

NI-RFSG C Function Reference

September 2007, 371933D-01

This help file contains information about the NI-RFSG functions, attributes, and values that you can use when programming your application.

© 2004—2007 National Instruments Corporation. All rights reserved.

NI-RFSG Functions

Expand this book to view the NI-RFSG functions.

Purpose

Initializes the NI-RFSG device and performs the following initialization actions:

- Creates a new instrument driver session.
- Opens a session to the device you specify for the **resourceName** parameter.
- If the **reset** parameter is set to VI_TRUE, the niRFSG_init function resets the device to a known state.
- Returns a ViSession handle that you use to identify the NI-RFSG device in all subsequent NI-RFSG function calls.



Note Before initializing the NI 5670/5671/5672, an NI 5421/5441/5442 arbitrary waveform generator (AWG) module must be associated with the NI 5610 upconverter module in MAX. After association, pass the NI 5610 upconverter module device name to this function to initialize both modules. To change the AWG association, modify the NI 5610 Properties page in MAX, or use the [niRFSG_InitWithOptions](#) function to override the association in MAX. For more information about MAX association, refer to the [NI RF Signal Generators Getting Started Guide](#).

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
------	------	-------------

resourceName	ViRsrc	Specifies the resource name of the device to initialize.
---------------------	--------	--

For NI-DAQmx devices, the syntax is the device name specified in MAX. Typical default names for NI-DAQmx devices in MAX are Dev2 or PXISlot2. You can rename an NI-DAQmx device by right-clicking on the name in MAX and entering a new name.

You can also specify the name of an IVI logical name configured with the IVI Configuration utility. For additional information, refer to the IVI Drivers topic of the MAX Help.



Caution NI-DAQmx device names are not case-sensitive. However, all IVI names, such as logical names, are case-sensitive. If you use an IVI logical name, make sure the name is identical to the name shown in the IVI Configuration Utility.

Default Value: None

idQuery	ViBoolean	Specifies whether you want NI-RFSG to perform an ID query.
----------------	-----------	--

Defined Values:

VI_TRUE (1)	Perform ID query.
VI_FALSE (0)	Do not perform ID query.

Default Value: VI_TRUE

reset	ViBoolean	Specifies whether you want the to reset the NI-RFSG device during the initialization procedure.
--------------	-----------	---

Defined Values:

VI_TRUE (1)	Reset device.
VI_FALSE (0)	Do not reset device.

Default Value: VI_FALSE

Output

Name	Type	Description
------	------	-------------

vi	ViSession*	Returns a ViSession handle that you use to identify the NI-RFSG device in all subsequent NI-RFSG function calls.
-----------	------------	--

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ConfigureRF

C Function Prototype

```
ViStatus niRFSG_ConfigureRF (ViSession vi, ViReal64 frequency,  
                             ViReal64 powerLevel);
```

Purpose

Configures the frequency and power level of the RF output signal. The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
frequency	ViReal64	Specifies the frequency of the generated RF signal. For arbitrary waveform generation, this parameter specifies, in Hz, the center frequency of the signal. NI-RFSG sets the NIRFSG_ATTR_FREQUENCY attribute to this value. Refer to the specifications document that shipped with your device for allowable frequency settings.
powerLevel	ViReal64	Specifies the power level of the generated RF signal, expressed in dBm. By default, this parameter specifies the average power of the signal. To configure the power level of a waveform with varying power content, set the NIRFSG_ATTR_POWER_LEVEL_TYPE attribute to NIRFSG_VAL_PEAK_POWER .

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Configures the NI-RFSG device to generate a continuous wave (CW) sine tone, apply IQ (vector) modulation to the RF output signal, or generate arbitrary waveforms according to scripts. The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

generationMode	ViInt32	Specifies the mode used by NI-RFSG to generate an RF output signal. NI-RFSG sets the NIRFSG_ATTR_GENERATION_MODE attribute to this value.
-----------------------	---------	---

After initializing the NI RF signal generator or calling the [niRFSG_reset](#) or [niRFSG_ResetDevice](#) functions, this parameter is set to NIRFSG_VAL_CW.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_AllocateArbWaveform

C Function Prototype

ViStatus niRFSG_AllocateArbWaveform (ViSession vi, ViConstString name, ViI

Purpose

Allocates onboard memory space for the waveform. Use this function to specify the total size of a waveform before writing the data. You only need to use this function if you are calling the [niRFSG_WriteArbWaveform](#) function multiple times to write a large waveform in blocks.

If direct download is enabled, NI-RFSG reserves the appropriate amount of onboard memory for the specified waveform. If direct download is disabled, NI-RFSG allocates host memory that can hold the requested number of samples. Refer to the [NIRFSG_ATTR_DIRECT_DOWNLOAD](#) attribute for more information about enabling or disabling direct download.

The `niRFSG_WriteArbWaveform` function returns an error if you write more data than the amount of memory allocated for it. If you do not write the entire allocation, the signal generator generates uninitialized data in the unwritten portions of the waveform. If you allocate a waveform prior to writing it, NI-RFSG ignores the **moreDataPending** parameter in the `niRFSG_WriteArbWaveform` function. The NI-RFSG device must be in the Configuration state before you call the `niRFSG_AllocateArbWaveform` function.



Note Direct Download is *always* enabled on the NI 5672.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
name	ViConstString	Specifies the name used to identify the waveform. This string is case-insensitive and alphanumeric, and it does not use reserved words .
size_in_samples	ViInt32	Specifies the number of samples to reserve in the onboard memory for the specified waveform.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_WriteArbWaveform

C Function Prototype

```
ViStatus niRFSG_WriteArbWaveform (ViSession vi,  
    ViConstString waveformName, ViInt32 numberOfSamples,  
    ViReal64[] iData, ViReal64[] qData, ViBoolean moreDataPending);
```

Purpose

Writes an arbitrary waveform to the NI-RFSG device starting at the position of the last data written in onboard memory. This function takes as data input the I and Q vectors of a complex baseband signal. If the waveform is already allocated, the **moreDataPending** parameter is ignored. Refer to the [niRFSG_AllocateArbWaveform](#) function for more information about allocating waveforms. The NI-RFSG device must be in the Configuration state before you call this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
name	ViConstString	Specifies the name used to identify the waveform. This string is case-insensitive and alphanumeric, and it does not use reserved words .
numberOfSamples	ViInt32	Specifies the number of samples in both the iData and qData arrays. The iData and qData arrays must have the same length. If the NIRFSG_ATTR_ARB_WAVEFORM_QUANTUM attribute value is <i>q</i> , then the number of samples should be a multiple of <i>q</i> . The specified number of samples cannot be 0.
iData	ViReal64[]	Specifies the in-phase (I) component of the complex baseband signal.
qData	ViReal64[]	Specifies the quadrature (Q) component of the complex baseband signal.
moreDataPending	ViBoolean	Specifies whether or not the data block contains the end of the waveform. Set this parameter to VI_TRUE to allow data to be appended later to the waveform. Splitting the waveform into multiple data blocks can reduce the memory requirements of the write operation. Append data to a previously written waveform by using the same waveform in the name parameter. Set moreDataPending to VI_FALSE to indicate that this data block contains the end of the waveform. If the waveform is already allocated, this parameter is ignored.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_WriteArbWaveformComplexF64

C Function Prototype

```
ViStatus niRFSG_WriteArbWaveform (ViSession vi,  
    ViConstString waveformName, ViInt32 numberOfSamples,  
    NIComplexNumber wfmData[], ViBoolean moreDataPending);
```

Purpose

Writes an arbitrary waveform to the NI-RFSG device starting at the position of the last data written in onboard memory. This function takes as data input the data array of a complex baseband signal. If the waveform is already allocated, the **moreDataPending** parameter is ignored. Refer to the [niRFSG_AllocateArbWaveform](#) function for more information about allocating waveforms. The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
name	ViConstString	Specifies the name used to identify the waveform. This string is case-insensitive and alphanumeric, and it does not use reserved words .
numberOfSamples	ViInt32	Specifies the number of samples in both the data array.
data	NIComplexNumber[]	Specifies the array of data to load into the waveform. The array must have at least as many elements as the value in the size_in_samples parameter in the niRFSG_AllocateArbWaveform function.
moreDataPending	ViBoolean	Specifies whether or not the data block contains the end of the waveform. Set this parameter to VI_TRUE to allow data to be appended later to the waveform. Splitting the waveform into multiple data blocks can reduce the memory requirements of the write operation. Append data to a previously written waveform by using the same waveform in the name parameter. Set moreDataPending to VI_FALSE to indicate that this data block contains the end of the waveform. If the waveform is already allocated, this parameter is ignored.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_WriteArbWaveformComplexI16

C Function Prototype

```
ViStatus niRFSG_WriteArbWaveform (ViSession vi,  
    ViConstString waveformName, ViInt32 numberOfSamples,  
    NIComplexI16 wfmData[]);
```

Purpose

Writes an arbitrary waveform to the NI-RFSG device starting at the position of the last data written in onboard memory. This function takes as data input the data array of a complex baseband signal. If the waveform is already allocated, the **moreDataPending** parameter is ignored. Refer to the [niRFSG_AllocateArbWaveform](#) function for more information about allocating waveforms. The NI-RFSG device must be in the Configuration state before calling this function.



Note This function only supports NIRFSG_VAL_PEAK_POWER mode as specified in the [NIRFSG_ATTR_POWER_LEVEL_TYPE](#) attribute. If a waveform is downloaded using this function, NIRFSG_ATTR_POWER_LEVEL_TYPE cannot be changed to NIRFSG_VAL_AVERAGE_POWER mode without causing error in the output.

Supported Devices: NI 5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
name	ViConstString	Specifies the name used to identify the waveform. This string is case-insensitive and alphanumeric, and it does not use reserved words .
numberOfSamples	ViInt32	Specifies the number of samples in the data array.
data	NIComplexNumber[]	Specifies the array of data to load into the waveform. The array must have at least as many elements as the value in the size_in_samples parameter in the niRFSG_AllocateArbWaveform function.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_SelectArbWaveform

C Function Prototype

```
ViStatus niRFSG_SelectArbWaveform (ViSession vi, ViConstString name);
```

Purpose

Specifies the waveform that is generated upon a call to the [niRFSG_Initiate](#) function when the **generationMode** parameter of the [niRFSG_ConfigureGenerationMode](#) function is set to `NIRFSG_VAL_ARB_WAVEFORM`. You must specify a waveform using the **name** parameter if you have written multiple waveforms. The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

name	ViConstString	Specifies the name of the stored waveform to generate. This is a case-insensitive alphanumeric string that does not use reserved words. NI-RFSG sets the NIRFSG_ATTR_ARB_SELECTED_WAVEFORM attribute to this value.
-------------	---------------	---

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ClearArbWaveform

IviFgenArbWfm Capability Group

C Function Prototype

```
ViStatus niRFSG_ClearArbWaveform (ViSession vi, ViConstString name);
```

Purpose

Deletes a specified waveform from the pool of currently defined waveforms. The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

name ViConstString Name of the stored waveform to delete.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ClearAllArbWaveforms

C Function Prototype

```
ViStatus niRFSG_ClearAllArbWaveforms (ViSession vi);
```

Purpose

Deletes all currently defined waveforms and scripts. The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Configures the signal [bandwidth](#) of the arbitrary waveform. NI-RFSG defines *signal bandwidth* as twice the maximum baseband signal deviation from 0 Hz. Usually, the baseband signal center frequency is 0 Hz. In such cases, the signal bandwidth is simply the baseband signal's minimum frequency subtracted from its maximum frequency, or f_{\max} minus f_{\min} . The driver uses this value to optimally configure the center frequency of the upconverter to help minimize phase noise. The generated signal will not be filtered to achieve the set bandwidth. However, specifying a bandwidth smaller than the actual bandwidth of the signal could potentially result in spectral distortion.



Note Based on your signal bandwidth, NI-RFSG decides whether to configure the upconverter center frequency in increments of 1 or 5 MHz. Failure to configure this attribute may result in the signal being placed out of the upconverter passband.

The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

signalBandwidth	ViReal64	Specifies the signal bandwidth used by NI-RFSG to generate an RF output signal. NI-RFSG sets the NIRFSG_ATTR_SIGNAL_BANDWIDTH attribute to this value.
------------------------	----------	--

Valid Values:
0 Hz to 20 MHz

Default Value: 100

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ConfigureDigitalModulationUserDefined

C Function Prototype

ViStatus niRFSG_ConfigureDigitalModulationUserDefinedWaveform (ViSession

Purpose

Specifies the message signal used for digital modulation when [NIRFSG_ATTR_DIGITAL_MODULATION_WAVEFORM_TYPE](#) is set to NIRFSG_VAL_USER_DEFINED.

Supported Devices: NI 5650/5651/5652

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
numberOfSamples	ViInt32	Specifies the number of samples in the message signal.
userDefinedWaveform	ViInt8[]	Specifies the user-defined message signal used for digital modulation.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ConfigureSoftwareStartTrigger

C Function Prototype

```
ViStatus niRFSG_ConfigureSoftwareStartTrigger (ViSession vi);
```

Purpose

Configures the Start trigger for software triggering. Refer to the [niRFSG_SendSoftwareEdgeTrigger](#) function for more information about using a software trigger.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ConfigureDigitalEdgeStartTrigger

C Function Prototype

```
ViStatus niRFSG_ConfigureDigitalEdgeStartTrigger (ViSession vi,  
ViConstString source, ViInt32 edge);
```

Purpose

Configures the Start trigger for digital edge triggering.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

source	ViConstString	Specifies the source terminal for the digital edge trigger. NI-RFSG sets NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_SOURCE to this value.
---------------	---------------	---

edge	ViInt32	Specifies the active edge for the Start trigger. NI-RFSG sets NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_EDGE to this value.
-------------	---------	--

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_DisableStartTrigger

C Function Prototype

```
ViStatus niRFSG_DisableStartTrigger (ViSession vi);
```

Purpose

Configures the device to not wait for a Start trigger after the [niRFSG_Initiate](#) function is called. Calling the niRFSG_DisableStartTrigger function is only necessary if the Start trigger has been previously configured and now must be disabled. The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: N/A

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Configures the Script trigger for software triggering. Refer to the [niRFSG_SendSoftwareEdgeTrigger](#) function for more information about using the software Script trigger.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

triggerIdentifier ViConstString Specifies which of the four available Script triggers is configured.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ConfigureDigitalEdgeScriptTrigger

Specific Function

C Function Prototype

```
ViStatus niRFSG_ConfigureDigitalEdgeScriptTrigger (ViSession vi,  
ViConstString triggerIdentifier, ViConstString source, ViInt32 edge);
```

Purpose

Configures the specified Script trigger for digital edge triggering.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: N/A
triggerIdentifier	ViConstString	Specifies which of the four available Script triggers is configured.
source	ViConstString	Specifies the source terminal for the digital edge Script trigger. NI-RFSG sets NIRFSG_ATTR_DIGITAL_EDGE_SCRIPT_TRIGGER_SOURCE to this value.
edge	ViInt32	Specifies the active edge for the digital edge Script trigger. NI-RFSG sets NIRFSG_ATTR_DIGITAL_EDGE_SCRIPT_TRIGGER_EDGE to this value.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Configures a specified Script trigger for digital level triggering.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
triggerIdentifier	ViConstString	Specifies which of the four available Script triggers is configured.
source	ViConstString	Specifies the trigger source terminal for the digital level script trigger. NI-RFSG sets NIRFSG_ATTR_DIGITAL_LEVEL_SCRIPT_TRIGGER_SOURCE to this value.
Level	ViInt32	Specifies the active level for the digital level script trigger. NI-RFSG sets NIRFSG_ATTR_DIGITAL_LEVEL_SCRIPT_TRIGGER_ACTIVE_LEVEL to this value.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Configures the device to not wait for the specified Script trigger after the [niRFSG_Initiate](#) function is called. Calling the niRFSG_DisableScriptTrigger function is only necessary if the Script trigger has been previously configured and now must be disabled. The NI-RFSG device must be in the Configuration state before you call this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

triggerIdentifier ViConstString Specifies which of the four available Script triggers is configured.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_SendSoftwareEdgeTrigger

C Function Prototype

```
ViStatus niRFSG_SendSoftwareEdgeTrigger (ViSession vi, ViInt32 trigger,  
ViConstString triggerIdentifier);
```

Purpose

Forces a particular trigger to occur. The specified trigger is generated regardless of whether the trigger has been configured as a Software trigger.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
trigger	ViInt32	Specifies the trigger to assert.
triggerIdentifier	ViConstString	Specifies which of the four available Script triggers is configured.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ExportSignal

C Function Prototype

```
ViStatus niRFSG_ExportSignal (ViSession vi, ViInt32 signal,  
                             ViConstString signalIdentifier, ViConstString outputTerminal);
```

Purpose

Exports various signals, clocks, and events from the signal generator to the RTSI lines, front panel, or other external terminals. The NI-RFSG device must be in the Configuration state before you call this function.

You can clear a previously routed signal by exporting the signal to " " (empty string).

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

signal	ViInt32	<p>Specifies the signal to route.</p> <ul style="list-style-type: none"> ✦ Onboard reference clock output terminal is not configurable on the NI 5670/5671/5672 device. ✦ Triggers and Marker events are not available on the NI 5650/5651/5652 device.
---------------	---------	---

Defined Values:

NIRFSG_VAL_START_TRIGGER	Start trigger
NIRFSG_VAL_SCRIPT_TRIGGER	Script trigger
NIRFSG_VAL_MARKER_EVENT	Marker event
NIRFSG_VAL_REF_CLOCK	The onboard 10 MHz synchronization clock (PCI chassis only)

Default Value: NIRFSG_VAL_START_TRIGGER

signalIdentifier ViConstString Specifies which instance of the selected signal to export.

Defined Values:

NIRFSG_VAL_SCRIPT_TRIGGER0	Script trigger 0
NIRFSG_VAL_SCRIPT_TRIGGER1	Script trigger 1
NIRFSG_VAL_SCRIPT_TRIGGER2	Script trigger 2
NIRFSG_VAL_SCRIPT_TRIGGER3	Script trigger 3
NIRFSG_VAL_MARKER_EVENT0	Marker 0

NIRFSG_VAL_MARKER_EVENT1	Marker 1
NIRFSG_VAL_MARKER_EVENT2	Marker 2
NIRFSG_VAL_MARKER_EVENT3	Marker 3
" " (empty string)	None (no signal to export)

This parameter is useful when the **signal** parameter is set to NIRFSG_VAL_SCRIPT_TRIGGER or NIRFSG_VAL_MARKER_EVENT. Otherwise, set the **signalIdentifier** parameter to "" (empty string).

outputTerminal ViConstString Specifies the terminal where the signal will be exported.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ConfigureRefClock

C Function Prototype

```
ViStatus niRFSG_ConfigureRefClock (ViSession vi,  
                                   ViConstString clockSource, ViReal64 refClockRate);
```


Purpose

Configures the NI-RFSG device reference clock. The reference clock ensures that the NI-RFSG devices are operating from a common timebase. The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from niRFSG_InitWithOptions function and identifies a particular instrument sess

Default Value: None

clockSource ViConstString Specifies the source of reference clock signal. Only certain combinations of [NIRFSG_ATTR_PXI_CHASSIS_CLK10_SOURCE](#) are valid, as shown in the

Ref Clock Source Setting	Valid
NIRFSG_VAL_ONBOARD_CLK_STR	NIRFSG_VAL NIRFSG_VAL
NIRFSG_VAL_REF_IN_STR	NIRFSG_VAL
NIRFSG_VAL_PXI_CLK10_STR	NIRFSG_VAL NIRFSG_VAL

NI-RFSG sets [NIRFSG_ATTR_REF_CLOCK_SOURCE](#) to this value.

refClockRate ViReal64 Specifies the reference clock rate, expressed in Hz. NI-RFSG sets [NIRFSG_ATTR_REF_CLOCK_RATE](#) to this value.

Default Value: 10E6 (10 MHz; this is the only supported value)

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Specifies the signal to drive the 10 MHz reference clock on the PXI backplane. This option can only be configured when the NI PXI-5610 is in Slot 2 of the PXI chassis. The NI-RFSG device must be in the Configuration state before you call this function.

Supported Devices: NI 5670/5671

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

pxiClk10Source ViConstString Specifies the signal to drive the 10 MHz reference clock to the PXI backplane (default), **RefIn**, or **OnboardClock** as the value for this control. Only the values **pxiClk10Source** and [NIRFSG_ATTR_REF_CLOCK_SOURCE](#) are valid,

Valid PXI Chassis Clk10 Setting	Valid
NIRFSG_VAL_NONE, NIRFSG_VAL_ONBOARD_CLK_STR	NIRFSG_VAL_ONBOARD_CLK_STR
NIRFSG_VAL_NONE, NIRFSG_VAL_REF_IN_STR	NIRFSG_VAL_REF_IN_STR
NIRFSG_VAL_NONE, NIRFSG_VAL_REF_IN_STR	NIRFSG_VAL_REF_IN_STR

NI-RFSG sets the [NIRFSG_ATTR_PXI_CHASSIS_CLK10_SOURCE](#) attribute.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_WriteScript

C Function Prototype

```
ViStatus niRFSG_WriteScript (ViSession vi, ViConstString script);
```


Purpose

Identifies a string containing a script that controls waveform generation in Script mode. Use the [niRFSG_ConfigureGenerationMode](#) function to specify Script mode before calling `niRFSG_WriteScript`. Refer to [Scripting Instructions](#) for information about using scripts.

The NI-RFSG device must be in the Configuration state before calling this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
script	ViConstString	Specifies a string containing a syntactically correct script. NI-RFSG supports multiple scripts that may be selected by name with the NIRFSG_ATTR_SELECTED_SCRIPT attribute. Refer to Scripting Instructions for more information about using scripts.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Enables or disables signal output. You can call this function in any software state, and it does not change the current software state. Setting **output_enabled** to VI_FALSE while in the Generation state stops signal generation, although generation continues internally.

Supported Devices: NI 5610 (upconverter only mode), NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

output_enabled ViBoolean Specifies whether you want to enable or disable the output.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_SetAttributeViInt32

C Function Prototype

```
ViStatus niRFSG_SetAttributeViInt32 (ViSession vi,  
    ViConstString channelName, ViAttr attributeID,  
    ViInt32 attributeValue);
```


Purpose

Sets the value of a ViInt32 attribute.

Use this low-level function to set the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid or is different than the value you specify.

NI-RFSG contains high-level functions that set most of the instrument attributes. Use the high-level driver functions as much as possible, as they handle order dependencies and multithread locking. The high-level functions also perform status checking only after setting all of the attributes. In contrast, when you set multiple attributes using the SetAttribute functions, the functions check the instrument status after each call.

Also, when state caching is enabled, the high-level functions that configure multiple attributes perform instrument I/O only for the attributes whose value you change. Thus, you can safely call the high-level functions without the penalty of redundant instrument I/O.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViInt32	Specifies the value to which you want to set the attribute.



Note Some values may not be valid. The allowed values depend on the current settings of the instrument session.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_SetAttributeViInt64

C Function Prototype

```
ViStatus niRFSG_SetAttributeViInt64 (ViSession vi,  
    ViConstString Channel_Name, ViAttr Attribute_ID,  
    ViInt64 Attribute_Value);
```

Purpose

Sets the value of a VInt64 attribute.

Use this low-level function to set the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid or is different than the value you specify.

NI-RFSG contains high-level functions that set most of the instrument attributes. Use the high-level driver functions as much as possible, as they handle order dependencies and multithread locking. The high-level functions also perform status checking only after setting all of the attributes. In contrast, when you set multiple attributes using the SetAttribute functions, the functions check the instrument status after each call.

Also, when state caching is enabled, the high-level functions that configure multiple attributes perform instrument I/O only for the attributes whose value you change. Thus, you can safely call the high-level functions without the penalty of redundant instrument I/O.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL.
--------------------	---------------	---

Default Value: "" (empty string)

attributeID	ViAttr	Pass the ID of an attribute.
--------------------	--------	------------------------------

Attribute_Value	ViInt64	Pass the value to which you want to set the attribute.
------------------------	---------	--



Note Some values may not be valid. The allowed values depend on the current settings of the instrument session.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Sets the value of a ViReal64 attribute.

Use this low-level function to set the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid or is different than the value you specify.

NI-RFSG contains high-level functions that set most of the instrument attributes. Use the high-level driver functions as much as possible, as they handle order dependencies and multithread locking. The high-level functions also perform status checking only after setting all of the attributes. In contrast, when you set multiple attributes using the SetAttribute functions, the functions check the instrument status after each call.

Also, when state caching is enabled, the high-level functions that configure multiple attributes perform instrument I/O only for the attributes whose value you change. Thus, you can safely call the high-level functions without the penalty of redundant instrument I/O.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViReal64	Pass the value to which you want to set the attribute.



Note Some values may not be valid. The allowed values depend on the current settings of the instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Sets the value of a ViString attribute.

Use this low-level function to set the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid or is different than the value you specify.

NI-RFSG contains high-level functions that set most of the instrument attributes. Use the high-level driver functions as much as possible, as they handle order dependencies and multithread locking. The high-level functions also perform status checking only after setting all of the attributes. In contrast, when you set multiple attributes using the SetAttribute functions, the functions check the instrument status after each call.

Also, when state caching is enabled, the high-level functions that configure multiple attributes perform instrument I/O only for the attributes whose value you change. Thus, you can safely call the high-level functions without the penalty of redundant instrument I/O.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViConstString	Pass the value to which you want to set the attribute.



Note Some values may not be valid. The allowed values depend on the current settings of the instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Sets the value of a ViBoolean attribute.

Use this low-level function to set the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid or is different than the value you specify.

NI-RFSG contains high-level functions that set most of the instrument attributes. Use the high-level driver functions as much as possible, as they handle order dependencies and multithread locking. The high-level functions also perform status checking only after setting all of the attributes. In contrast, when you set multiple attributes using the SetAttribute functions, the functions check the instrument status after each call.

Also, when state caching is enabled, the high-level functions that configure multiple attributes perform instrument I/O only for the attributes whose value you change. Thus, you can safely call the high-level functions without the penalty of redundant instrument I/O.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViBoolean	Pass the value to which you want to set the attribute.



Note Some values may not be valid. The allowed values depend on the current settings of the instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Sets the value of a ViSession attribute.

Use this low-level function to set the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid or is different than the value you specify.

NI-RFSG contains high-level functions that set most of the instrument attributes. Use the high-level driver functions as much as possible, as they handle order dependencies and multithread locking. The high-level functions also perform status checking only after setting all of the attributes. In contrast, when you set multiple attributes using the SetAttribute functions, the functions check the instrument status after each call.

Also, when state caching is enabled, the high-level functions that configure multiple attributes perform instrument I/O only for the attributes whose value you change. Thus, you can safely call the high-level functions without the penalty of redundant instrument I/O.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViSession	Pass the value to which you want to set the attribute.



Note Some values may not be valid. The allowed values depend on the current settings of the instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_GetAttributeViInt32

C Function Prototype

```
ViStatus niRFSG_GetAttributeViInt32 (ViSession vi,  
    ViConstString channelName, ViAttr attributeID,  
    ViInt32* attributeValue);
```


Purpose

Queries the value of a ViInt32 attribute.

Use this low-level function to get the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL.
--------------------	---------------	---

Default Value: "" (empty string)

attributeID	ViAttr	Pass the ID of an attribute.
--------------------	--------	------------------------------

Output

Name	Type	Description
attributeValue	ViInt32*	Returns the current value of the attribute. Pass the address of a ViInt32 variable.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_GetAttributeViInt64

C Function Prototype

```
ViStatus niRFSG_GetAttributeViInt64 (ViSession vi, ViConstString  
    Channel_Name, ViAttr Attribute_ID, ViInt64* Attribute_Value);
```

Purpose

Queries the value of a `ViInt64` attribute.

You can use this low-level function to get the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL.
--------------------	---------------	---

Default Value: "" (empty string)

attributeID	ViAttr	Pass the ID of an attribute.
--------------------	--------	------------------------------

Output

Name	Type	Description
Attribute_Value	ViInt64*	Returns the current value of the attribute. Pass the address of a ViInt64 variable.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Queries the value of a ViReal64 attribute.

Use this low-level function to get the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.

Output

Name	Type	Description
attributeValue	ViReal64*	Returns the current value of the attribute. Pass the address of a ViReal64 variable.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Queries the value of a ViString attribute.

Use this low-level function to get the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid.

You must provide a ViString (ViChar array) to serve as a buffer for the value. Pass the number of bytes in the buffer as the Buffer Size parameter. If the current value of the attribute, including the terminating NULL byte, is larger than the size you indicate in the buffer size parameter, the function copies buffer size-1 bytes into the buffer, places an ASCII NULL byte at the end of the buffer, and returns the buffer size you must pass to get the entire value. For example, if the value is "123456" and the buffer size is 4, the function places "123" into the buffer and returns 7.

To call this function to get only the required buffer size, pass 0 for the buffer size and VI_NULL for the attribute value buffer.

If you want the function to fill in the buffer regardless of the number of bytes in the value, pass a negative number for the **bufferSize** parameter.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
bufferSize	ViInt32	Pass the number of bytes in the ViChar buffer you specify for the attributeValue parameter. If you pass a negative number, the function copies the value to the buffer regardless of the number of bytes in the value. If you pass 0, you can pass VI_NULL for the attributeValue parameter.

Output

Name	Type	Description
attributeValue	ViChar[]	The buffer in which the function returns the current value of the attribute. The buffer must be of type ViChar and have at least as many bytes as indicated in the bufferSize parameter. If you specify 0 for the bufferSize parameter, you can pass VI_NULL for this parameter.

Return Value

Name	Type	Description
status	ViStatus	

Purpose

Queries the value of a ViBoolean attribute.

Use this low-level function to get the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.

Output

Name	Type	Description
attributeValue	ViBoolean*	Returns the current value of the attribute. Pass the address of a ViBoolean variable.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Queries the value of a ViSession attribute.

Use this low-level function to get the values of inherent IVI attributes, class-defined attributes, and instrument-specific attributes. If the attribute represents an instrument state, this function performs instrument I/O in the following cases:

- State caching is disabled for the entire session or for the particular attribute.
- State caching is enabled, and the currently cached value is invalid.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.

Output

Name	Type	Description
attributeValue	ViSession*	Returns the current value of the attribute. Pass the address of a ViSession variable.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Purpose

Checks the validity of a value you specify for a ViInt32 attribute.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViInt32	Pass the value that you want to verify as a valid value for the attribute.



Note Some of the values might not be valid depending on the current settings of the instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_CheckAttributeViInt64

C Function Prototype

```
ViStatus niRFSG_CheckAttributeViInt64 (ViSession vi,  
    ViConstString Channel_Name, ViAttr Attribute_ID,  
    ViInt64 Attribute_Value);
```

Purpose

Checks the validity of a value you specify for a ViInt64 attribute.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
Attribute_Value	ViInt64	Pass the value that you want to verify as a valid value for the attribute.



Note Some of the values might not be valid depending on the current settings of the instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_CheckAttributeViReal64

C Function Prototype

```
ViStatus niRFSG_CheckAttributeViReal64 (ViSession vi,  
    ViConstString channelName, ViAttr attributeID,  
    ViReal64 attributeValue);
```


Purpose

Checks the validity of a value you specify for a ViReal64 attribute.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViReal64	Pass the value that you want to verify as a valid value for the attribute.



Note Some of the values might not be valid depending on the current settings of the instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_CheckAttributeViString

C Function Prototype

```
ViStatus niRFSG_CheckAttributeViString (ViSession vi,  
    ViConstString channelName, ViAttr attributeID,  
    ViConstString attributeValue);
```

Purpose

Checks the validity of a value you specify for a ViString attribute.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViConstString	Pass the value that you want to verify as a valid value for the attribute. The value must be a NULL-terminated string.



Note Some of the values might not be valid depending on the current settings of the instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_CheckAttributeViBoolean

C Function Prototype

```
ViStatus niRFSG_CheckAttributeViBoolean (ViSession vi,  
    ViConstString channelName, ViAttr attributeID,  
    ViBoolean attributeValue);
```


Purpose

Checks the validity of a value you specify for a ViBoolean attribute.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViBoolean	Pass the value that you want to verify as a valid value for the attribute.



Note Some of the values might not be valid depending on the current settings of the instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_CheckAttributeViSession

C Function Prototype

```
ViStatus niRFSG_CheckAttributeViSession (ViSession vi,  
    ViConstString channelName, ViAttr attributeID,  
    ViSession attributeValue);
```

Purpose

Checks the validity of a value you specify for a ViSession attribute.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
channelName	ViConstString	Specifies the channel to which the attribute applies if this attribute is channel-based. If the attribute is not channel-based, set this parameter to "" (empty string) or VI_NULL. Default Value: "" (empty string)
attributeID	ViAttr	Pass the ID of an attribute.
attributeValue	ViSession	Pass the value that you want to verify as a valid value for the attribute.



Note Some of the values might not be valid depending on the current settings of the instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_Initiate

C Function Prototype

```
ViStatus niRFSG_Initiate (ViSession vi);
```


Purpose

Initiates signal generation, causing the NI-RFSG device to leave the Configuration state and enter the Generation state. If the settings have not been committed to the device before you call this function, they are committed with this function. The operation returns when the RF output signal settles.

To return to the Configuration state, call the [niRFSG_Abort](#) function.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_CheckGenerationStatus

C Function Prototype

```
ViStatus niRFSG_CheckGenerationStatus (ViSession vi, ViBoolean* isDone);
```

Purpose

Checks the status of the generation. Call this function to check for any errors that might occur during the signal generation or to check whether the device has finished generating.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Output

Name	Type	Description
isDone	ViBoolean*	Returns information about the completion of signal generation.

Defined Values:

VI_TRUE	Signal generation is complete.
VI_FALSE	Signal generation is occurring.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_Abort

C Function Prototype

```
ViStatus niRFSG_Abort (ViSession vi);
```


Purpose

Aborts a previously initiated signal generation.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_close

C Function Prototype

```
ViStatus niRFSG_close (ViSession vi);
```

Purpose

Performs the following closing actions:

- Aborts any signal generation in progress.
- Destroys the instrument driver session.



Note After calling this function, you cannot use NI-RFSG again until you call the [niRFSG_init](#) function or the [niRFSG_InitWithOptions](#) function.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_InitWithOptions

C Function Prototype

```
ViStatus niRFSG_InitWithOptions (ViRsrc resourceName, ViBoolean idQuery,  
                                ViBoolean resetDevice, ViConstString optionString, ViSession* vi);
```


Purpose

Initializes the NI-RFSG device (the upconverter module and the AWG module). This function can receive the AWG **resourceName** parameter through the **optionString** parameter.

This function performs the following initialization actions:

- Creates a new IVI instrument driver session.
- Opens a session to the device you specify using the **resourceName** parameter.
- If the **reset** parameter is set to VI_TRUE, this function resets the device to a known state.
- Returns a ViSession handle that you use to identify the NI-RFSG device in all subsequent NI-RFSG function calls.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name **Type**
resourceName ViRsrc

Description

Specifies the resource name of the device to initialize.

For NI-DAQmx devices, the syntax is the device name specified in MAX for NI-DAQmx devices in MAX are Dev2 or PXISlot2. You can rename an right-clicking on the name in MAX and entering a new name.

You can also specify the name of an IVI logical name configured with the For additional information, refer to the IVI Drivers topic of the MAX Help.



Caution NI-DAQmx device names are not c
 However, all IVI names, such as logical nam
 sensitive. If you use an IVI logical name, mal
 name is identical to the name shown in the I
 Utility.

Default Value: None

idQuery ViBoolean

Specifies whether you want NI-RFSG to perform an ID query.

Defined Values:

VI_TRUE (1)	Perform ID query.
VI_FALSE (0)	Do not perform ID query.

Default Value: VI_TRUE

reset ViBoolean

Specifies whether you want the to reset the NI-RFSG device during the i

Defined Values:

VI_TRUE (1)	Reset device.
VI_FALSE (0)	Do not reset device.

Default Value: VI_FALSE

optionString ViConstString

Specifies the initial value of certain attributes for the session. The followi
 attributes and the name you pass in this parameter to identify the attribu

Name	Attribute Name
RangeCheck	NIRFSG_ATTR_RANGE_CHECK
QueryInstrStatus	NIRFSG_ATTR_QUERY_INSTR
Cache	NIRFSG_ATTR_CACHE

RecordCoercions	NIRFSG_ATTR_RECORD_COEF
DriverSetup	NIRFSG_ATTR_DRIVER_SETUP
DriverSetup	NIRFSG_ATTR_SIMULATE

The format of this string consists of the following relations:

"AttributeName=Value"

where

AttributeName is the name of the attribute and

Value is the value to which the attribute will be set. To set multiple attribute assignments with a comma, as shown in the following option string:

"RangeCheck=1,QueryInstrStatus=0,Cache=1,DriverSetup=AWG:pxi1slot4"

To simulate a particular AWG module, use the following option string:

"Simulate=1,DriverSetup=AWGModel:pxi1slot4"

This option string is valid for simulation purposes only.

Output

Name	Type	Description
vi	ViSession*	Returns a ViSession handle that you use to identify the NI-RFSG device RFSG function calls.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_Commit

C Function Prototype

```
ViStatus niRFSG_Commit (ViSession vi);
```

Purpose

Asserts the configured hardware parameters. This function verifies attribute values, reserves the device, and commits the attribute values to the device. If the attributes values are all valid, the device configuration matches the session configuration. Calling this function moves the NI-RFSG device from the Configuration state to the Committed state. After calling this function, changing any attribute reverts the NI-RFSG device to the Configuration state.

Supported devices: NI 5610 (upconverter only mode), NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_PerformThermalCorrection

C Function Prototype

```
ViStatus niRFSG_PerformThermalCorrection (ViSession vi);
```

Purpose

Corrects for any signal drift due to environmental temperature variation when generating the same signal for extended periods of time without a parameter change. NI-RFSG compensates automatically for variations in device temperature when any signal generation parameters are changed. Under normal circumstances of short-term signal generation, NI-RFSG automatic compensation ensures stable power levels, and you do not need to use this function.

Use this function when generating the same signal for an extended period of time in a temperature-fluctuating environment. The NI-RFSG device must be in the Generation state before calling this function.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_WaitUntilSettled

C Function Prototype

```
ViStatus niRFSG_WaitUntilSettled (ViSession vi,  
    ViInt32 maxTimeMilliseconds);
```

Purpose

Waits until the RF output signal has settled. This function is useful for devices that support on-the-fly configuration changes (changes while in the Generation state). Call this function after making a dynamic change to wait for the output to settle.

You can also call this function after calling the [niRFSG_Commit](#) function to wait for changes to settle. The `niRFSG_WaitUntilSettled` function is not needed after calling the [niRFSG_Initiate](#) function because the `niRFSG_Initiate` function does not return until the output is settled.

Supported Devices: NI 5610 (upconverter only mode), NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None
maxTimeMilliseconds	ViInt32	Defines the maximum time the function waits for the output to be settled. If the maximum time is exceeded, this function returns the an error. If you set this parameter to -1, NI-RFSG waits indefinitely until it is settled. The units are expressed in milliseconds. Default Value: 10000

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_reset

C Function Prototype

```
ViStatus niRFSG_reset (ViSession vi);
```

Purpose

Resets all attributes to their default values and moves the NI-RFSG device to the Configuration state. This function aborts the generation, clears all routes, and resets session attributes to the initial values. During a reset, routes of signals between this and other devices are released, regardless of which device created the route.

Generally, calling this function instead of the [niRFSG_ResetDevice](#) function is acceptable. The `niRFSG_reset` function executes faster than the `niRFSG_ResetDevice` function.

Supported Devices: NI 5610 (upconverter only mode), NI 5650/5651/5652/5671/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ResetDevice

C Function Prototype

```
ViStatus niRFSG_ResetDevice (ViSession vi);
```

Purpose

Performs a hard reset on the device which consists of the following actions:

- Signal generation is stopped
- All routes are released
- External bidirectional terminals are tristated
- FPGAs are reset
- Hardware is configured to its default state
- All session attributes are reset to their default states

During a reset, routes of signals between this and other devices are released, regardless of which device created the route.

- **NI 5610, NI 5670/5671/5672**

— After calling this function, the device requires 25 seconds before returning to full functionality. NI-RFSG enforces this condition by adding a wait, if needed, the next time you try to access the device.



Note You must call the `niRFSG_ResetDevice` function if the NI-RFSG device has shut down due to high temperature conditions.

Supported Devices: NI 5610 (upconverter only mode), NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ResetWithDefaults

C Function Prototype

```
ViStatus niRFSG_ResetWithDefaults (ViSession vi);
```

Purpose

Performs a software reset of the device, returning it to the default state and applying any initial default settings from the IVI Configuration Store.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_GetChannelName

C Function Prototype

```
ViStatus niRFSG_GetChannelName (ViSession vi, ViInt32 Index, ViInt32  
    BufferSize, ViChar Channel_Name[]);
```

Purpose

Returns the channel string that is in the channel table at an index you specify.

Supported Devices:

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Index	ViInt32	Specifies a 1-based index into the channel table.
BufferSize	ViInt32	Specifies the size of the buffer for the channel string

Output

Name	Type	Description
Channel_Name	ViChar[]	Returns a channel string from the channel table at the index you specify in the Index parameter. Do not modify the contents of the channel string.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_self_test

C Function Prototype

```
ViStatus niRFSG_self_test (ViSession vi, ViInt16* selfTestResult,  
                           ViChar[] selfTestMessage);
```

Purpose

Performs a self-test on the NI-RFSG device and returns the test results. This function performs a simple series of tests to ensure that the NI-RFSG device is powered up and responding.

This function does not affect external I/O connections or connections between devices. Complete functional testing and calibration are not performed by this function. The NI-RFSG device must be in the Configuration state before you call this function.

Supported Devices: NI 5610 (upconverter only mode), NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Output

Name	Type	Description
selfTestResult	ViInt16*	This parameter contains the value returned from the NI-RFSG device self test.

Self-Test Code	Description
0	Self test passed
1	Self test failed

selfTestMessage ViChar[] Returns the self-test response string from the NI-RFSG device. For an explanation of the string contents, refer to the **status** parameter of this function..

You must pass a ViChar array with at least 256 bytes.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_SelfCal

C Function Prototype

ViStatus niRFSG_SelfCal (ViSession vi);

Purpose

Performs an internal (self-) calibration on the device. If the calibration is successful, new calibration data and constants are stored in the onboard nonvolatile memory of the module.

Supported Devices: NI 5610 (upconverter only mode), NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_error_query

C Function Prototype

```
ViStatus niRFSG_error_query (ViSession vi, ViInt32 *Error_Code,  
                             ViChar Error_Message[]);
```

Purpose

Reads an error code and an error message from the instrument error queue.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Output

Name	Type	Description
Error_Code	ViInt32*	Returns the error code read from the instrument error queue.
Error_Message	ViChar[]	Returns the error message string read from the instrument error message queue. You must pass a ViChar array with at least 256 bytes.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_error_message

C Function Prototype

```
ViStatus niRFSG_error_message (ViSession vi, ViStatus errorCode,  
                               ViChar[] errorMessage);
```

Purpose

Converts an error code returned by an NI-RFSG function into a user-readable string.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	The ViSession handle that you obtain from niRFSG_init or niRFSG_InitWithOptions . The handle identifies a particular instrument session. You can pass VI_NULL for this parameter. Passing VI_NULL is useful when niRFSG_init or niRFSG_InitWithOptions fails. Default Value: VI_NULL
errorCode	ViStatus	Pass the status parameter that is returned from any NI-RFSG function. Default Value: 0 (VI_SUCCESS)

Output

Name	Type	Description
errorMessage	ViChar[]	Returns the user-readable message string that corresponds to the status code you specify. You must pass a ViChar array with at least 256 bytes to this parameter.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_GetError

C Function Prototype

```
ViStatus niRFSG_GetError (ViSession vi, ViStatus* errorCode,  
                          ViInt32 errorDescriptionBufferSize, ViChar errorDescription[]);
```

Purpose

Retrieves and then clears the IVI error information for the session or the current execution thread.



Note If the **bufferSize** parameter is 0, this function does not clear the error information. By passing 0 to the **bufferSize** parameter, you can determine the buffer size required to get the entire error description string. You can then call this function again with a sufficiently large buffer. If you specify a valid IVI session for the **vi** parameter, this function retrieves and clears the error information for the session. If you pass `VI_NULL` for the **vi** parameter, this function retrieves and clears the error information for the current execution thread. If the **vi** parameter is an invalid session, this function does nothing and returns an error. Normally, the error information describes the first error that occurred since the user last called this function or the [niRFSG_ClearError](#) function.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

bufferSize	ViInt32	Pass the number of bytes in the ViChar array you specify for the description parameter.
-------------------	---------	--

If the error description, including the terminating NULL byte, contains more bytes than you indicate in this parameter, the function copies **bufferSize** - 1 bytes into the buffer, places an ASCII NULL byte at the end of the buffer, and returns the buffer size you must pass to get the entire value. For example, if the value is 123456 and the buffer size is 4, the function places 123 into the buffer and returns 7. If you pass a negative number, the function copies the value to the buffer regardless of the number of bytes in the value. If you pass 0, you can pass VI_NULL for the **description** parameter.

Default Value: None

Output

Name	Type	Description
errorCode	ViStatus*	Returns the error code for the session or execution thread. If you pass 0 for the BufferSize parameter, you can pass VI_NULL for this parameter.
description	ViChar[]	Returns the error description for the IVI session or execution thread.

If there is no description, the function returns an empty string. The buffer must contain at least as many elements as the value you specify with the **bufferSize** parameter. If the error description, including the terminating NULL byte, contains more bytes than you indicate with the **bufferSize** parameter, the function copies **bufferSize** - 1 bytes into the buffer, places an ASCII NULL byte at the end of the buffer, and returns the buffer size you must pass to get the entire value. For example, if the value is 123456 and the buffer size is 4, the function places 123 into the buffer and returns 7. If you pass 0, you can pass VI_NULL for this parameter.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ClearError

C Function Prototype

```
ViStatus niRFSG_ClearError (ViSession vi);
```

Purpose

Clears the error information associated with the session. If you pass `VI_NULL` for the `vi` parameter, this function clears the error information for the current execution thread.



Note The [niRFSG_GetError](#) function clears the error information after it is retrieved. A call to the `niRFSG_ClearError` function is only necessary when a call to the `niRFSG_GetError` function is not used to retrieve error information.

The IVI Engine also maintains this error information separately for each thread. This feature is useful if you do not have a session handle to pass to the [niRFSG_ClearError](#) function or the [niRFSG_GetError](#) function, which occurs when a call to the [niRFSG_init](#) function or the [niRFSG_InitWithOptions](#) function fails.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_revision_query

C Function Prototype

```
ViStatus niRFSG_revision_query (ViSession vi,  
                               ViChar[] instrumentDriverRevision, ViChar[] firmwareRevision);
```

Purpose

Returns the revision numbers of the NI-RFSG driver and the instrument firmware.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None

Output

Name	Type	Description
instrumentDriverRevision	ViChar[]	Returns the value of the NIRFSG_ATTR_SPECIFIC_DRIVER_REVISION attribute in the form of a string. You must pass a ViChar array with at least 256 bytes.
firmwareRevision	ViChar[]	Returns the value of the NIRFSG_ATTR_INSTRUMENT_FIRMWARE_REVISION attribute in the form of a string. You must pass a ViChar array with at least 256 bytes.

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_QueryArbWaveformCapabilities

C Function Prototype

```
ViStatus niRFSG_QueryArbWaveformCapabilities (ViSession vi,  
                                              ViInt32* maxNumberWaveforms, ViInt32* waveformQuantum,  
                                              ViInt32* minWaveformSize, ViInt32* maxWaveformSize);
```

Purpose

Queries and returns the waveform capabilities of the NI-RFSG device. These capabilities are related to the current device configuration. The NI-RFSG device must be in the Configuration or the Generation state before calling this function.

Supported Devices: NI 5670/5671/5672

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session. Default Value: None

Output

Name	Type	Description
maxNumberWaveforms	ViInt32*	Returns the value of the NIRFSG_ATTR_ARB_MAX_NUMBER_WAVEFORMS attribute. This value is the maximum number of waveforms you can write.
waveformQuantum	ViInt32*	Returns the value of the NIRFSG_ATTR_ARB_WAVEFORM_QUANTUM attribute. If the waveform quantum is q , then the size of the waveform that you write should be a multiple of q . The units are expressed in samples.
minWaveformSize	ViInt32*	Returns the value of the NIRFSG_ATTR_ARB_WAVEFORM_SIZE_MIN attribute. The number of samples of the waveform that you write must be greater than or equal to this value.
maxWaveformSize	ViInt32*	Returns the value of the NIRFSG_ATTR_ARB_WAVEFORM_SIZE_MAX attribute. The number of samples of the waveform that you write must be less than or equal to this value.

Return Value

Name **Type** **Description**

status ViStatus Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_LockSession

C Function Prototype

```
ViStatus niRFSG_LockSession (ViSession vi, ViBoolean* callerHasLock);
```

Purpose

Obtains a multithread lock on the instrument session. Before doing so, this function waits until all other execution threads have released their locks on the instrument session.

Other threads might have obtained a lock on this session in the following ways:

- Your application already called the `niRFSG_LockSession` function.
- A call to NI-RFSG locked the session.

After the call to this function returns successfully, no other threads can access the instrument session until you call the [niRFSG_UnlockSession](#) function. Use the `niRFSG_LockSession` function and the `niRFSG_UnlockSession` function around a sequence of calls to NI-RFSG functions if you require that the NI-RFSG device retain its settings through the end of the sequence.

You can safely make nested calls to the `niRFSG_LockSession` function within the same thread. To completely unlock the session, balance each call to the `niRFSG_LockSession` function with a call to the `niRFSG_UnlockSession` function. If, however, you use the **callerHasLock** parameter in all calls to the `niRFSG_LockSession` function and the `niRFSG_UnlockSession` function within a function, the IVI Library locks the session only once within the function regardless of the number of calls you make to the `niRFSG_LockSession` function. Locking the session only once allows you to call `niRFSG_UnlockSession` just once at the end of the function.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Output

Name	Type	Description
callerHasLock	ViBoolean*	Keeps track of whether you obtain a lock and therefore need to unlock the session. Pass the address of a local ViBoolean variable. In the declaration of the local variable, initialize it to VI_FALSE. Pass the address of the same local variable to any other calls you make to the niRFSG_LockSession function or the niRFSG_UnlockSession function in the same function.

This parameter serves as a convenience. If you do not want to use this parameter, pass VI_NULL.

The parameter is an input/output parameter. The [niRFSG_LockSession](#) function and the [niRFSG_UnlockSession](#) each inspect the current value and take the following actions:

- If the value is VI_TRUE, the [niRFSG_LockSession](#) function does not lock the session again. If the value is VI_FALSE, the [niRFSG_LockSession](#) function obtains the lock and sets the value of the parameter to VI_TRUE.
- If the value is VI_FALSE, the [niRFSG_UnlockSession](#) function does not attempt to unlock the session. If the value is VI_TRUE, the [niRFSG_UnlockSession](#) function releases the lock and sets the value of the parameter to VI_FALSE.

Thus, you can call the [niRFSG_UnlockSession](#) function at the end of your function without worrying about whether you have the lock.

Example:

```
ViStatus TestFunc (ViSession vi, ViInt32 flags)
{
    ViStatus error = VI_SUCCESS;
    ViBoolean haveLock = VI_FALSE;

    if (flags & BIT_1)
    {
        viCheckErr( niRFSG_LockSession(vi, &haveLock));
        viCheckErr( TakeAction1(vi));
    }
    if (flags & BIT_2)
    {
```

```
viCheckErr( niRFSG_UnlockSession(vi, &haveLock));
viCheckErr( TakeAction2(vi));
viCheckErr( niRFSG_LockSession(vi, &haveLock);
}
if (flags & BIT_3)
viCheckErr( TakeAction3(vi));
}
```

Error:

```
/*
At this point, you cannot really be sure that you have the lock.
Fortunately, the haveLock variable takes care of that for you.
*/
niRFSG_UnlockSession(vi, &haveLock);
return error;
}
```

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_UnlockSession

C Function Prototype

```
ViStatus niRFSG_UnlockSession (ViSession vi, ViBoolean* callerHasLock);
```

Purpose

Releases a lock obtained on an NI-RFSG device session by calling the [niRFSG_LockSession](#) function.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

Output

Name	Type	Description
callerHasLock	ViBoolean*	Keeps track of whether you obtain a lock and therefore need to unlock the session. Pass the address of a local ViBoolean variable. In the declaration of the local variable, initialize it to VI_FALSE. Pass the address of the same local variable to any other calls you make to the niRFSG_LockSession function or the niRFSG_UnlockSession function in the same function.

This parameter serves as a convenience. If you do not want to use this parameter, pass VI_NULL.

The parameter is an input/output parameter. The [niRFSG_LockSession](#) function and the [niRFSG_UnlockSession](#) each inspect the current value and take the following actions:

- If the value is VI_TRUE, the [niRFSG_LockSession](#) function does not lock the session again. If the value is VI_FALSE, the [niRFSG_LockSession](#) function obtains the lock and sets the value of the parameter to VI_TRUE.
- If the value is VI_FALSE, the [niRFSG_UnlockSession](#) function does not attempt to unlock the session. If the value is VI_TRUE, the [niRFSG_UnlockSession](#) function releases the lock and sets the value of the parameter to VI_FALSE.

Thus, you can call the [niRFSG_UnlockSession](#) function at the end of your function without worrying about whether you have the lock.

Example:

```
ViStatus TestFunc (ViSession vi, ViInt32 flags)
{
    ViStatus error = VI_SUCCESS;
    ViBoolean haveLock = VI_FALSE;

    if (flags & BIT_1)
    {
        viCheckErr( niRFSG_LockSession(vi, &haveLock));
        viCheckErr( TakeAction1(vi));
    }
    if (flags & BIT_2)
    {
```



```
viCheckErr( niRFSG_UnlockSession(vi, &haveLock));
viCheckErr( TakeAction2(vi));
viCheckErr( niRFSG_LockSession(vi, &haveLock);
}
if (flags & BIT_3)
viCheckErr( TakeAction3(vi));
}
```

Error:

```
/*
At this point, you cannot really be sure that you have the lock.
Fortunately, the haveLock variable takes care of that for you.
*/
niRFSG_UnlockSession(vi, &haveLock);
return error;
}
```

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

niRFSG_ConfigureIQEnabled [OBSOLETE]

C Function Prototype

ViStatus niRFSG_ConfigureIQEnabled (ViSession vi, ViBoolean enabled);

Purpose

Configures the NI-RFSG device to apply IQ (vector) modulation to the RF output signal. IQ modulation must be enabled in order to generate any arbitrary (non-sine) waveform; if IQ modulation is disabled, a sine tone is always generated, regardless if an arbitrary waveform is written. The NI-RFSG device must be in the Configuration state before calling this function.



Note This property is obsolete. Use the [NIRFSG_ATTR_GENERATION_MODE](#) property to enable IQ modulation instead.

Upon device initialization, or calling the [niRFSG_reset](#) function or the [niRFSG_ResetDevice](#) function, IQ modulation is disabled.

Parameters

Input

Name	Type	Description
vi	ViSession	Identifies your instrument session. The ViSession handle is obtained from the niRFSG_init function or the niRFSG_InitWithOptions function and identifies a particular instrument session.

Default Value: None

enabled ViBoolean NI-RFSG sets the [NIRFSG_ATTR_IQ_ENABLED](#) attribute to this value.

Defined Values:

VI_TRUE	Enables IQ (vector) modulation (arbitrary waveform generation)
VI_FALSE	Disables IQ (vector) modulation (sine wave generation)

Default Value: VI_FALSE

Return Value

Name	Type	Description
------	------	-------------

status	ViStatus	Returns the status code of this operation. The status code either indicates success or describes an error or warning condition. Examine the status code from each call to an instrument driver function to determine if an error occurred.
---------------	----------	--

To obtain a text description of the status code, call the [niRFSG_error_message](#) function. To obtain additional information about the error condition, call the [niRFSG_GetError](#) function. To clear the error information from the driver, call the [niRFSG_ClearError](#) function.

The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

NI-RFSG Attributes

Expand this book to view the NI-RFSG attributes.

NIRFSG_ATTR_FREQUENCY

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	niRFSG_ConfigureRF

Description

Specifies the frequency of the generated RF signal. For arbitrary waveform generation, this attribute specifies the center frequency of the signal, expressed in Hertz. To set this attribute, the NI-RFSG device must be in the Configuration state (NI 5670/5671/5672 device) or Generation state (NI 5650/5651/5652 device).

Defined Values:

Refer to the [specifications document](#) for your device allowable frequency settings.

Default Value: 100 MHz

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_POWER_LEVEL

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	niRFSG_ConfigureRF

Description

Specifies the power level of the generated RF signal. The [NIRFSG_ATTR_POWER_LEVEL_TYPE](#) attribute specifies whether the power level is the average power of the signal or the peak power of the signal. By default, NIRFSG_ATTR_POWER_LEVEL_TYPE specifies the average power of the signal. Refer to this attribute for more information about average power versus peak power.

Defined Values:

Refer to the [specifications document](#) for your device for supported power level settings.

Default Value: -145 dBm

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_POWER_LEVEL_TYPE

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the manner in which the driver interprets the value of the [NIRFSG_ATTR_POWER_LEVEL](#) attribute.

Defined Values:

NIRFSG_VAL_AVERAGE_POWER	Specifies that the power level type is average power. Average power indicates the desired power averaged in time. The driver maximizes the dynamic range by scaling the IQ waveform. If you write more than one waveform, NI-RFSG scales each waveform without preserving the power level ratio between the waveforms.
NIRFSG_VAL_PEAK_POWER	Specifies that the power level type is peak power. Peak power indicates the maximum power level of the RF signal averaged over one period of the RF carrier signal frequency (the peak envelope power). This setting requires that the magnitude of the IQ waveform must always be less than or equal to one. When using the peak power level type, the power level of the RF signal matches the specified power level at moments when the magnitude of the IQ waveform equals one. If you write more than one waveform, the relative scaling between waveforms is preserved.



Note If this attribute is set to NIRFSG_VAL_AVERAGE_POWER while in Script generation mode, the driver scales each waveform so that all waveforms have the same average power. The average power level of each waveform matches the value set with the NIRFSG_ATTR_POWER_LEVEL attribute. You can disable this scaling operation by setting the NIRFSG_ATTR_POWER_LEVEL_TYPE attribute to NIRFSG_VAL_PEAK_POWER.



Note This property is only valid if set to Peak Power while in Script mode for the NI 5672.

Converting from Average Power to Peak Power

Typically, this attribute is set to NIRFSG_VAL_AVERAGE_POWER. However, some instrument modes require this attribute to be set to NIRFSG_VAL_PEAK_POWER. Use the following equations to calculate the equivalent peak power given the desired average power for your waveform.

$$\text{AverageMagnitude}^2 = \frac{1}{N} \sum_{i=0}^{n-1} (I_i^2 + Q_i^2)$$

$$\text{PeakPower} = \text{AveragePower} + 10 \log \left(\frac{1}{\text{AverageMagnitude}^2} \right)$$

Default Value: NIRFSG_VAL_AVERAGE_POWER

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_PEAK_ENVELOPE_POWER

Data type	Access	Coercion	High Level Functions
ViReal64	RO	None	None

Description

Specifies the maximum instantaneous power, expressed in dBm, of the current RF output signal.



Note This attribute is ignored when the [NIRFSG_ATTR_POWER_LEVEL_TYPE](#) attribute is set to NIRFSG_VAL_PEAK_POWER.

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_LOCAL_OSCILLATOR_OUT_0_EN

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Specifies whether the local oscillator signal is present at the front panel LOCAL OSC OUT 0 connector. The frequency of this signal is approximately 3.2 GHz plus the RF frequency specified using the [NIRFSG_ATTR_FREQUENCY](#) attribute.

Defined Values:

VI_TRUE	The local oscillator signal is present at the front panel LOCAL OSC OUT 0 connector.
VI_FALSE	The LOCAL OSC OUT 0 connector signal is terminated.

Default Value: VI_FALSE

Supported Devices: NI 5610 (upconverter only mode), NI 5670/5671/5672

NIRFSG_ATTR_OUTPUT_ENABLED

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	niRFSG_ConfigureOutputEnabled

Description

Enables or disables signal generation. This attribute can be set in any software state, and it does not change the current state. Setting the NIRFSG_ATTR_OUTPUT_ENABLED attribute to VI_FALSE while in the Generation state stops signal output, although generation continues internally.

Defined Values:

VI_TRUE	Enables signal output.
VI_FALSE	Disables signal output.

Default Value: VI_TRUE

Supported Devices: NI 5610 (upconverter only mode), NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_FREQUENCY_TOLERANCE

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Specifies the maximum frequency error allowed during the software upconversion process. NI-RFSG may introduce a frequency error up to the specified amount to optimize computational speed and onboard memory usage while upconverting [phase-continuous](#) signals. If the [NIRFSG_ATTR_PHASE_CONTINUITY_ENABLED](#) attribute is set to `NIRFSG_VAL_DISABLE`, the `NIRFSG_ATTR_FREQUENCY_TOLERANCE` attribute is ignored, and the driver does not introduce a frequency error. To set the `NIRFSG_ATTR_FREQUENCY_TOLERANCE` attribute, the NI-RFSG device must be in the Configuration state.

- **NI 5671** — This attribute applies only when the [NIRFSG_ATTR_IQ_RATE](#) attribute is set to a value > 8.33MS/s.

Valid Values: 1 Hz to 10 MHz

Default Value: 50 Hz

Supported Devices: NI 5670/5671

NIRFSG_ATTR_DEVICE_TEMPERATURE

Data type	Access	Coercion	High Level Functions
ViReal64	RO	None	None

Description

Indicates the signal generator device temperature in degrees Celsius.

Serial signals between the sensor and the system control unit could modulate the signal being generated, causing phase spurs. After the device is thoroughly warmed up, its temperature varies only slightly (less than 1 °C) and slowly, so it is not necessary to constantly poll this temperature sensor. For these reasons, NI-RFSG reads the temperature sensor not more than once every minute.

Queries to this attribute return the previous sensor reading until at least one minute has passed since the previous sensor reading occurred. Refer to the [thermal management](#) section for more information about device temperature.

Supported Devices: NI 5610 (upconverter only mode), NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_ATTENUATOR_HOLD_ENABLED

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Enables or disables attenuator hold. While this property is set to VI_TRUE, changing the power level causes NI-RFSG to scale the digital data sent to the AWG rather than change the attenuators. Changing power levels in this manner allows the device to increase/decrease the power level in more accurate increments but may affect signal-to-noise ratios (noise density).

The frequency cannot be changed while attenuator hold is enabled.

Setting this attribute to VI_TRUE limits the power levels that can be attained. With attenuator hold enabled, the power level must satisfy the following conditions:

- Power level \leq the maximum power level set with the [NIRFSG_ATTR_ATTENUATOR_HOLD_MAX_POWER](#) attribute
- Power level \geq (the maximum power level set with the [NIRFSG_ATTR_ATTENUATOR_HOLD_MAX_POWER](#) attribute -70 dB)
- Power level ≥ -145 dBm

To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

VI_TRUE	Attenuator hold is enabled.
VI_FALSE	Attenuator hold is disabled.

Default Value: VI_FALSE

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_ATTENUATOR_HOLD_MAX_POW

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Specifies the maximum power level of the RF output signal when the [NIRFSG_ATTR_ATTENUATOR_HOLD_ENABLED](#) attribute is set to VI_TRUE. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

Refer to the [specifications document](#) for your device for allowable maximum power levels.

Default Value: 17 dBm

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_IF_CARRIER_FREQUENCY

Data type	Access	Coercion	High Level Functions
ViReal64	RO	None	None

Description

Indicates the IF carrier frequency generated by the NI 5421/5441/5442 AWG module, expressed in Hertz. The specified IF carrier frequency is related to the RF output as follows:

$$\text{RF Frequency (MHz)} = \text{Upconverter Center Frequency} + \text{IF Carrier Frequency} - 25 \text{ MHz}$$

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_IF_POWER

Data type	Access	Coercion	High Level Functions
ViReal64	RO	None	None

Description

Indicates the output power from the NI 5421/5441/5442 AWG module, expressed in dBm. If an arbitrary waveform is being generated, this attribute specifies either the average power or the peak power of the signal, depending on the setting of the [NIRFSG_ATTR_POWER_LEVEL_TYPE](#) attribute.

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_DIGITAL_PATTERN

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Enables or disables digital pattern on the NI 5421/5441/5442 AWG module. This attribute must be set to VI_TRUE to enable signal routing to and from the Digital Data & Control connector. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

VI_TRUE	Signal routing enabled.
VI_FALSE	Signal routing disabled.

Default Value: VI_FALSE

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_UPCONVERTER_CENTER_FREQ

Data type	Access	Coercion	High Level Functions
ViReal64	5670/5671/5672: RO 5610: R/W	None	None

Description

Indicates the center frequency for the upconverted RF signal, expressed in Hertz.

Supported Devices: NI 5610 (upconverter only mode), NI 5670/5671/5672

NIRFSG_ATTR_UPCONVERTER_GAIN

Data type	Access	Coercion	High Level Functions
ViReal64	5670/5671/5672: RO 5610: R/W	None	None

Description

Indicates the gain that the upconverter is applying to the signal, expressed in dB.

Supported Devices: NI 5610 (upconverter only mode), NI 5670/5671/5672

NIRFSG_ATTR_UPCONVERTER_LOOP_BANDWI

Specific Attribute

Data type	Access	Applies to	Coercion	High Level Functions
ViInt32	R/W	N/A	None	None

Description

Configures the loop bandwidth of the upconverter tuning PLLs. To set this attribute, the NI-RFSG device must be in the Configuration state.



Note This property is ignored for signal bandwidths greater than or equal to 10 MHz.

Defined Values:

NIRFSG_VAL_LOW	Specifies that the upconverter module uses a low loop bandwidth.
NIRFSG_VAL_MEDIUM	Specifies that the upconverter module uses a medium loop bandwidth.
NIRFSG_VAL_HIGH	Specifies that the upconverter module uses a high loop bandwidth.

Default Value: NIRFSG_VAL_HIGH

Supported Devices: NI 5610 (upconverter only mode), NI 5670/5671/5672

NIRFSG_ATTR_UPCONVERTER_CENTER_FREQ

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Specifies the frequency by which the IF signal is moved. The NI 5610 device default settings specify a multiple of 5 MHz for signal bandwidths <10 MHz and a multiple of 1 MHz for signal bandwidths >10 MHz. This attribute allows you to set any other desired increment.

This attribute and the

[NIRFSG_ATTR_UPCONVERTER_CENTER_FREQUENCY_INCREMENT_A](#)

attribute can be used to modify how NI-RFSG selects upconverter center frequency values.

Supported Devices: NI 5672

NIRFSG_ATTR_UPCONVERTER_CENTER_FREQ

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

This attribute and the [NIRFSG_ATTR_UPCONVERTER_CENTER_FREQUENCY_INCREMENT](#) attribute can be used to modify how NI-RFSG selects upconverter center frequency values.

The *anchor* is the reference point. All upconverter center frequencies are an integer multiple of the upconverter center frequency increment away from the upconverter center frequency increment anchor, as expressed in the following equation:

$$\text{Upconverter Center Frequency} = (k \times \text{Upconverter Center Frequency Increment}) + \text{Upconverter Center Frequency Increment Anchor}$$

where

k is any positive or negative integer.

Supported Devices: NI 5672

NIRFSG_ATTR_ALLOW_OUT_OF_SPECIFICATIO

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Allows you to set the frequency and power values beyond the limits of the NI-RFSG device specifications. This capability allows a wider frequency and power range, but accuracy cannot be guaranteed, and results may vary by unit. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

NIRFSG_VAL_ENABLE	Frequency and power settings can be specified.
NIRFSG_VAL_DISABLED	Frequency and power settings cannot be specified.

Default Value: NIRFSG_VAL_DISABLED

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_GENERATION_MODE

Data type	Access	Applies to	Coercion	High Level Functions
ViInt32	R/W	N/A	None	niRFSG_ConfigureGenerationMode

Description

Use this attribute to specify whether to generate a continuous wave (CW) signal, a single arbitrary waveform, or a script, upon calling the [niRFSG_Initiate](#) function. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

NIRFSG_VAL_CW	Configures the RF signal generator to generate a CW signal.
NIRFSG_VAL_ARB_WAVEFORM	Configures the RF signal generator to generate the arbitrary waveform specified by the NIRFSG_ATTR_ARB_SELECTED_WAVEFORM attribute.
NIRFSG_VAL_SCRIPT	Configures the RF signal generator to generate arbitrary waveforms as directed by the NIRFSG_ATTR_SELECTED_SCRIPT attribute. Refer to Scripting Instructions for more information about scripting.

Default Value: NIRFSG_VAL_CW

Supported Devices: NI 5650/5651/5652 (CW support only), NI 5670/5671/5672

NIRFSG_ATTR_SIGNAL_BANDWIDTH

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Specifies the bandwidth of the arbitrary signal. This value must be less than or equal to $(0.8 \times \text{IQ rate})$.

NI-RFSG defines *signal bandwidth* as twice the maximum baseband signal deviation from 0 Hz. Usually, the baseband signal center frequency is 0 Hz. In such cases, the signal bandwidth is simply the baseband signal's minimum frequency subtracted from its maximum frequency, or $f_{\text{max}} - f_{\text{min}}$. NI-RFSG uses this value to optimally configure the center frequency of the upconverter to help minimize phase noise. The generated signal will not be filtered to achieve the set bandwidth. However, specifying a bandwidth smaller than the actual bandwidth of the signal could potentially result in spectral distortion.

This attribute only applies when the [NIRFSG_ATTR_GENERATION_MODE](#) attribute is set to `NIRFSG_VAL_ARB_WAVEFORM` or `NIRFSG_VAL_SCRIPT`. To set this attribute, the NI-RFSG device must be in the Configuration state.



Note Based on your signal bandwidth, NI-RFSG decides whether to configure the upconverter center frequency in increments of 1 or 5 MHz. Failure to configure this attribute may result in the signal being placed out of the upconverter passband.

Valid Values:

0 Hz to 20 MHz

Default Value: 100 Hz

Supported Devices: NI 5610, NI 5670/5671/5672

NIRFSG_ATTR_ARB_SELECTED_WAVEFORM

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	niRFSG_SelectArbWaveform

Description

Specifies the selected waveform from the pool of available waveforms. If only one waveform is available, pass an empty string. To set this attribute, the NI-RFSG device must be in the Configuration state.

Valid Values:

Any null terminated string less than or equal to 256 characters long. Names are case-sensitive.

Default Value: "" (empty string)

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_ARB_MAX_NUMBER_WAVEFORM

Data type	Access	Coercion	High Level Functions
ViInt32	RO	None	niRFSG_QueryArbWaveformCapabilities

Description

Specifies the maximum number of waveforms the NI-RFSG device can hold in memory.

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_ARB_WAVEFORM_SIZE_MIN

Data type	Access	Coercion	High Level Functions
ViInt32	RO	None	niRFSG_QueryArbWaveformCapabilities

Description

Specifies the size (in samples) of the smallest allowed waveform.

Supported Devices: NI 5670/5671/5672

- **NI 5671** — The value of this attribute depends on the [NIRFSG_ATTR_IQ_RATE](#). Set the NIRFSG_ATTR_IQ_RATE before reading this attribute.

NIRFSG_ATTR_ARB_WAVEFORM_SIZE_MAX

Data type	Access	Coercion	High Level Functions
ViInt32	RO	None	niRFSG_QueryArbWaveformCapabilities

Description

Specifies the size (in samples) of the largest waveform that is allowed. To check this attribute, the NI-RFSG device must be in the Configuration state.

Supported Devices: NI 5670/5671/5672

- **NI 5671** — The value of this attribute depends on the setting of the [NIRFSG_ATTR_IQ_RATE](#) attribute. Set the [NIRFSG_ATTR_IQ_RATE](#) attribute before reading this attribute.

NIRFSG_ATTR_ARB_WAVEFORM_QUANTUM

Data type	Access	Coercion	High Level Functions
ViInt32	RO	None	niRFSG_QueryArbWaveformCapabilities

Description

Obtains the quantum value that NI-RFSG uses. The waveform length (the number of samples) must be a multiple of this quantum. The other restrictions on the length of the waveform are the [minimum](#) and [maximum](#) arbitrary waveform sizes.

Supported Devices: NI 5670/5671/5672

- **NI 5671** — The value of this attribute depends on the [NIRFSG_ATTR_IQ_RATE](#). Set the NIRFSG_ATTR_IQ_RATE before reading this attribute.

NIRFSG_ATTR_DIGITAL_IF_EQUALIZATION_ENA

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

When enabled, NI-RFSG equalizes the waveform data to correct for variations in the IF response of the NI-RFSG device. Enabling the digital IF equalization improves the modulation error rates (MER) and error vector magnitude (EVM) for signals with large bandwidths (> 500 kHz), but it increases [tuning times](#). To set this attribute, the NI-RFSG device must be in the Configuration state. Refer to the [Software Equalizer](#) section for more information about digital IF equalization.

This attribute only applies when the [NIRFSG_ATTR_GENERATION_MODE](#) attribute is set to NIRFSG_VAL_ARB_WAVEFORM or NIRFSG_VAL_SCRIPT.

Defined Values:

NIRFSG_VAL_ENABLE	NI-RFSG equalizes the waveform data to correct for variations in the IF response of the NI-RFSG device.
NIRFSG_VAL_DISABLE	NI-RFSG does not equalize the waveform data to correct for variations in the IF response of the NI-RFSG device.

Default Values:

- NI 5670/5671–NIRFSG_VAL_DISABLED
- NI 5672–NIRFSG_VAL_ENABLED

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_IQ_RATE

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

This attribute specifies the IQ rate of the arbitrary waveform in samples per second (S/s). The NI-RFSG driver automatically coerces the specified IQ rate up to the next valid IQ rate. To set this attribute, the NI-RFSG device must be in the Configuration state.



Note NI-RFSG internally uses a FIR filter with flat response up to $(0.4 \times \text{IQ rate})$. Given a desired signal with the maximum frequency content f , sample the signal at an IQ rate greater than or equal to $(f/0.4)$.

Defined Values:

Device	Value	Available Rates
NI 5670	50E6	50 MS/s
	100E6	100 MS/s
NI 5671	50E6	50 MS/s
	100E6	100 MS/s
	—	$(100 \text{ MS/s})/n$, where n is divisible by 2 between 12—512, and divisible by 4 between 512—1024 ($n = 12, 14, 16, \dots, 512, 516, 520, \dots, 1024$). Setting the IQ Rate to one of these value enables the DUC .
NI 5672	—	Supports IQ rates up to 100 MS/s. You should read this value back after setting it to see what the actual IQ rate is.



Note The IQ rate will be coerced to what rates your specific device can achieve.

Default Value: 100E6 (100 MS/s)

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_PHASE_CONTINUITY_ENABLED

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies whether the driver maintains [phase continuity](#) in the arbitrary waveforms. When this attribute is set to `NIRFSG_VAL_ENABLE`, NI-RFSG may increase the waveform size. When this attribute is set to `NIRFSG_VAL_DISABLE`, the [NIRFSG_ATTR_FREQUENCY_TOLERANCE](#) attribute specifies the maximum allowable frequency error that can be introduced when keeping the signal phase-continuous. To set the `NIRFSG_ATTR_PHASE_CONTINUITY_ENABLED` attribute, the NI-RFSG device must be in the Configuration state.

`NIRFSG_ATTR_PHASE_CONTINUITY_ENABLED` only applies when the [NIRFSG_ATTR_GENERATION_MODE](#) attribute is set to `NIRFSG_VAL_ARB_WAVEFORM` or `NIRFSG_VAL_SCRIPT`.

- **NI 5671** — When using the NI 5671 with IQ rates ≤ 8.33 MS/s, an input phase-continuous signal is always phase-continuous upon output, and this attribute has no effect.
- **NI 5672** — Phase continuity is *always* enabled on this device.

Defined Values:

Phase Continuity Enabled Attribute Settings with IQ Rates > 8.33 MS/s.		
Attribute Setting	Arb Mode	Script Mode
<code>NIRFSG_VAL_AUTO</code>	The arbitrary waveform may be repeated to ensure phase continuity after upconversion. This setting could cause waveform size to increase.	Warning condition — NI-RFSG cannot guarantee a phase-continuous output signal in script mode. Phase continuity is automatically disabled in script mode and the arbitrary waveform is played back without regard to any possible phase discontinuities introduced by upconversion.
<code>NIRFSG_VAL_ENABLE</code>	The arbitrary waveform may be repeated to ensure phase continuity after upconversion. Enabling this attribute could cause waveform size to increase.	Error condition — NI-RFSG cannot guarantee a phase-continuous output signal in script mode.
<code>NIRFSG_VAL_DISABLED</code>	The arbitrary waveform is played back without regard to any possible phase discontinuities introduced by upconversion. The time duration of the original waveform is maintained.	The arbitrary waveform is played back without regard to any possible phase discontinuities introduced by upconversion. The time duration of the original waveform is maintained.

Default Value: NIRFSG_VAL_AUTO

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_IQ_SWAP_ENABLED

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Enables or disables the inverse phase rotation of the IQ signal by swapping the I and Q inputs. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

VI_TRUE	NI-RFSG device applies noninverse phase rotation of the IQ signal.
VI_FALSE	NI-RFSG device applies inverse phase rotation of the IQ signal.

Default Value: VI_FALSE

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_DIRECT_DOWNLOAD

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies whether the [niRFSG_WriteArbWaveform](#) function writes waveforms immediately to the device or instead copies the waveform to host memory for download later. NI-RFSG reads and validates this attribute when an arbitrary waveform is first allocated.

- **NI 5672**- Direct download is *always* enabled.
- **NI 5671**- To increase performance when using large waveforms, enable direct download. To maximize reconfigurability, disable direct download.

Perform the following steps to enable direct download:

- Set the IQ rate to ≤ 8.33 MS/s with the [NIRFSG_ATTR_IQ_RATE](#) attribute.
 - Set the [NIRFSG_ATTR_POWER_LEVEL_TYPE](#) attribute to NIRFSG_VAL_PEAK_POWER.
 - Disable the [NIRFSG_ATTR_IQ_SWAP_ENABLED](#) attribute.
 - Disable the [NIRFSG_ATTR_DIGITAL_IF_EQUALIZATIO](#) attribute.
- **NI 5670**- Direct download is *always* disabled.

Defined Values:

NIRFSG_VAL_ENABLE	Direct download is enabled.
-------------------	-----------------------------

NIRFSG_VAL_DISABLE	Direct download is disabled.
--------------------	------------------------------

Default Value: NIRFSG_VAL_DISABLED

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_ARB_WAVEFORM_SOFTWARE_S

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Specifies how much to scale the data by before writing it with the [niRFSG_WriteArbWaveform](#) function.

Default Value: 1.0

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_STREAMING_ENABLED

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Enables and disables continuous streaming of waveform data.

Defined Values:

NIRFSG_VAL_ENABLE	Streaming is enabled.
NIRFSG_VAL_DISABLED	Streaming is disabled.

Default Value:NIRFSG_VAL_DISABLED

Supported Devices: NI 5672

NIRFSG_ATTR_STREAMING_WAVEFORM_NAME

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	None

Description

Specifies the name of the waveform used to continually stream data during generation.

Default Value: "" (empty string)

Supported Devices: NI 5672

NIRFSG_ATTR_STREAMING_SPACE_AVAILABLE

Data type	Access	Coercion	High Level Functions
ViInt64	RO	None	None

Description

Indicates the space available (in samples) in the streaming waveform for writing new data. This attribute also reports the available space in the waveform (allocated size minus previously written data). During generation, this available space may be in multiple locations with, for example, part of the available space at the end of the streaming waveform and the rest at the beginning. In this situation, writing a block of waveform data the size of the total space available in the streaming waveform causes NI-RFSG to return an error, as NI-RFSG will not wrap the data from the end of the waveform to the beginning and cannot write data past the end of the waveform buffer.

To avoid writing data past the end of the waveform, write new data to the waveform in a fixed size that is an integer divisor of the total size of the streaming waveform.

To set this attribute, the NI-RFSG device must be in the Generation state.

Supported Devices: NI 5672

NIRFSG_ATTR_DATA_TRANSFER_BLOCK_SIZE

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

The number of samples to download to onboard memory at one time. This attribute is useful when the total data to be transferred to onboard memory is large.

Default Value: 1M

Supported Devices: NI 5672

NIRFSG_ATTR_DIRECT_DMA_ENABLED

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Enables the device for direct DMA writes. When enabled, the [niRFSG_WriteArbWaveform](#) function writes data residing on the direct DMA device (specified in the data address in the [NIRFSG_ATTR_DIRECT_DMA_WINDOW_ADDRESS](#) attribute) to the NI-RFSG device onboard memory.

Defined Values:

VI_TRUE	Data is written to the device onboard memory.
VI_FALSE	Data is not written to the device onboard memory.

Default Value: VI_FALSE

Supported Devices: NI 5672

NIRFSG_ATTR_DIRECT_DMA_WINDOW_ADDRESS

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the window address (beginning of window) of the waveform data source. This window address is specified by your direct DMA-compatible data source.

Default Value: 0

Supported Devices: NI 5672

NIRFSG_ATTR_DIRECT_DMA_WINDOW_SIZE

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the size of the memory window (in bytes, not samples) provided by your direct DMA-compatible data source.

Default Value: 0

Supported Devices: NI 5672

NIRFSG_ATTR_DUC_PRE_FILTER_GAIN

Specific Attribute

Data type	Access	Applies to	Coercion	High Level Functions
ViReal64	R/W	N/A	None	None

Description

Specifies the DUC prefilter gain. Reduce this value to prevent overflow in the DUC interpolation filters. Other gains on the NI-RFSG device are automatically adjusted to compensate for nonunity DUC prefilter gain. This attribute only applies to DUC-equipped RF signal generators. To set this attribute, the NI-RFSG device must be in the Configuration state. The units are expressed in dB.

The following table lists the behavior of this attribute on supported devices:

Supported Device	Attribute Behavior
NI 5671	This attribute only applies when the NIRFSG_ATTR_IQ_RATE attribute is set to a value ≤ 8.33 MS/s.
NI 5672	This attribute is always applicable.

NIRFSG_ATTR_DUC_FIR_FILTER_TYPE

Data type	Access	Applies to	Coercion	High Level Functions
ViInt32	R/W	N/A	None	None

Description

Pulse-shaping filter type for the FIR filter.

Defined Values:

NIRFSG_VAL_DUC_NONE	No filter type is applied.
NIRFSG_VAL_DUC_ROOT_RAISED_COSINE	Applies a root-raised cosine filter to the data with the alpha specified with the NIRFSG_ATTR_DUC_FIR_FILTER_ROOT_RAISED_COSIN attribute.
NIRFSG_VAL_DUC_RAISED_COSINE	Applies a raised cosine filter to the data with the alpha value specified with the NIRFSG_ATTR_DUC_FIR_FILTER_RAISED_COSINE_ALP attribute.

Default Value: NIRFSG_VAL_DUC_NONE

Supported Devices: NI 5671/5672

NIRFSG_ATTR_DUC_FIR_FILTER_ROOT_RAISEL

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Alpha value to use when calculating the pulse-shaping FIR filter coefficients. Only used when the [NIRFSG_ATTR_DUC_FIR_FILTER_TYPE](#) attribute is set to NIRFSG_VAL_DUC_ROOT_RAISED_COSINE.

Supported Devices: NI 5671/5672

NIRFSG_ATTR_DUC_FIR_FILTER_RAISED_COSI

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Alpha value to use when calculating the pulse-shaping FIR filter coefficients. This attribute is only used when the [NIRFSG_ATTR_DUC_FIR_FILTER_TYPE](#) attribute is set to NIRFSG_VAL_DUC_RAISED_COSINE.

Supported Devices: NI 5671/5672

NIRFSG_ATTR_ANALOG_MODULATION_TYPE

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the analog modulation format to use.

Defined Values:

NIRFSG_VAL_NONE	Disables analog modulation.
NIRFSG_VAL_FM	Specifies that the analog modulation type is FM.

Default Value: NIRFSG_VAL_NONE

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_ANALOG_MODULATION_WAVEF

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the type of waveform to use as the message signal for analog modulation.

Defined Values:

NIRFSG_VAL_SINE	Specifies that the analog modulation waveform type is sine.
NIRFSG_VAL_SQUARE	Specifies that the analog modulation waveform type is square.
NIRFSG_VAL_TRIANGLE	Specifies that the analog modulation waveform type is triangle.

Default Value: NIRFSG_VAL_SINE

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_ANALOG_MODULATION_WAVEF

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Specifies the frequency of the waveform to use as the message signal in analog modulation.

Default Value: 1 kHz

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_ANALOG_MODULATION_FM_DEV

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Specifies the [deviation](#) to use in frequency modulation.

Default Value: 1 kHz

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_DIGITAL_MODULATION_TYPE

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the digital modulation format to use.

Defined Values:

NIRFSG_VAL_NONE	Disables digital modulation.
NIRFSG_VAL_FSK	Specifies that the digital modulation type is frequency-shift keying (FSK).
NIRFSG_VAL_OOK	Specifies that the digital modulation type is on-off keying (OOK).

Default Value: NIRFSG_VAL_NONE

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_DIGITAL_MODULATION_SYMBOL

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Specifies the symbol rate of the bit stream for FSK modulation.

Default Value: 1 kHz

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_DIGITAL_MODULATION_WAVEFC

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the type of waveform to use as the message signal in digital modulation.

Defined Values:

NIRFSG_VAL_PRBS	Specifies that the digital modulation waveform type is pseudorandom bit sequence (PRBS).
NIRFSG_VAL_USER_DEFINED	Specifies that the digital modulation waveform type is user defined. To specify the user-defined waveform, call the niRFSG_ConfigureDigitalModulationUserDefinedWaveform function.

Default Value: NIRFSG_VAL_PRBS

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_DIGITAL_MODULATION_PRBS_0

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the order of the [PRBS](#) internally generated by hardware and used as the message signal in digital modulation.

Default Value: 18

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_DIGITAL_MODULATION_PRBS_S

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the seed of the internally generated [PRBS](#).

Default Value: 1

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_DIGITAL_MODULATION_FSK_DE

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	None

Description

Specifies the deviation to use in FSK modulation.

Default Value: 1 kHz

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_REF_CLOCK_SOURCE

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	niRFSG_ConfigureRefClock

Description

Specifies the reference clock source. To set this attribute, the NI-RFSG device must be in the Configuration state. Only certain combinations of this attribute and the [NIRFSG_ATTR_PXI_CHASSIS_CLK10_SOURCE](#) attribute are valid, as shown in the following table.

NI 5670/5671 devices also allow you to drive the PXI 10 MHz backplane clock on PXI chassis *only* using the `NIRFSG_ATTR_PXI_CHASSIS_CLK10_SOURCE` attribute.

Defined Values:

Value	Description
<code>NIRFSG_VAL_ONBOARD_CLK_STR</code>	Use the onboard reference clock as the clock source.
<code>NIRFSG_VAL_REF_IN_STR</code>	Use the clock signal present at the front panel REF IN connector as the clock source.
<code>NIRFSG_VAL_PXI_CLK10_STR</code>	Use the <code>PXI_CLK10</code> signal, which is present on the PXI backplane, as the clock source.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_REF_CLOCK_RATE

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	niRFSG_ConfigureRefClock

Description

Specifies the rate of the reference clock, expressed in Hertz. To set this attribute, the NI-RFSG device must be in the Configuration state.

Default Value: 10 MHz

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_PXI_CHASSIS_CLK10_SOURCE

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	niRFSG_ConfigurePXIChassisClk10

Description

Specifies the clock source for driving the PXI 10 MHz backplane reference clock. This option can only be configured if the upconverter is in Slot 2 of a PXI chassis. To set this attribute, the NI-RFSG device must be in the Configuration state.

Valid Timing Configurations:

NIRFSG_ATTR_PXI_CHASSIS_CLK10_SOURCE Setting	<u>NIRFSG_ATTR_REF_CLOCK_SOURCE</u> Setting
NIRFSG_VAL_NONE_STR, NIRFSG_VAL_ONBOARD_CLK_STR (these settings are not valid on the NI 5672)	NIRFSG_VAL_ONBOARD_CLK_STR (this setting is not valid on the NI 5672)
NIRFSG_VAL_NONE_STR, NIRFSG_VAL_REF_IN_STR	NIRFSG_VAL_REF_IN_STR
NIRFSG_VAL_NONE_STR, NIRFSG_VAL_REF_IN_STR	NIRFSG_VAL_PXI_CLK10_STR

Defined Values:

Value	Description
NIRFSG_VAL_NONE_STR	Do not drive the PXI_CLK10 signal.
NIRFSG_VAL_ONBOARD_CLK_STR	Use the highly stable oven-controlled onboard reference clock to drive the PXI_CLK10 signal. This value is not valid on the NI 5672.
NIRFSG_VAL_REF_IN_STR	Use the clock present at the front panel REF IN connector to drive the PXI_CLK10 signal.

Default Value: NIRFSG_VAL_NONE_STR

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_EXPORTED_REF_CLOCK_OUTPUT

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	None

Description

Specifies the destination terminal for exporting the reference clock on the NI 5650/5651/5652 RF signal sources. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

NIRFSG_VAL_REF_OUT_STR	Export the Reference Out clock.
NIRFSG_VAL_DO_NOT_EXPORT_STR	Do not export the Reference Out clock.

Default Value: NIRFSG_VAL_DO_NOT_EXPORT_STR

Supported Devices: NI 5650/5651/5652

NIRFSG_ATTR_ARB_ONBOARD_SAMPLE_CLOC

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the clock mode on the NI 5421/5441/5442 AWG module. To set this attribute, the NI-RFSG device must be in the Configuration state.



Note Using the high resolution clock may result in increased phase noise.

Valid Values:

NIRFSG_VAL_HIGH_RESOLUTION	Specifies that the clock mode is high resolution. High resolution sampling is when a sample rate is generated by a high resolution clock source.
NIRFSG_VAL_DIVIDE_DOWN	Specifies that the clock mode is divide down. Divide down sampling is when sample rates are generated by dividing the source frequency.

Default Value: NIRFSG_VAL_DIVIDE_DOWN

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_ARB_SAMPLE_CLOCK_SOURCE

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	None

Description

Specifies the sample clock source for the NI 5421/5441/5442 AWG module. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

NIRFSG_VAL_ONBOARD_CLK_STR	Use the AWG module onboard clock as the clock source.
NIRFSG_VAL_CLK_IN_STR	Use the external clock as the clock source.

Default Value: NIRFSG_VAL_ONBOARD_CLK_STR

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_ARB_SAMPLE_CLOCK_RATE

Data type	Access	Coercion	High Level Functions
ViReal64	RO	None	None

Description

Returns the rate of the sample clock in Hz on the NI 5421/5441/5442 AWG module.

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_START_TRIGGER_TYPE

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	niRFSG_ConfigureDigitalEdgeStartTrigger niRFSG_ConfigureSoftwareStartTrigger niRFSG_DisableStartTrigger

Description

Specifies the start trigger type. Depending upon the value of this attribute, more attributes may be needed to fully configure the trigger.

Use this attribute to specify whether you want the Start trigger to be a digital edge or software trigger. You can also choose `NIRFSG_VAL_NONE` as the value for this attribute. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

<code>NIRFSG_VAL_NONE</code>	No trigger is configured.
<code>NIRFSG_VAL_DIGITAL_EDGE</code>	The data operation does not start until a digital edge is detected. The source of the digital edge is specified with the NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_SOURCE attribute, and the active edge is specified in the NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_EDGE attribute.
<code>NIRFSG_VAL_SOFTWARE</code>	The data operation does not start until a software event occurs. You may create a software event by calling the niRFSG_SendSoftwareEdgeTrigger function.

Default Value: `NIRFSG_VAL_NONE`

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGEI

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	niRFSG_ConfigureDigitalEdgeStartTrigger

Description

Specifies the source terminal for the start trigger. This attribute is used when the [NIRFSG_ATTR_START_TRIGGER_TYPE](#) attribute is set to NIRFSG_VAL_DIGITAL_EDGE. The NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_SOURCE attribute is not case-sensitive. To set the NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_SOURCE attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

""	Empty string.
NIRFSG_VAL_PFI0_STR	PFI 0 on the front panel SMB connector.
NIRFSG_VAL_PFI1_STR	PFI 1 on the front panel SMB connector.
NIRFSG_VAL_PFI2_STR	PFI 2 on the front panel DDC connector.
NIRFSG_VAL_PFI3_STR	PFI 3 on the front panel DDC connector.
NIRFSG_VAL_PXI_TRIG0_STR	PXI trigger line 0.
NIRFSG_VAL_PXI_TRIG1_STR	PXI trigger line 1.
NIRFSG_VAL_PXI_TRIG2_STR	PXI trigger line 2.
NIRFSG_VAL_PXI_TRIG3_STR	PXI trigger line 3.
NIRFSG_VAL_PXI_TRIG4_STR	PXI trigger line 4.
NIRFSG_VAL_PXI_TRIG5_STR	PXI trigger line 5.
NIRFSG_VAL_PXI_TRIG6_STR	PXI trigger line 6.
NIRFSG_VAL_PXI_TRIG7_STR	PXI trigger line 7.
NIRFSG_VAL_PXI_STAR_STR	PXI Star trigger line.

Default Value: "" (empty string)

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGEREI

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	niRFSG_ConfigureDigitalEdgeStartTrigger

Description

Specifies the active edge for the start trigger. This attribute is used when the [NIRFSG_ATTR_START_TRIGGER_TYPE](#) attribute is set to NIRFSG_VAL_DIGITAL_EDGE. To set the NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_EDGE attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

NIRFSG_VAL_RISING_EDGE	Occurs when the signal transitions from low level to high level.
NIRFSG_VAL_FALLING_EDGE	Occurs when the signal transitions from high level to low level.

Default Value: NIRFSG_VAL_RISING_EDGE

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_EXPORTED_START_TRIGGER_O

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	None

Description

Specifies the destination terminal for exporting the Start trigger. To set this attribute, the NI-RFSG device must be in the Configuration state. For trigger delay information, refer to the [triggering](#) section.

Defined Values:

""	Empty string.
NIRFSG_VAL_PFI0_STR	PFI 0 on the front panel SMB connector.
NIRFSG_VAL_PFI1_STR	PFI 1 on the front panel SMB connector.
NIRFSG_VAL_PFI2_STR	PFI 2 on the front panel DDC connector.
NIRFSG_VAL_PFI3_STR	PFI 3 on the front panel DDC connector.
NIRFSG_VAL_PXI_TRIG0_STR	PXI trigger line 0.
NIRFSG_VAL_PXI_TRIG1_STR	PXI trigger line 1.
NIRFSG_VAL_PXI_TRIG2_STR	PXI trigger line 2.
NIRFSG_VAL_PXI_TRIG3_STR	PXI trigger line 3.
NIRFSG_VAL_PXI_TRIG4_STR	PXI trigger line 4.
NIRFSG_VAL_PXI_TRIG5_STR	PXI trigger line 5.
NIRFSG_VAL_PXI_TRIG6_STR	PXI trigger line 6.
NIRFSG_VAL_PXI_TRIG7_STR	PXI trigger line 7.

Default Value: "" (empty string)

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_SCRIPT_TRIGGER_TYPE

Specific Attribute

Data type	Access	Applies to	Coercion	High Level Functions
ViInt32	R/W	N/A	None	niRFSG_ConfigureDigitalEdgeScriptTrigger

Description

Specifies the script trigger type. Depending upon the value of this attribute, more attributes may be needed to fully configure the trigger. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

NIRFSG_VAL_NONE	No trigger is configured. Signal generation starts immediately.
NIRFSG_VAL_DIGITAL_EDGE	The data operation does not start until a digital edge is detected. The source of the digital edge is specified with the NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_SOURCE attribute, and the active edge is specified with the NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_EDGE attribute.
NIRFSG_VAL_SOFTWARE	The data operation does not start until a software event occurs. You may create a software event by calling the niRFSG_SendSoftwareEdgeTrigger function.

Default Value: NIRFSG_VAL_NONE

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_DIGITAL_EDGE_SCRIPT_TRIGGE

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	niRFSG_ConfigureDigitalEdgeScriptTrigger

Description

Specifies the source terminal for the script trigger. This attribute is used when the [NIRFSG_ATTR_SCRIPT_TRIGGER_TYPE](#) attribute is set to NIRFSG_VAL_DIGITAL_EDGE. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

""	Empty string.
NIRFSG_VAL_PFI0_STR	PFI 0 on the front panel SMB connector.
NIRFSG_VAL_PFI1_STR	PFI 1 on the front panel SMB connector.
NIRFSG_VAL_PFI2_STR	PFI 2 on the front panel DDC connector.
NIRFSG_VAL_PFI3_STR	PFI 3 on the front panel DDC connector.
NIRFSG_VAL_PXI_TRIG0_STR	PXI trigger line 0.
NIRFSG_VAL_PXI_TRIG1_STR	PXI trigger line 1.
NIRFSG_VAL_PXI_TRIG2_STR	PXI trigger line 2.
NIRFSG_VAL_PXI_TRIG3_STR	PXI trigger line 3.
NIRFSG_VAL_PXI_TRIG4_STR	PXI trigger line 4.
NIRFSG_VAL_PXI_TRIG5_STR	PXI trigger line 5.
NIRFSG_VAL_PXI_TRIG6_STR	PXI trigger line 6.
NIRFSG_VAL_PXI_TRIG7_STR	PXI trigger line 7.
NIRFSG_VAL_PXI_STAR_STR	PXI Star trigger line.

Default Value: "" (empty string)

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_DIGITAL_EDGE_SCRIPT_TRIGGE

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	niRFSG_ConfigureDigitalEdgeScriptTrigger

Description

Specifies the active edge for the script trigger. This attribute is used when the [NIRFSG_ATTR_SCRIPT_TRIGGER_TYPE](#) attribute is set to NIRFSG_VAL_DIGITAL_EDGE. To set the NIRFSG_ATTR_DIGITAL_EDGE_SCRIPT_TRIGGER_EDGE attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

NIRFSG_VAL_RISING_EDGE	Occurs when the signal transitions from low level to high level.
NIRFSG_VAL_FALLING_EDGE	Occurs when the signal transitions from high level to low level.

Default Value: NIRFSG_VAL_RISING_EDGE

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_DIGITAL_LEVEL_SCRIPT_TRIGG

Data type	Access	Coercion	High Level Functions
ViString	R/W	None	None

Description

Specifies the source terminal for the Script trigger. This attribute is used when the [NIRFSG_ATTR_SCRIPT_TRIGGER_TYPE](#) attribute is set to NIRFSG_VAL_DIGITAL_LEVEL. The NIRFSG_ATTR_DIGITAL_LEVEL_SCRIPT_TRIGGER_SOURCE attribute is not case-sensitive. To set the NIRFSG_ATTR_DIGITAL_LEVEL_SCRIPT_TRIGGER_SOURCE attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

""	Empty string.
NIRFSG_VAL_PFI0_STR	PFI 0 on the front panel SMB connector.
NIRFSG_VAL_PFI1_STR	PFI 1 on the front panel SMB connector.
NIRFSG_VAL_PFI2_STR	PFI 2 on the front panel DDC connector.
NIRFSG_VAL_PFI3_STR	PFI 3 on the front panel DDC connector.
NIRFSG_VAL_PXI_TRIG0_STR	PXI trigger line 0.
NIRFSG_VAL_PXI_TRIG1_STR	PXI trigger line 1.
NIRFSG_VAL_PXI_TRIG2_STR	PXI trigger line 2.
NIRFSG_VAL_PXI_TRIG3_STR	PXI trigger line 3.
NIRFSG_VAL_PXI_TRIG4_STR	PXI trigger line 4.
NIRFSG_VAL_PXI_TRIG5_STR	PXI trigger line 5.
NIRFSG_VAL_PXI_TRIG6_STR	PXI trigger line 6.
NIRFSG_VAL_PXI_TRIG7_STR	PXI trigger line 7.
NIRFSG_VAL_RTISI0_STR	RTSI trigger line 0.
NIRFSG_VAL_RTISI1_STR	RTSI trigger line 1.
NIRFSG_VAL_RTISI2_STR	RTSI trigger line 2.
NIRFSG_VAL_RTISI3_STR	RTSI trigger line 3.
NIRFSG_VAL_RTISI4_STR	RTSI trigger line 4.
NIRFSG_VAL_RTISI5_STR	RTSI trigger line 5.
NIRFSG_VAL_RTISI6_STR	RTSI trigger line 6.
NIRFSG_VAL_PXI_STAR_STR	PXI STAR Line.

Default Value: "" (empty string)

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_DIGITAL_LEVEL_SCRIPT_TRIGG

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

Specifies the active level for the Script trigger. This attribute is used when the [NIRFSG_ATTR_SCRIPT_TRIGGER_TYPE](#) attribute is set to NIRFSG_VAL_DIGITAL_LEVEL.

Defined Values:

NIRFSG_VAL_ACTIVE_HIGH	Trigger when the digital trigger signal is high.
NIRFSG_VAL_ACTIVE_LOW	Trigger when the digital trigger signal is low.

Default Value:

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_EXPORTED_SCRIPT_TRIGGER_C

Specific Attribute

Data type	Access	Applies to	Coercion	High Level Functions
ViString	R/W	N/A	None	None

Description

Specifies the destination terminal for exporting the script trigger. To set this attribute, the NI-RFSG device must be in the Configuration state. For trigger delay information, refer to the [triggering section](#).

Defined Values:

""	Empty string.
NIRFSG_DO_NOT_EXPORT_STR	The signal is not exported.
NIRFSG_VAL_PFI0_STR	PFI 0 on the front panel SMB connector.
NIRFSG_VAL_PFI1_STR	PFI 1 on the front panel SMB connector.
NIRFSG_VAL_PFI2_STR	PFI 2 on the front panel DDC connector.
NIRFSG_VAL_PFI3_STR	PFI 3 on the front panel DDC connector.
NIRFSG_VAL_PXI_TRIG0_STR	PXI trigger line 0.
NIRFSG_VAL_PXI_TRIG1_STR	PXI trigger line 1.
NIRFSG_VAL_PXI_TRIG2_STR	PXI trigger line 2.
NIRFSG_VAL_PXI_TRIG3_STR	PXI trigger line 3.
NIRFSG_VAL_PXI_TRIG4_STR	PXI trigger line 4.
NIRFSG_VAL_PXI_TRIG5_STR	PXI trigger line 5.
NIRFSG_VAL_PXI_TRIG6_STR	PXI trigger line 6.
NIRFSG_VAL_RTSI0_STR	RTSI trigger line 0.
NIRFSG_VAL_RTSI1_STR	RTSI trigger line 1.
NIRFSG_VAL_RTSI2_STR	RTSI trigger line 2.
NIRFSG_VAL_RTSI3_STR	RTSI trigger line 3.
NIRFSG_VAL_RTSI4_STR	RTSI trigger line 4.
NIRFSG_VAL_RTSI5_STR	RTSI trigger line 5.
NIRFSG_VAL_RTSI6_STR	RTSI trigger line 6.

Default Value: "" (empty string)

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_SELECTED_SCRIPT

Specific Attribute

Data type	Access	Applies to	Coercion	High Level Functions
ViString	R/W	N/A	None	None

Description

Specifies the script in onboard memory to generate upon calling the [niRFSG_Initiate](#) function when the [NIRFSG_ATTR_GENERATION_MODE](#) attribute is set to NIRFSG_VAL_SCRIPT.

The NIRFSG_ATTR_SELECTED_SCRIPT attribute is ignored when the NIRFSG_ATTR_GENERATION_MODE attribute is set to NIRFSG_VAL_ARB_WAVEFORM or NIRFSG_VAL_CW. To set the NIRFSG_ATTR_SELECTED_SCRIPT attribute, the NI-RFSG device must be in the Configuration state.

Supported Devices: NI 5670/5671/5672

NIRFSG_ATTR_MEMORY_SIZE

Data type	Access	Coercion	High Level Functions
ViInt32	RO	None	None

Description

The total amount of memory in bytes on the RF signal generator.

NIRFSG_ATTR_SERIAL_NUMBER

Specific Attribute

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

Description

Returns the serial number of the RF module

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_RANGE_CHECK

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Specifies whether to validate attribute values and function parameters. Range checking parameters is very useful for debugging. After you validate your program, set this attribute to `VI_FALSE` to disable range checking and maximize performance.

Defined Values:

<code>VI_TRUE</code>	Enable range checking.
<code>VI_FALSE</code>	Disable range checking.

Default Value: `VI_TRUE`



Note Use the [niRFSG_InitWithOptions](#) function to override the default value.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_QUERY_INSTRUMENT_STATUS

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Specifies whether NI-RFSG queries the NI-RFSG device status after each operation. Querying the device status is useful for debugging. After you validate your program, set this attribute to `VI_FALSE` to disable status checking and maximize performance.

NI-RFSG can choose to ignore status checking for particular attributes, regardless of the setting of this attribute.

Defined Values:

<code>VI_TRUE</code>	NI-RFSG queries the instrument status after each operation.
<code>VI_FALSE</code>	NI-RFSG does not query the instrument status.

Default Value: `VI_FALSE`



Note Use the [niRFSG_InitWithOptions](#) function to override the default value.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_CACHE

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Specifies whether to cache the value of attributes. When caching is enabled, NI-RFSG tracks the current NI-RFSG device settings and avoids sending redundant commands to the device.

NI-RFSG can always cache or never cache particular attributes, regardless of the setting of this attribute.

Defined Values:

VI_TRUE	Enables caching.
VI_FALSE	Disables caching.

Default Value: VI_TRUE



Note Use the [niRFSG_InitWithOptions](#) function to override this value.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_SIMULATE

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Specifies NI-RFSG simulates I/O operations. This attribute is useful for debugging applications without using hardware. Once a session is opened, you cannot change the simulation state. Use the [niRFSG_InitWithOptions](#) function to enable simulation.

Defined Values:

VI_TRUE	Simulation is enabled.
VI_FALSE	Simulation is disabled.

Default Value: VI_FALSE

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_RECORD_COERCIONS

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Specifies whether the IVI engine keeps a list of the value coercions it makes for integer and real type attributes.

Defined Values:

VI_TRUE	The IVI engine keeps a list of coercions.
VI_FALSE	The IVI engine does not keep a list of coercions.

Default Value: VI_FALSE



Note NIRFSG_ATTR_RECORD_COERCIONS is unsupported.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_INTERCHANGE_CHECK

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	None

Description

Specifies whether to perform interchangeability checking and retrieve interchangeability warnings.

Defined Values:

VI_TRUE	Interchange check is enabled.
VI_FALSE	Interchange check is disabled.

Default Value: VI_FALSE



Note This attribute is currently not supported.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_SPECIFIC_DRIVER_DESCRIPTION

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Returns Returns a string that contains a brief description of NI-RFSG.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_SPECIFIC_DRIVER_PREFIX

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Returns a string that contains the prefix for NI-RFSG. The name of each user-callable function in NI-RFSG starts with this prefix.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_SPECIFIC_DRIVER_VENDOR

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Returns a string that contains the name of the vendor that supplies NI-RFSG.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_SPECIFIC_DRIVER_REVISION

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Returns a string that contains additional version information about NI-RFSG.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_SPECIFIC_DRIVER_CLASS_SPEC

Data type	Access	Coercion	High Level Functions
ViInt32	RO	None	None

Description

The major version number of the class specification with which NI-RFSG is compliant.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_SPECIFIC_DRIVER_CLASS_SPEC

Data type	Access	Coercion	High Level Functions
ViInt32	RO	None	None

Description

The minor version number of the class specification with which NI-RFSG is compliant.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_SUPPORTED_INSTRUMENT_MOI

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Contains a model code of the NI-RFSG device.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_GROUP_CAPABILITIES

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Returns a string that contains a comma-separated list of class-extension groups that NI-RFSG implements.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_INSTRUMENT_MANUFACTURER

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Returns a string that contains the name of the manufacturer of the NI-RFSG device you are currently using.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_INSTRUMENT_MODEL

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Returns a string that contains the model number or name of the NI-RFSG device that you are currently using.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_INSTRUMENT_FIRMWARE_REVISION

Data type	Access	Coercion	High Level Functions
ViString	RO	None	niRFSG_revision_query

Description

Returns a string that contains the firmware revision information for the NI-RFSG device you are currently using.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_LOGICAL_NAME

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Contains the logical name you specified when opening the current IVI session. You can pass a logical name to the [niRFSG_init](#) function or the [niRFSG_InitWithOptions](#) function. The IVI Configuration Utility must contain an entry for the logical name. The logical name entry refers to a driver session section in the IVI Configuration file. The driver session section specifies a physical device and initial user options.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_IO_RESOURCE_DESCRIPTOR

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

Indicates the resource name NI-RFSG uses to identify the physical device. If you initialize NI-RFSG with a logical name, this attribute contains the resource name that corresponds to the entry in the IVI Configuration Utility.

If you initialize NI-RFSG with the resource name, this attribute contains that value.

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_DRIVER_SETUP

Data type	Access	Coercion	High Level Functions
ViString	RO	None	None

Description

The driver setup string is used to set the initial values for attributes that are specific to NI-RFSG.

The driver setup string is in the following format:

Tag:Value

Tag is the name of the driver setup string attribute. *Value* is the value set to the attribute. To set multiple attributes, separate their assignments with a semicolon.

The following describes the DriverSetup string tags:

AWG—specifies the resource name of the Arbitrary Waveform Generator to use for this session. If this driver setup attribute is not specified, the resource name for the upconverter associated in MAX is used.

Example: DriverSetup=AWG:pxi1slot4

Refer to the [niRFSG_InitWithOptions](#) function for additional information about the **optionsString** parameter. Refer to the [NI RF Signal Generators Getting Started Guide](#) for more information about MAX setup.

Default Value: "" (empty string)

Supported Devices: NI 5650/5651/5652/5670/5671/5672

NIRFSG_ATTR_IQ_ENABLED [OBSOLETE]

Data type	Access	Coercion	High Level Functions
ViBoolean	R/W	None	niRFSG_ConfigureIQEnabled

Description

Enables or disables IQ (vector) modulation of the output signal. Enabling this attribute is required for generating arbitrary signals. To set this attribute, the NI-RFSG device must be in the Configuration state.

Defined Values:

VI_TRUE	IQ modulation is enabled..
VI_FALSE	IQ modulation is disabled.

Default Value: VI_FALSE

NIRFSG_ATTR_THERMAL_CORRECTION_ENABI [OBSOLETE]

Data type	Access	Coercion	High Level Functions
ViInt32	R/W	None	None

Description

This obsolete property has no effect.

Specifies whether NI-RFSG periodically tries to correct for output power variations due to changes in temperature.

Enabling this attribute provides automatic adjustment of output signal power for changes in NI 5670 gain with temperature, but it may cause noise on the output signal when the correction is being applied. For additional information about thermal correction, refer to the [temperature monitoring](#) section.

Defined Values:

NIRFSG_VAL_ENABLE	Thermal correction enabled.
NIRFSG_VAL_DISABLED	Thermal correction disabled.

Default Value: NIRFSG_VAL_ENABLE

NIRFSG_ATTR_SAMPLE_CLOCK_RATE [OBSOLETE]

Data type	Access	Coercion	High Level Functions
ViReal64	R/W	None	niRFSG_ConfigureRefClock

Description

Specifies the sample rate at which the arbitrary waveform was produced, expressed in samples per second. Use the [NIRFSG_ATTR_IQ_RATE](#) attribute in place of this attribute.

Defined Values:

Value	Description
50E6	50 MS/s
100E6	100 MS/s

Alphabetical Attribute List and Default Values

The following table lists the default values for each property you can configure for your device. An "N/A" in a table cell indicates that the listed property is not supported for that device. A dash indicates that the property does not have a default value or that it is a read-only property. " " indicates an empty string.

C/C++ Attribute	NI 5650/5651/5652
<u>NIRFSG_ATTR_ALLOW_OUT_OF_SPECIFICATION_USER_SETTINGS</u>	NIRFSG_VAL_DISAB
<u>NIRFSG_ATTR_ANALOG_MODULATION_FM_DEVIATION</u>	1 kHz
<u>NIRFSG_ATTR_ANALOG_MODULATION_TYPE</u>	NIRFSG_VAL_NONE
<u>NIRFSG_ATTR_ANALOG_MODULATION_WAVEFORM_FREQUENCY</u>	1 kHz
<u>NIRFSG_ATTR_ANALOG_MODULATION_WAVEFORM_TYPE</u>	NIRFSG_VAL_SINE
<u>NIRFSG_ATTR_ARB_MAX_NUMBER_WAVEFORMS</u>	N/A
<u>NIRFSG_ATTR_ARB_ONBOARD_SAMPLE_CLOCK_MODE</u>	N/A
<u>NIRFSG_ATTR_ARB_SAMPLE_CLOCK_RATE</u>	N/A
<u>NIRFSG_ATTR_ARB_SAMPLE_CLOCK_SOURCE</u>	N/A
<u>NIRFSG_ATTR_ARB_SELECTED_WAVEFORM</u>	N/A
<u>NIRFSG_ATTR_ARB_WAVEFORM_QUANTUM</u>	N/A
<u>NIRFSG_ATTR_ARB_WAVEFORM_SIZE_MAX</u>	N/A
<u>NIRFSG_ATTR_ARB_WAVEFORM_SIZE_MIN</u>	N/A
<u>NIRFSG_ATTR_ARB_WAVEFORM_SOFTWARE_SCALING_FACTOR</u>	N/A
<u>NIRFSG_ATTR_ATTENUATOR_HOLD_ENABLED</u>	N/A
<u>NIRFSG_ATTR_ATTENUATOR_HOLD_MAX_POWER</u>	N/A
<u>NIRFSG_ATTR_CACHE</u>	VI_TRUE
<u>NIRFSG_ATTR_DATA_TRANSFER_BLOCK_SIZE</u>	N/A
<u>NIRFSG_ATTR_DEVICE_TEMPERATURE</u>	—
<u>NIRFSG_ATTR_DIGITAL_EDGE_SCRIPT_TRIGGER_EDGE</u>	N/A
<u>NIRFSG_ATTR_DIGITAL_EDGE_SCRIPT_TRIGGER_SOURCE</u>	N/A
<u>NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_EDGE</u>	N/A
<u>NIRFSG_ATTR_DIGITAL_EDGE_START_TRIGGER_SOURCE</u>	N/A
<u>NIRFSG_ATTR_DIGITAL_IF_EQUALIZATION_ENABLED</u>	N/A
<u>NIRFSG_ATTR_DIGITAL_LEVEL_SCRIPT_TRIGGER_ACTIVE_LEVEL</u>	N/A
<u>NIRFSG_ATTR_DIGITAL_LEVEL_SCRIPT_TRIGGER_SOURCE</u>	N/A
<u>NIRFSG_ATTR_DIGITAL_MODULATION_FSK_DEVIATION</u>	1 kHz
<u>NIRFSG_ATTR_DIGITAL_MODULATION_PRBS_ORDER</u>	18
<u>NIRFSG_ATTR_DIGITAL_MODULATION_PRBS_SEED</u>	1
<u>NIRFSG_ATTR_DIGITAL_MODULATION_SYMBOL_RATE</u>	1 kHz

<u>NIRFSG ATTR DIGITAL MODULATION TYPE</u>	NIRFSG_VAL_NONE
<u>NIRFSG ATTR DIGITAL MODULATION WAVEFORM TYPE</u>	NIRFSG_VAL_PRBS
<u>NIRFSG ATTR DIGITAL PATTERN</u>	N/A
<u>NIRFSG ATTR DIRECT DMA ENABLED</u>	N/A
<u>NIRFSG ATTR DIRECT DMA WINDOW ADDRESS</u>	N/A
<u>NIRFSG ATTR DIRECT DMA WINDOW SIZE</u>	N/A
<u>NIRFSG ATTR DIRECT DOWNLOAD</u>	N/A
<u>NIRFSG ATTR DRIVER SETUP</u>	""
<u>NIRFSG ATTR DUC FIR FILTER RAISED COSINE ALPHA</u>	N/A
<u>NIRFSG ATTR DUC FIR FILTER TYPE</u>	N/A
<u>NIRFSG ATTR DUC PRE FILTER GAIN</u>	N/A
<u>NIRFSG ATTR EXPORTED REF CLOCK OUTPUT TERMINAL</u>	NIRFSG_VAL_DO_NC
<u>NIRFSG ATTR EXPORTED SCRIPT TRIGGER OUTPUT TERMINAL</u>	N/A
<u>NIRFSG ATTR EXPORTED START TRIGGER OUTPUT TERMINAL</u>	""
<u>NIRFSG ATTR FREQUENCY</u>	100 MHz
<u>NIRFSG ATTR FREQUENCY TOLERANCE</u>	N/A
<u>NIRFSG ATTR GENERATION MODE</u>	NIRFSG_VAL_CW
<u>NIRFSG ATTR GROUP CAPABILITIES</u>	—
<u>NIRFSG ATTR IF CARRIER FREQUENCY</u>	N/A
<u>NIRFSG ATTR IF POWER</u>	N/A
<u>NIRFSG ATTR INSTRUMENT FIRMWARE REVISION</u>	—
<u>NIRFSG ATTR INSTRUMENT MANUFACTURER</u>	—
<u>NIRFSG ATTR INSTRUMENT MODEL</u>	—
<u>NIRFSG ATTR INTERCHANGE CHECK</u>	VI_FALSE
<u>NIRFSG ATTR IO RESOURCE DESCRIPTOR</u>	—
<u>NIRFSG ATTR IQ RATE</u>	N/A
<u>NIRFSG ATTR IQ SWAP ENABLED</u>	N/A
<u>NIRFSG ATTR LOCAL OSCILLATOR OUT 0 ENABLED</u>	N/A
<u>NIRFSG ATTR LOGICAL NAME</u>	—
<u>NIRFSG ATTR MEMORY SIZE</u>	—
<u>NIRFSG ATTR OUTPUT ENABLED</u>	VI_TRUE
<u>NIRFSG ATTR PEAK ENVELOPE POWER</u>	N/A
<u>NIRFSG ATTR PHASE CONTINUITY ENABLED</u>	N/A
<u>NIRFSG ATTR POWER LEVEL</u>	-145 dBm
<u>NIRFSG ATTR POWER LEVEL TYPE</u>	N/A
<u>NIRFSG ATTR PXI CHASSIS CLK10 SOURCE</u>	N/A
<u>NIRFSG ATTR QUERY INSTRUMENT STATUS</u>	VI_FALSE
<u>NIRFSG ATTR RANGE CHECK</u>	VI_TRUE
<u>NIRFSG ATTR RECORD COERCIONS</u>	VI_FALSE

<u>NIRFSG ATTR REF CLOCK RATE</u>	10 MHz
<u>NIRFSG ATTR REF CLOCK SOURCE</u>	NIRFSG_VAL_ONBO.
<u>NIRFSG ATTR SCRIPT TRIGGER TYPE</u>	N/A
<u>NIRFSG ATTR SELECTED SCRIPT</u>	N/A
<u>NIRFSG ATTR SERIAL NUMBER</u>	—
<u>NIRFSG ATTR SIGNAL BANDWIDTH</u>	N/A
<u>NIRFSG ATTR SIMULATE</u>	VI_FALSE
<u>NIRFSG ATTR SPECIFIC DRIVER CLASS SPEC MAJOR VERSION</u>	—
<u>NIRFSG ATTR SPECIFIC DRIVER CLASS SPEC MINOR VERSION</u>	—
<u>NIRFSG ATTR SPECIFIC DRIVER DESCRIPTION</u>	—
<u>NIRFSG ATTR SPECIFIC DRIVER PREFIX</u>	—
<u>NIRFSG ATTR SPECIFIC DRIVER REVISION</u>	—
<u>NIRFSG ATTR SPECIFIC DRIVER VENDOR</u>	—
<u>NIRFSG ATTR START TRIGGER TYPE</u>	N/A
<u>NIRFSG ATTR STREAMING ENABLED</u>	N/A
<u>NIRFSG ATTR STREAMING SPACE AVAILABLE IN WAVEFORM</u>	N/A
<u>NIRFSG ATTR STREAMING WAVEFORM NAME</u>	N/A
<u>NIRFSG ATTR SUPPORTED INSTRUMENT MODELS</u>	—
<u>NIRFSG ATTR UPCONVERTER CENTER FREQUENCY</u>	N/A
<u>NIRFSG ATTR UPCONVERTER CENTER FREQUENCY INCREMENT</u>	N/A
<u>NIRFSG ATTR UPCONVERTER CENTER FREQUENCY INCREMENT ANCHOR</u>	N/A
<u>NIRFSG ATTR UPCONVERTER GAIN</u>	N/A
<u>NIRFSG ATTR UPCONVERTER LOOP BANDWIDTH</u>	N/A