEED`<br>`ED` | 0xBFFA2002 |
| | COM | `E_IVIDOWNCONVERTER_MAX_TIME_EXCEEDED` | 0x80042002 |

Table 20-2 defines the recommended format of the message string associated with the errors. In C, these strings are returned by the Get Error function. In COM, these strings are the description contained in the ErrorInfo object.

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

**Table 20-2.** IviDownconverter Error Message Strings

| Name | Message String |
|---|---|
| Trigger Not Software | "%s: Trigger source is not set to software trigger." |
| Frequency List Unknown | "%s: The selected frequency list is not defined." |
| Max Time Exceeded | "%s: Maximum time exceeded before the operation completed." |

## 20.1 IVI.NET IviDownconverter Exceptions and Warnings

This section defines the list of IVI.NET exceptions and warnings that are specific to the IviDownconverter class. For general information on IVI.NET exceptions and warnings, refer to *IVI-3.1: Driver Architecture*

*Specification* and section 12, *Common IVI.NET Exceptions and Warnings*, of *IVI-3.2: Inherent Capabilities Specification*.

The IVI.NET exceptions defined in this specification are declared in the Ivi.Downconverter namespace.

- FrequencyListUnknownException

## 20.1.1 FrequencyListUnknownException

**Description**

This exception is used when the driver finds that a specified frequency list is not defined.

**Constructors**

```
Ivi.Downconverter.FrequencyListUnknownException(
                                      String message,
                                      String listName);

Ivi.Downconverter.FrequencyListUnknownException();

Ivi.Downconverter.FrequencyListUnknownException(
                                      String message);

Ivi.Downconverter.FrequencyListUnknownException(
                                      String message,
                                      System.Exception innerException);
```

**Message String**

```
The selected frequency list is not defined.
List name: <listName>
```

**Parameters**

| Inputs | Description | Base Type |
|--------|-------------|-----------|
| listName | The name of the list that is undefined. | String |

**Usage**

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

# 21. IviDownconverter Hierarchies

## 21.1 IviDownconverter .NET Hierarchy

The full IviDownconverter .NET Hierarchy includes the Inherent Capabilities Hierarchy as defined in Section 4.1, *.NET Inherent Capabilities* of *IVI-3.2: Inherent Capabilities Specification*. To avoid redundancy, it is omitted from Table 17-1.

**Table 21-1.** IviDownconverter COM Hierarchy

| COM Interface Hierarchy | Generic Name | Type |
|---|---|---|
| **Calibration** | | |
| Calibrate | Calibrate | M |
| GetCalibratedStatus | Is Calibrated | M |
| GetCalibrationStatus | Is Calibration Complete | M |
| **ExternalLO** | | |
| Enabled | External LO Enabled | P |
| Frequency | External LO Frequency | P |
| **ExternalMixer** | | |
| Enabled | External Mixer Enabled | P |
| Harmonic | External Mixer Harmonic | P |
| NumberOfPorts | External Mixer Number Of Ports | P |
| **Bias** | | |
| Configure | Configure External Mixer Bias | M |
| Enabled | External Mixer Bias Enabled | P |
| Level | External Mixer Bias Level | P |
| Limit | External Mixer Bias Limit | P |
| **IFOutput** | | |
| WaitUntilSettled | Wait Until Settled | M |
| ActiveIFOutput | Active IF Output | P |
| FilterBandwidth | IF Output Filter Bandwidth | P |
| Count | IF Output Count | P |
| Enabled | IF Output Enabled | P |
| Frequency | IF Output Frequency | P |
| Gain | IF Output Gain | P |
| IsSettled | Is Settled | P |
| GetName | Get IF Output Name | M |
| VideoDetectorBandwidth | IF Output Video Detector Bandwidth | P |
| **ReferenceOscillator** | | |
| Configure | Configure Reference Oscillator | M |
| ExternalFrequency | Reference Oscillator External Frequency | P |
| Output Enabled | Reference Oscillator Output Enabled | P |

**Table 21-1.** IviDownconverter COM Hierarchy

| COM Interface Hierarchy | Generic Name | Type |
|---|---|---|
| Source | Reference Oscillator Source | P |
| **RFInput** | | |
| ActiveRFInput | Active RF Input | P |
| Attenuation | RF Input Attenuation | P |
| CorrectionsEnabled | RF Input Corrections Enabled | P |
| SendSoftwareTrigger | Send Software Trigger | M |
| Bypass | Bypass | P |
| Count | RF Input Count | P |
| Coupling | RF Input Coupling | P |
| Frequency | RF Input Frequency | P |
| GetName | Get RF Input Name | M |
| PreselectorEnabled | Preselector Enabled | P |
| **FrequencySweep** | | |
| Configure | Configure Frequency Sweep | M |
| GetBandCrossingInfo | Get Band Crossing Info | M |
| Mode | Frequency Sweep Mode | P |
| NumberOfBands | Number Of Bands | P |
| TriggerSource | Frequency Sweep Trigger Source | P |
| WaitUntilComplete | Wait Until Frequency Sweep Complete | M |
| **Analog** | | |
| ConfigureStartStop | Configure Frequency Sweep Start Stop | M |
| Start | Frequency Sweep Start | P |
| Stop | Frequency Sweep Stop | P |
| Time | Frequency Sweep Time | P |
| **List** | | |
| ClearAll | Clear All Frequency Sweep Lists | M |
| ConfigureDwell | Configure Frequency Sweep List Dwell | M |
| CreateList | Create Frequency Sweep List | M |
| Reset | Reset Frequency Sweep List | M |
| Dwell | Frequency Sweep List Dwell | P |
| SelectedName | Frequency Sweep List Selected Name | P |
| SingleStepEnabled | Frequency Sweep List Single Step Enabled | P |
| **Step** | | |
| ConfigureDwell | Configure Frequency Step Dwell | M |
| ConfigureStartStop | Configure Frequency Step Start Stop | M |
| Reset | Reset Frequency Step | M |
| Dwell | Frequency Step Dwell | P |
| Scaling | Frequency Step Scaling | P |

**Table 21-1.** IviDownconverter COM Hierarchy

| COM Interface Hierarchy | Generic Name | Type |
|---|---|---|
| SingleStepEnabled | Frequency Step Single Step Enabled | P |
| Size | Frequency Step Size | P |
| Start | Frequency Step Start | P |
| Stop | Frequency Step Stop | P |

### 21.1.1 IviDownconverter .NET Interfaces

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

- IIviDownconverterReferenceOscillator
- IIviDownconverterRFInput
- IIviDownconverterIFOutput
- IIviDownconverterExternalLO
- IiviDownconverterExternalMixer
- IIviDownconverterCalibration

The IIviDownconverterRFInput interface contains interface reference properties for accessing additional the following additional IviDownconverter interface(s):

- IIviDownconverterFrequencySweep

The IIviDownconverterExternalMixer interface contains interface reference properties for accessing additional the following additional IviDownconverter interface(s):

- IIviDownconverterExternalMixerBias

The IIviDownconverterFrequencySweep interface contains interface reference properties for accessing additional the following additional IviDownconverter interface(s):

- IIviDownconverterFrequencySweepList
- IIviDownconverterFrequencyStep
- IiviDownconverterFrequencySweepAnalog

### 21.1.2 .NET Interface Reference Properties

Interface reference properties are used to navigate the IviDownconverter .NET hierarchy. This section describes the interface reference properties that the IviDownconverter interfaces define.  All interface reference properties are read-only.

**Table 21-2.** IviDownconverter Interface Reference Propertiess

| Interface | GUID |
|---|---|
| IIviDownconverterCalibration | Calibration |
| IIviDownconverterExternalLO | ExternalLO |
| IIviDownconverterExternalMixer | ExternalMixer |
| IIviDownconverterExternalMixerBias | ExternalMixer.Bias |
| IIviDownconverterFrequencyStep | RFInput.FrequencySweep.Step |
| IIviDownconverterFrequencySweep | RFInput.FrequencySweep |
| IIviDownconverterFrequencySweepAnalog | RFInput.FrequencySweep.Analog |
| IIviDownconverterFrequencySweepList | RFInput.FrequencySweep.List |
| IIviDownconverterIFOutput | IFOutput |
| IIviDownconverterReferenceOscillator | ReferenceOscillator |
| IIviDownconverterRFInput | RFInput |

## *21.2 IviDownconverter COM Hierarchy*

The full IviDownconverter COM Hierarchy includes the Inherent Capabilities Hierarchy as defined in Section 4.2, *COM Inherent Capabilities* of *IVI-3.2: Inherent Capabilities Specification*. To avoid redundancy, it is omitted from Table 17-1.

**Table 21-3.** IviDownconverter COM Hierarchy

| COM Interface Hierarchy | Generic Name | Type |
|---|---|---|
| **ExternalLO** | | |
| Enabled | External LO Enabled | P |
| Frequency | External LO Frequency | P |
| **ExternalMixer** | | |
| Enabled | External Mixer Enabled | P |
| Harmonic | External Mixer Harmonic | P |
| NumberOfPorts | External Mixer Number Of Ports | P |
| **Bias** | | |
| Configure | Configure External Mixer Bias | M |
| Enabled | External Mixer Bias Enabled | P |
| Level | External Mixer Bias Level | P |
| Limit | External Mixer Bias Limit | P |
| **IFOutput** | | |
| WaitUntilSettled | Wait Until Settled | M |
| ActiveIFOutput | Active IF Output | P |
| FilterBandwidth | IF Output Filter Bandwidth | P |
| Count | IF Output Count | P |
| Enabled | IF Output Enabled | P |
| Frequency | IF Output Frequency | P |
| Gain | IF Output Gain | P |
| IsSettled | Is Settled | P |
| Name | IF Output Name | P |
| VideoDetectorBandwidth | IF Output Video Detector Bandwidth | P |
| **ReferenceOscillator** | | |
| Configure | Configure Reference Oscillator | M |
| ExternalFrequency | Reference Oscillator External Frequency | P |
| Output Enabled | Reference Oscillator Output Enabled | P |
| Source | Reference Oscillator Source | P |
| **RFInput** | | |
| ActiveRFInput | Active RF Input | P |
| Attenuation | RF Input Attenuation | P |
| CorrectionsEnabled | RF Input Corrections Enabled | P |
| SendSoftwareTrigger | Send Software Trigger | M |

**Table 21-3.** IviDownconverter COM Hierarchy

| COM Interface Hierarchy | Generic Name | Type |
|---|---|---|
| Bypass | Bypass | P |
| Calibrate | Calibrate | M |
| IsCalibrated | Is Calibrated | M |
| IsCalibrationComplete | Is Calibration Complete | M |
| Count | RF Input Count | P |
| Coupling | RF Input Coupling | P |
| Frequency | RF Input Frequency | P |
| Name | RF Input Name | P |
| PreselectorEnabled | Preselector Enabled | P |
| **FrequencySweep** | | |
| Configure | Configure Frequency Sweep | M |
| GetBandCrossingInfo | Get Band Crossing Info | M |
| Mode | Frequency Sweep Mode | P |
| NumberOfBands | Number Of Bands | P |
| TriggerSource | Frequency Sweep Trigger Source | P |
| WaitUntilComplete | Wait Until Frequency Sweep Complete | M |
| **Analog** | | |
| ConfigureStartStop | Configure Frequency Sweep Start Stop | M |
| Start | Frequency Sweep Start | P |
| Stop | Frequency Sweep Stop | P |
| Time | Frequency Sweep Time | P |
| **List** | | |
| ClearAll | Clear All Frequency Sweep Lists | M |
| ConfigureDwell | Configure Frequency Sweep List Dwell | M |
| CreateList | Create Frequency Sweep List | M |
| Reset | Reset Frequency Sweep List | M |
| Dwell | Frequency Sweep List Dwell | P |
| SelectedName | Frequency Sweep List Selected Name | P |
| SingleStepEnabled | Frequency Sweep List Single Step Enabled | P |
| **Step** | | |
| ConfigureDwell | Configure Frequency Step Dwell | M |
| ConfigureStartStop | Configure Frequency Step Start Stop | M |
| Reset | Reset Frequency Step | M |
| Dwell | Frequency Step Dwell | P |
| Scaling | Frequency Step Scaling | P |
| SingleStepEnabled | Frequency Step Single Step Enabled | P |
| Size | Frequency Step Size | P |
| Start | Frequency Step Start | P |

**Table 21-3.** IviDownconverter COM Hierarchy

| COM Interface Hierarchy | Generic Name | Type |
|---|---|---|
| Stop | Frequency Step Stop | P |

### 21.2.1 IviDownconverter COM Interfaces

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

- IIviDownconverterReferenceOscillator
- IIviDownconverterRFInput
- IIviDownconverterIFOutput
- IIviDownconverterExternalLO
- IIviDownconverterExternalMixer

The IIviDownconverterRFInput interface contains interface reference properties for accessing additional the following additional IviDownconverter interface(s):

- IIviDownconverterFrequencySweep

The IIviDownconverterExternalMixer interface contains interface reference properties for accessing additional the following additional IviDownconverter interface(s):

- IIviDownconverterExternalMixerBias

The IIviDownconverterFrequencySweep interface contains interface reference properties for accessing additional the following additional IviDownconverter interface(s):

- IIviDownconverterFrequencySweepList
- IIviDownconverterFrequencyStep
- IIviDownconverterFrequencySweepAnalog

*Table 21-4. IviDownconverter* lists the interfaces that this specification defines and their GUIDs.

**Table 21-4.** IviDownconverter Interface GUIDs

| Interface | GUID |
|---|---|
| IIviDownconverter | {47ed53aa-a398-11d4-ba58-000064657374} |
| IIviDownconverterExternalLO | {47ed53ab-a398-11d4-ba58-000064657374} |
| IIviDownconverterExternalMixer | {47ed53ac-a398-11d4-ba58-000064657374} |
| IIviDownconverterExternalMixerBias | {47ed53ad-a398-11d4-ba58-000064657374} |
| IIviDownconverterFrequencyStep | {47ed53ae-a398-11d4-ba58-000064657374} |
| IIviDownconverterFrequencySweep | {47ed53af-a398-11d4-ba58-000064657374} |
| IIviDownconverterFrequencySweepAnalog | {47ed53b0-a398-11d4-ba58-000064657374} |
| IIviDownconverterFrequencySweepList | {47ed53b1-a398-11d4-ba58-000064657374} |
| IIviDownconverterIFOutput | {47ed53b2-a398-11d4-ba58-000064657374} |
| IIviDownconverterReferenceOscillator | {47ed53b3-a398-11d4-ba58-000064657374} |
| IIviDownconverterRFInput | {47ed53b4-a398-11d4-ba58-000064657374} |

### 21.2.1 COM Interface Reference Properties

Interface reference properties are used to navigate the IviDownconverter COM hierarchy. This section describes the interface reference properties that the IviDownconverter interfaces define. All interface reference properties are read-only.

**Table 21-5.** IviDownconverter Interface Reference Propertiess

| Interface | Interface Reference Property |
|---|---|
| `IIviDownconverterExternalLO` | `ExternalLO` |
| `IIviDownconverterExternalMixer` | `ExternalMixer` |
| `IIviDownconverterExternalMixerBias` | `ExternalMixer.Bias` |
| `IIviDownconverterFrequencyStep` | `RFInput.FrequencySweep.Step` |
| `IIviDownconverterFrequencySweep` | `RFInput.FrequencySweep` |
| `IIviDownconverterFrequencySweepAnalog` | `RFInput.FrequencySweep.Analog` |
| `IIviDownconverterFrequencySweepList` | `RFInput.FrequencySweep.List` |
| `IIviDownconverterIFOutput` | `IFOutput` |
| `IIviDownconverterReferenceOscillator` | `ReferenceOscillator` |
| `IIviDownconverterRFInput` | `RFInput` |

## 21.2.2 IviDownconverter COM Category

The IviDownconverter class COM Category shall be "IviDownconverter", and the Category ID (CATID) shall be {47ed515e-a398-11d4-ba58-000064657374}.

## 21.3 IviDownconverter C Function Hierarchy

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

**Table 17-3.** IviDownconverter Function Hierarchy

| Name or Class | Function Name |
|---|---|
| *Attribute Accessors…* | |
| GetAttributeViBoolean | IviDownconverter_GetAttributeViBoolean |
| GetAttributeViInt32 | IviDownconverter_GetAttributeViInt32 |
| GetAttributeViReal64 | IviDownconverter_GetAttributeViReal64 |
| GetAttributeViSession | IviDownconverter_GetAttributeViSession |
| GetAttributeViString | IviDownconverter_GetAttributeViString |
| SetAttributeViBoolean | IviDownconverter_SetAttributeViBoolean |
| SetAttributeViInt32 | IviDownconverter_SetAttributeViInt32 |
| SetAttributeViReal64 | IviDownconverter_SetAttributeViReal64 |
| SetAttributeViSession | IviDownconverter_SetAttributeViSession |
| SetAttributeViString | IviDownconverter_SetAttributeViString |
| Close | IviDownconverter_close |
| *External Mixer…* | |
| *Bias…* | |
| Configure External Mixer Bias | IviDownconverter_ConfigureExternalMixerBias |
| *IF Output…* | |
| Configure IF Output Filter Bandwidth | IviDownconverter_ConfigureIFOutputFilterBandwidth |
| Configure IF Output Enabled | IviDownconverter_ConfigureIFOutputEnabled |
| Configure IF Output Gain | IviDownconverter_ConfigureIFOutputGain |
| Configure IF Output Video Detector Bandwidth | IviDownconverter_ConfigureIFOutputVideoDetectorBandwidth |
| Get IF Output Name | IviDownconverter_GetIFOutputName |
| Set Active IF Output | IviDownconverter_SetActiveIFOutput |
| Initialize | IviDownconverter_init |
| Initialize With Options | IviDownconverter_InitWithOptions |
| *Reference Oscillator…* | |
| Configure Reference Oscillator | IviDownconverter_ConfigureReferenceOscillator |
| Configure Reference Oscillator Output Enabled | IviDownconverter_ConfigureReferenceOscillatorOutputEnabled |
| *RF Input…* | |
| Configure Bypass | IviDownconverter_ConfigureBypass |
| Configure Preselector Enabled | IviDownconverter_ConfigurePreselectorEnabled |
| Configure RF Input Attenuation | IviDownconverter_ConfigureRFInputAttenuation |
| Configure RF Input Frequency | IviDownconverter_ConfigureRFInputFrequency |
| Get RF Input Name | IviDownconverter_GetRFInputName |

| Name or Class | Function Name |
|---|---|
| Set Active RF Input | `IviDownconverter_SetActiveRFInput` |
| *Frequency Sweep…* | |
| Configure Frequency Sweep | `IviDownconverter_ConfigureFrequencySweep` |
| Configure Frequency Sweep Time | `IviDownconverter_ConfigureFrequencySweepTime` |
| Get Band Crossing Info | `IviDownconverter_GetBandCrossingInfo` |
| *Analog…* | |
| Configure Frequency Sweep Start Stop | `IviDownconverter_ConfigureFrequencySweepStart Stop` |
| *List…* | |
| Clear All Frequency Sweep Lists | `IviDownconverter_ClearAllFrequencySweepLists` |
| Configure Frequency Sweep List Dwell | `IviDownconverter_ConfigureFrequencySweepListD well` |
| Create Frequency Sweep List | `IviDownconverter_CreateFrequencySweepList` |
| Reset Frequency Sweep List | `IviDownconverter_ResetFrequencySweepList` |
| *Step…* | |
| Configure Frequency Step Dwell | `IviDownconverter_ConfigureFrequencyStepDwell` |
| Configure Frequency Step Start Stop | `IviDownconverter_ConfigureFrequencyStepStartS top` |
| Reset Frequency Step | `IviDownconverter_ResetFrequencyStep` |
| *Action…* | |
| Calibrate | `IviDownconverter_Calibrate` |
| Is Calibrated | `IviDownconverter_IsCalibrated` |
| Is Calibration Complete | `IviDownconverter_IsCalibrationComplete` |
| Send Software Trigger | `IviDownconverter_SendSoftwareTrigger` |
| Wait Until Settled | `IviDownconverter_WaitUntilSettled` |
| Wait Until Frequency Sweep Complete | `IviDownconverter_WaitUntilFrequencySweepCompl ete` |
| *Utility…* | |
| Clear Error | `IviDownconverter_ClearError` |
| Clear Interchange Warnings | `IviDownconverter_ClearInterchangeWarnings` |
| Disable | `IviDownconverter_Disable` |
| Error Message | `IviDownconverter_error_message` |
| Error Query | `IviDownconverter_error_query` |
| Get Error | `IviDownconverter_GetError` |
| Get Next Coercion Record | `IviDownconverter_GetNextCoercionRecord` |
| Get Next Interchange Warning | `IviDownconverter_GetNextInterchangeWarning` |
| Get Specific Driver C Handle | `IviDownconverter_GetSpecificDriverCHandle` |
| Invalidate All Attributes | `IviDownconverter_InvalidateAllAttributes` |
| Lock Session | `IviDownconverter_LockSession` |
| Reset | `IviDownconverter_reset` |
| Reset Interchange Check | `IviDownconverter_ResetInterchangeCheck` |
| ResetWithDefaults | `IviDownconverter_ResetWithDefaults` |

| Name or Class | Function Name |
|---|---|
| Revision Query | IviDownconverter_revision_query |
| Self Test | IviDownconverter_self_test |
| Unlock Session | IviDownconverter_UnlockSession |

## 21.4 IviDownconverter C Attribute Hierarchy

The IviDownconverter class attribute hierarchy is shown in the following table.

**Table 21-6** IviDownconverter C Attributes Hierarchy

| Category or Generic Attribute Name | C Defined Constant |
|---|---|
| *External LO* | |
| External LO Enabled | IVIDOWNCONVERTER_ATTR_EXTERNAL_LO_ENABLED |
| External LO Frequency | IVIDOWNCONVERTER_ATTR_EXTERNAL_LO_FREQUENCY |
| *External Mixer* | |
| *Bias* | |
| External Mixer Bias Enabled | IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_BIAS_EN ABLED |
| External Mixer Bias Level | IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_BIAS_LE VEL |
| External Mixer Bias Limit | IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_BIAS_LI MIT |
| External Mixer Enabled | IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_ENABLED |
| External Mixer Harmonic | IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_HARMONI C |
| External Mixer Number Of Ports | IVIDOWNCONVERTER_ATTR_EXTERNAL_MIXER_NUMBER_ OF_PORTS |
| *IF Output* | |
| Active IF Output | IVIDOWNCONVERTER_ATTR_ACTIVE_IF_OUTPUT |
| IF Output Filter Bandwidth | IVIDOWNCONVERTER_ATTR_IF_OUTPUT_FILTER_BANDW IDTH |
| IF Output Count | IVIDOWNCONVERTER_ATTR_IF_OUTPUT_COUNT |
| IF Output Enabled | IVIDOWNCONVERTER_ATTR_IF_OUTPUT_ENABLED |
| IF Output Frequency | IVIDOWNCONVERTER_ATTR_IF_OUTPUT_FREQUENCY |
| IF Output Gain | IVIDOWNCONVERTER_ATTR_IF_OUTPUT_GAIN |
| IF Output Video Detector Bandwidth | IVIDOWNCONVERTER_ATTR_IF_OUTPUT_VIDEO_DETECT OR_BANDWIDTH |
| Is Settled | IVIDOWNCONVERTER_ATTR_IS_SETTLED |
| *Inherent IVI Attributes* | |
| *Advanced Session Information* | |
| Driver Setup | IVIDOWNCONVERTER_ATTR_DRIVER_SETUP |
| I/O Resource Descriptor | IVIDOWNCONVERTER_ATTR_IO_RESOURCE_DESCRIPTOR |
| Logical Name | IVIDOWNCONVERTER_ATTR_LOGICAL_NAME |
| *Class Driver Identification* | |

**Table 21-6** IviDownconverter C Attributes Hierarchy

| Category or Generic Attribute Name | C Defined Constant |
|---|---|
| Class Driver Class Spec Major Version | `IVIDOWNCONVERTER_ATTR_CLASS_DRIVER_CLASS_SPEC_MAJOR_VERSION` |
| Class Driver Class Spec Minor Version | `IVIDOWNCONVERTER_ATTR_CLASS_DRIVER_CLASS_SPEC_MINOR_VERSION` |
| Class Driver Description | `IVIDOWNCONVERTER_ATTR_CLASS_DRIVER_DESCRIPTION` |
| Class Driver Prefix | `IVIDOWNCONVERTER_ATTR_CLASS_DRIVER_PREFIX` |
| Class Driver Revision | `IVIDOWNCONVERTER_ATTR_CLASS_DRIVER_REVISION` |
| Class Driver Vendor | `IVIDOWNCONVERTER_ATTR_CLASS_DRIVER_VENDOR` |
| *Driver Capabilities* | |
| Class Group Capabilities | `IVIDOWNCONVERTER_ATTR_GROUP_CAPABILITIES` |
| Supported Instrument Models | `IVIDOWNCONVERTER_ATTR_SUPPORTED_INSTRUMENT_MODELS` |
| *Driver Identification* | |
| Specific Driver Class Spec Major Version | `IVIDOWNCONVERTER_ATTR_SPECIFIC_DRIVER_CLASS_SPEC_MAJOR_VERSION` |
| Specific Driver Class Spec Minor Version | `IVIDOWNCONVERTER_ATTR_SPECIFIC_DRIVER_CLASS_SPEC_MINOR_VERSION` |
| Specific Driver Description | `IVIDOWNCONVERTER_ATTR_SPECIFIC_DRIVER_DESCRIPTION` |
| Specific Driver Prefix | `IVIDOWNCONVERTER_ATTR_SPECIFIC_DRIVER_PREFIX` |
| Specific Driver Revision | `IVIDOWNCONVERTER_ATTR_SPECIFIC_DRIVER_REVISION` |
| Specific Driver Vendor | `IVIDOWNCONVERTER_ATTR_SPECIFIC_DRIVER_VENDOR` |
| *Instrument Identification* | |
| Instrument Firmware Revision | `IVIDOWNCONVERTER_ATTR_INSTRUMENT_FIRMWARE_REVISION` |
| Instrument Manufacturer | `IVIDOWNCONVERTER_ATTR_INSTRUMENT_MANUFACTURER` |
| Instrument Model | `IVIDOWNCONVERTER_ATTR_INSTRUMENT_MODEL` |
| *User Options* | |
| Cache | `IVIDOWNCONVERTER_ATTR_CACHE` |
| Interchange Check | `IVIDOWNCONVERTER_ATTR_INTERCHANGE_CHECK` |
| Query Instrument Status | `IVIDOWNCONVERTER_ATTR_QUERY_INSTRUMENT_STATUS` |
| Range Check | `IVIDOWNCONVERTER_ATTR_RANGE_CHECK` |
| Record Value Coercions | `IVIDOWNCONVERTER_ATTR_RECORD_COERCIONS` |
| Simulate | `IVIDOWNCONVERTER_ATTR_SIMULATE` |

**Table 21-6** IviDownconverter C Attributes Hierarchy

| Category or Generic Attribute Name | C Defined Constant |
|---|---|
| *Reference Oscillator* | |
| Reference Oscillator External Frequency | `IVIDOWNCONVERTER_ATTR_REFERENCE_OSCILLATOR_E XTERNAL_FREQUENCY` |
| Reference Oscillator Source | `IVIDOWNCONVERTER_ATTR_REFERENCE_OSCILLATOR_S OURCE` |
| Reference Oscillator Output Enabled | `IVIDOWNCONVERTER_ATTR_REFERENCE_OSCILLATOR_O UTPUT_ENABLED` |
| *RF Input* | |
| Active RF Input | `IVIDOWNCONVERTER_ATTR_ACTIVE_RF_INPUT` |
| Bypass | `IVIDOWNCONVERTER_ATTR_BYPASS` |
| Corrections Enabled | `IVIDOWNCONVERTER_ATTR_RF_INPUT_CORRECTIONS_E NABLED` |
| *Frequency Sweep* | |
| Frequency Sweep Mode | `IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_MODE` |
| Frequency Sweep Trigger Source | `IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_TRIGGE R_SOURCE` |
| Is Sweeping | `IVIDOWNCONVERTER_ATTR_IS_SWEEPING` |
| Number of Bands | `IVIDOWNCONVERTER_ATTR_NUM_BANDS` |
| *Analog* | |
| Frequency Sweep Start | `IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_START` |
| Frequency Sweep Stop | `IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_STOP` |
| Frequency Sweep Time | `IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_TIME` |
| *List* | |
| Frequency Sweep List Dwell | `IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_LIST_D WELL` |
| Frequency Sweep List Selected Name | `IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_LIST_S ELECTED_NAME` |
| Frequency Sweep List Single Step Enabled | `IVIDOWNCONVERTER_ATTR_FREQUENCY_SWEEP_LIST_S INGLE_STEP_ENABLED` |
| *Step* | |
| Frequency Step Dwell | `IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_DWELL` |
| Frequency Step Scaling | `IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_SCALING` |
| Frequency Step Single Step Enabled | `IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_SINGLE_ STEP_ENABLED` |
| Frequency Step Size | `IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_SIZE` |
| Frequency Step Start | `IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_START` |
| Frequency Step Stop | `IVIDOWNCONVERTER_ATTR_FREQUENCY_STEP_STOP` |

**Table 21-6** IviDownconverter C Attributes Hierarchy

| Category or Generic Attribute Name | C Defined Constant |
|---|---|
| Preselector Enabled | `IVIDOWNCONVERTER_ATTR_PRESELECTOR_ENABLED` |
| RF Input Attenuation | `IVIDOWNCONVERTER_ATTR_RF_INPUT_ATTENUATION` |
| RF Input Count | `IVIDOWNCONVERTER_ATTR_RF_INPUT_COUNT` |
| RF Input Coupling | `IVIDOWNCONVERTER_ATTR_RF_INPUT_COUPLING` |
| RF Input Frequency | `IVIDOWNCONVERTER_ATTR_RF_INPUT_FREQUENCY` |

# Appendix A   Specific Driver Development Guidelines

## A.1   Introduction

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

## A.2   Disabling Unused Extension Groups

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 IviDownconverterBypass Extension Group

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

**Table A-1.** Values for Disabling the Bypass Extension Group

| Attribute | Value |
|-----------|-------|
| Bypass | False |

### Disabling the IviDownconverterExternalMixer Extension Group

The IviDownconverterExternalMixer extension group does not affect instrument behavior.  Therefore, this specification does not recommend attribute values that disable the IviDownconverterExternalMixer extension group.

**Table A-2.** Values for Disabling the IviDownconverterExternalMixer Extension Group

| Attribute | Value |
|-----------|-------|
| External Mixer Enabled | False |

**Disabling the IviDownconverterFrequencyStep Extension Group**

Attribute values that effectively disable the IviDownconverterFrequencyStep extension group are shown in the following table. This functionality is also encapsulated in the Configure Frequency Sweep function.

**Table A-3.** Values for Disabling the IviDownconverterFrequencyStep Extension Group

| Attribute | Value |
|---|---|
| Frequency Sweep Mode | None |

**Disabling the IviDownconverterFrequencySweep Extension Group**

Attribute values that effectively disable the IviDownconverterFrequencySweep extension group are shown in the following table. This functionality is also encapsulated in the Configure Frequency Sweep function.

**Table A-4.** Values for Disabling the IviDownconverterFrequencySweep Extension Group

| Attribute | Value |
|---|---|
| Frequency Sweep Mode | None |

**Disabling the IviDownconverterFrequencySweepList Extension Group**

Attribute values that effectively disable the IviDownconverterFrequencySweepList extension group are shown in the following table. This functionality is also encapsulated in the Configure Frequency Sweep function.

**Table A-5.** Values for Disabling the IviDownconverterFrequencySweepList Extension Group

| Attribute | Value |
|---|---|
| Frequency Sweep Mode | None |

**Disabling the IviDownconverterBandCrossingInformation Extension Group**

The IviDownconverterBandCrossingInformation extension group does not affect instrument behavior. Therefore, this specification does not recommend attribute values that disable the IviDownconverterBandCrossingInformation extension group.

**Disabling the IviDownconverterSoftwareTrigger Extension Group**

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

**Disabling the IviDownconverterIFFilter Extension Group**

The IviDownconverterIFFilter extension group does not affect instrument behavior. Therefore, this specification does not recommend attribute values that disable the IviDownconverterIFFilter extension group.

**Disabling the IviDownconverterPreselector Extension Group**

Attribute values that effectively disable the IviDownconverterPreselector extension group are shown in the following table. This functionality is also encapsulated in the Configure Preselector Enabled function.

**Table A-6.** Values for Disabling the IviDownconverterPreslector Extension Group

| Attribute | Value |
|---|---|
| Preselector Enabled | False |

**Disabling the IviDownconverterVideoDetectorBandwidth Extension Group**

The IviDownconverterVideoDetectorBandwidth extension group does not affect instrument behavior. Therefore, this specification does not recommend attribute values that disable the IviDownconverterVideoDetectorBandwidth extension group.

**Disabling the IviDownconverterCalibration Extension Group**

The IviDownconverterCalibration extension group does not affect instrument behavior. Therefore, this specification does not recommend attribute values that disable the IviDownconverterCalibration extension group.

**Disabling the IviDownconverterReferenceOscillator Extension Group**

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

**Table A-7.** Values for Disabling the IviDownconverterReferenceOscillator Extension Group

| Attribute | Value |
|---|---|
| Reference Oscillator Source | Internal |

## A.3    Special Consideration for Query Instrument Status

Based on the value of Query Instr Status, the instrument may be queried by the specific driver to determine if it has encountered an error.

# Appendix B   Interchangeability Checking Rules

## B.1    Introduction

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

## B.2    When to Perform Interchangeability Checking

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

- The Interchange Check attribute is set to True

- The user calls one of the following functions:

    – Wait Until Settled

## B.3    Interchangeability Checking Rules

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

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

- An attribute that affects the behavior of the instrument is not in a state that the user specifies.

- The user sets a class driver defined attribute to an instrument-specific value.

- The user configures the value of an attribute that the class defines as read-only. In a few cases the class drivers define read-only attributes that specific drivers might implement as read/write.

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

### IviDownconverterBase Capability Group

1. If the IF Output Enabled attribute is set to False, no other attributes in the IviDownconverterBase capability group or in any extension group need be in a user-specified state.

2. If the External LO Enabled attribute is set to false, then the External LO Frequency attribute does not need to be in a user-specified state.

### IviDownconverterBypass Extension Group

1. If  the Bypass attribute is set to True, no attributes in the IviDownverterBase capability group or in any extension group need be in a user-specified state.

### IviDownconverterExternalMixer Extension Group

1. If the External Mixer Enabled attribute is set to False, then all remaining group attributes need not be in a user specified state.

2. If the External Mixer Bias Enabled attribute is set to False, then the External Mixer Bias and External Mixer Bias Limit attributes need not be in a user specified state.

**IviDownconverterFrequencyStep Extension Group**

1. If the Frequency Sweep Mode attribute is set to None, no attribute in the IviDownverterFrequencyStep extension group need be in a user-specified state.
2. If Frequency Step Single Step Enabled is set to true, the Frequency Step Dwell does not need to be in a user specified state.
3. If the IviDownconverterFrequencySweep extension group does not need to be checked for interchangeability, then attributes in the IviRFSigGenFrequencyStep extension group need not be in a user-specified state.

**IviDownconverterFrequencySweep Extension Group**

1. If the Frequency Sweep Mode attribute is set to None, no attribute in the IviDownverterFrequencySweep extension group need be in a user-specified state.
2. If the Frequency Sweep Mode attribute is set to a value other than Sweep, no other attributes besides Frequency Sweep Trigger Source need be in a user-specified state.

**IviDownconverterFrequencySweepList Extension Group**

1. If the Frequency Sweep Mode attribute is set to None, no attribute in the IviDownverterFrequencySweepList extension group need be in a user-specified state.
2. If Frequency Step Single Step Enabled is set to true, the Frequency Step Dwell does not need to be in a user specified state.
3. If the IviDownconverterFrequencySweepList extension group does not need to be checked for interchangeability, then attributes in the IviRFSigGenFrequencyStep extension group need not be in a user-specified state.

**IviDownconverterBandCrossingInformation Extension Group**

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

**IviDownconverterSoftwareTrigger Extension Group**

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

**IviDownconverterIFFilter Extension Group**

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

**IviDownconverterPreselector Extension Group**

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

**IviDownconverterVideoDetectorBandwidth Extension Group**

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

**IviDownconverterCalibration Extension Group**

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

**IviDownconverterReferenceOscillator Extension Group**

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