Spectrum analyzer

IVI-C Programming Guide

E01A
Aug, 2022

SIGLENT TECHNOLOGIES CO.,LTD

1 Revision History

This chapter declares the modifications of IVI driver in the most recent release of the programming guide version.

Version E01A at Introduction

This version, as the first version, will be compared with later versions. When the next version is released, the differences between the two versions will be marked.

2 Introduction

2.1 Models Supported

The series of SIGLENT spectrometer supporting this IVI-C driver is shown below.

Series	Release Version Supporting IVI-C Driver
SSA3000X	1.3.9.8 and higher
SVA1000X	3.2.2.5.0 and higher
SSA3000X Plus	3.2.2.5.0 and higher
SSA3000X-R	3.2.2.5.0 and higher
SSA5000A	1.1.2.1.6 and higher
SHA800A	1.1.2.1.0 and higher

2.2 Software Requirement

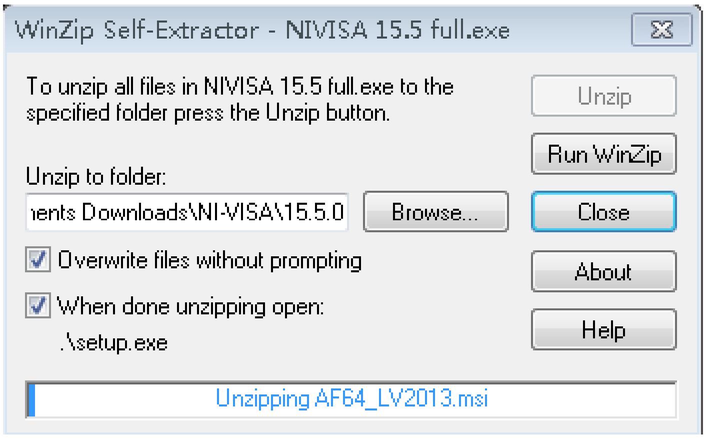
This chapter describes how to configure the IVI driver to control the instrument. If you want to use the IVI Driver, you must install NI-VISA, the IVI Compliance Package, and a C language development system that supports the IVI driver library.

2.3 Install NI-MAX

Currently, NI-VISA is packaged in two versions: Full version and Run-Time Engine version. The full version includes the NI device drivers and a tool named NI-MAX which is a user interface to control and test remotely connected devices. You need to install the full version of NI-VISA.

You can get the NI-VISA 15.5 full version or higher version from *https://www.ni.com/en-us/support/downloads/drivers/download.ni-visa.html#306031*.

a. Double click the NIVISA 15.5 full.exe, a dialog will be shown as below:



WinZip Self-Extractor - NIVISA 15.5 full.exe

To unzip all files in NIVISA 15.5 full.exe to the specified folder press the Unzip button.

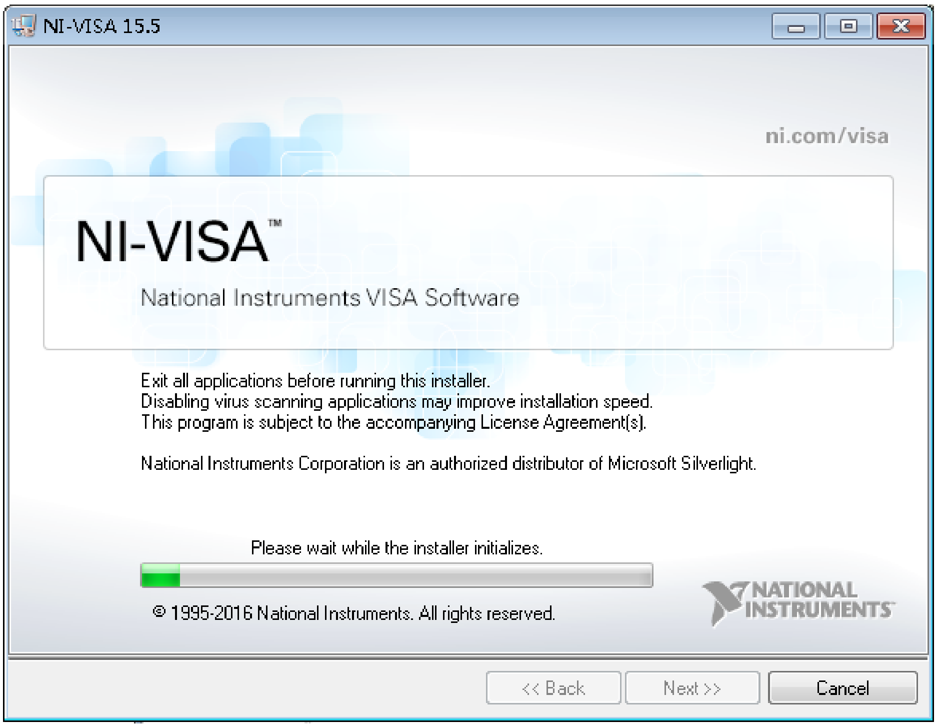
Unzip to folder:
ıents Downloads\NI-VISA\15.5.0

☑ Overwrite files without prompting

☑ When done unzipping open:
.\setup.exe

Unzipping AF64_LV2013.msi

b. Click Unzip, the installation process will automatically launch after unzipping files. If your computer needs to install .NET Framework 4, it may auto start.



NI-VISA 15.5

ni.com/visa

NI-VISA™

National Instruments VISA Software

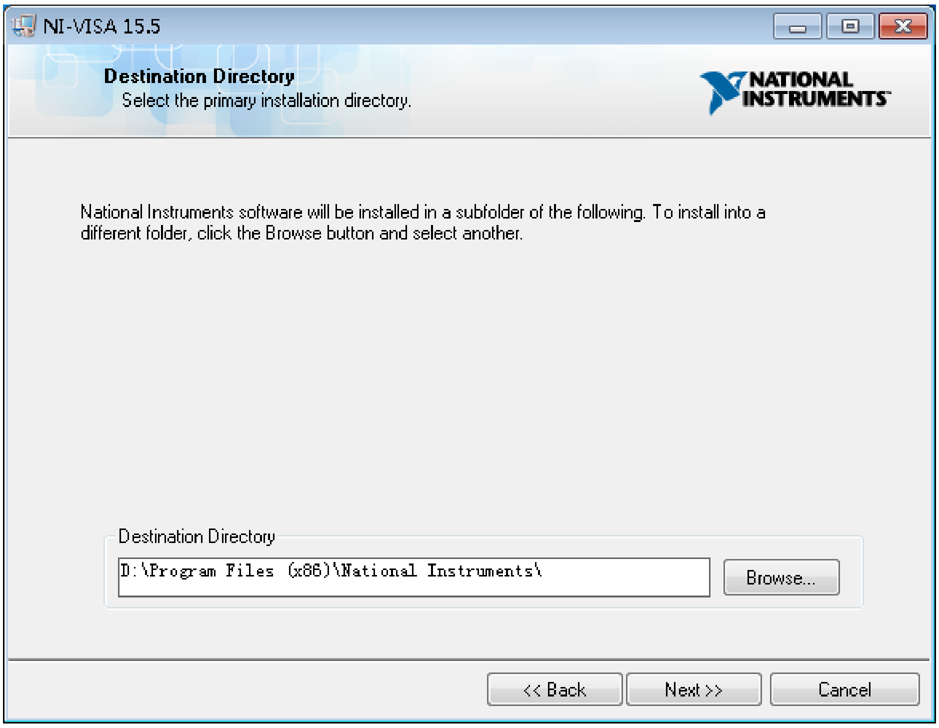
Exit all applications before running this installer.
Disabling virus scanning applications may improve installation speed.
This program is subject to the accompanying License Agreement(s).

National Instruments Corporation is an authorized distributor of Microsoft Silverlight.

Please wait while the installer initializes.

© 1995-2016 National Instruments. All rights reserved.

c. The NI-VISA installing dialog is shown above. Click Next to start the installation process.



Destination Directory

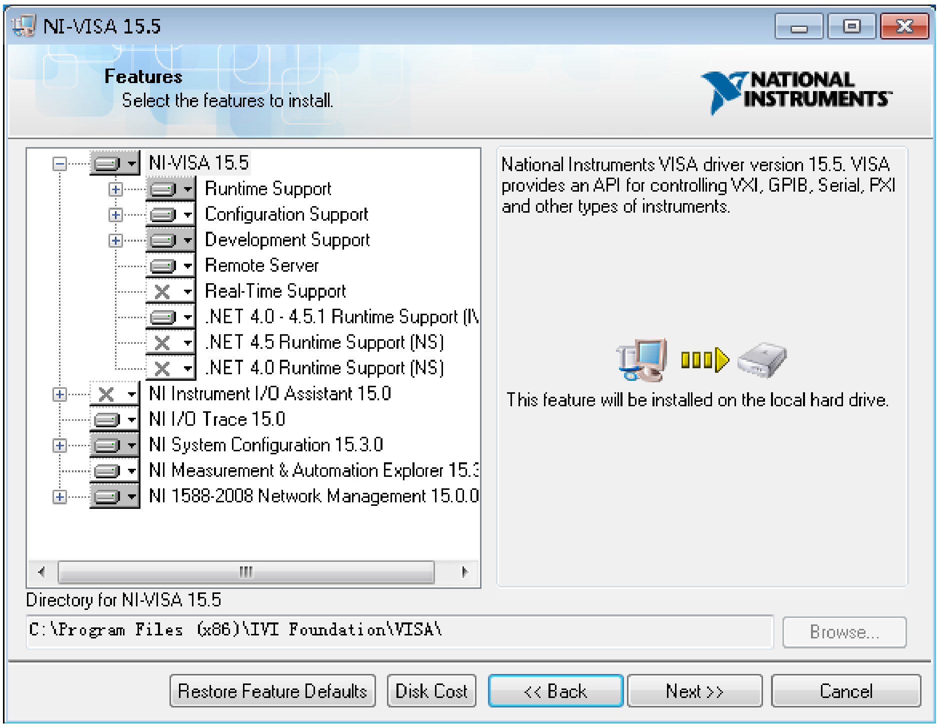
Select the primary installation directory.

National Instruments software will be installed in a subfolder of the following. To install into a different folder, click the Browse button and select another.

Destination Directory

D:\Program Files (x86)\National Instruments\

d. Set the install path. The default path is “C:\Program Files\National Instruments\”. You can change it. Click Next.



Features

Select the features to install.

- NI-VISA 15.5
  - Runtime Support
  - Configuration Support
  - Development Support
  - Remote Server
  - ☒ Real-Time Support
  - .NET 4.0 - 4.5.1 Runtime Support (N
  - ☒ .NET 4.5 Runtime Support (NS)
  - ☒ .NET 4.0 Runtime Support (NS)
- ☒ NI Instrument I/O Assistant 15.0
- NI I/O Trace 15.0
- NI System Configuration 15.3.0
- NI Measurement & Automation Explorer 15.3
- NI 1588-2008 Network Management 15.0.0

National Instruments VISA driver version 15.5. VISA provides an API for controlling VXI, GPIB, Serial, PXI and other types of instruments.

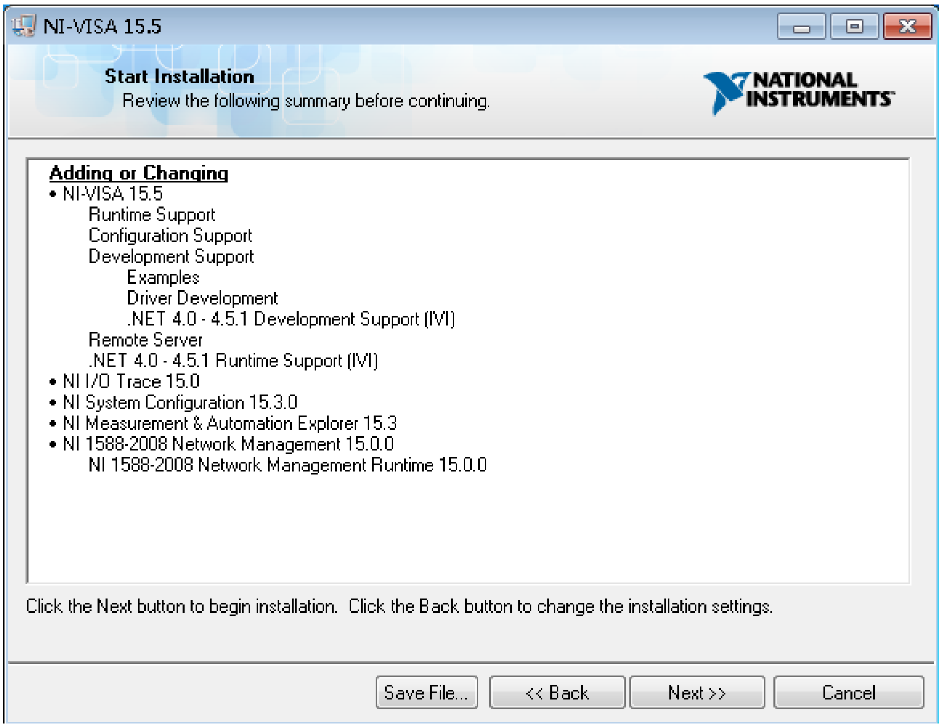
This feature will be installed on the local hard drive.

Directory for NI-VISA 15.5

C:\Program Files (x86)\IVI Foundation\VISA\

e. Click Next twice, in the License Agreement dialog, select “I accept the above 2

License Agreement(s).” ,and click Next.

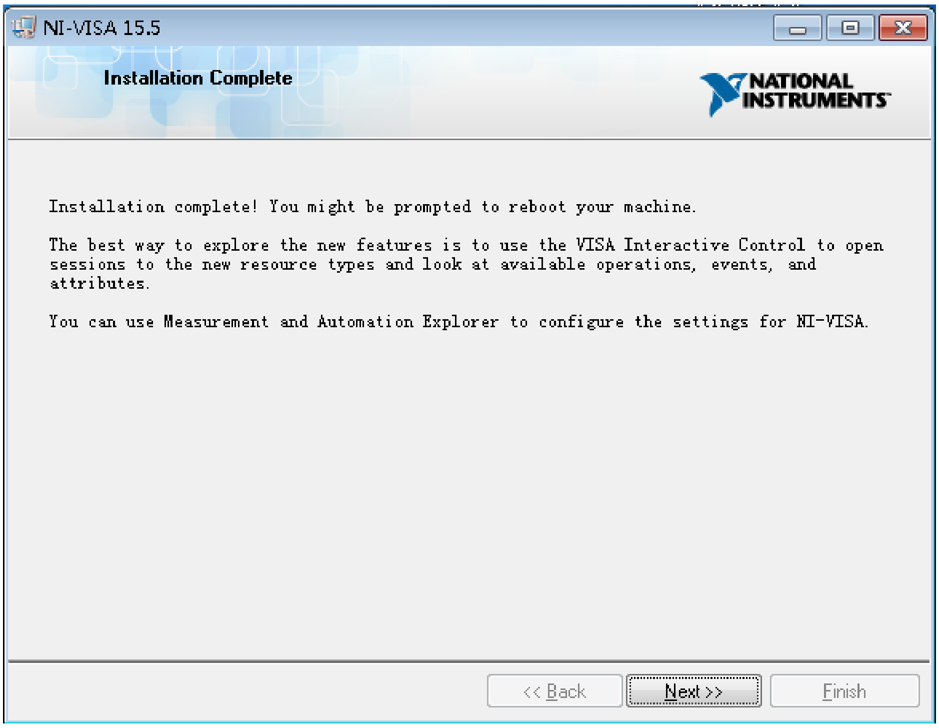


Adding or Changing

- NI-VISA 15.5
  - Runtime Support
  - Configuration Support
  - Development Support
    - Examples
    - Driver Development
    - .NET 4.0 - 4.5.1 Development Support (IVI)
  - Remote Server
  - .NET 4.0 - 4.5.1 Runtime Support (IVI)
- NI I/O Trace 15.0
- NI System Configuration 15.3.0
- NI Measurement & Automation Explorer 15.3
- NI 1588-2008 Network Management 15.0.0
  - NI 1588-2008 Network Management Runtime 15.0.0

Click the Next button to begin installation. Click the Back button to change the installation settings.

f. Click Next to begin the installation.



Installation complete! You might be prompted to reboot your machine.

The best way to explore the new features is to use the VISA Interactive Control to open sessions to the new resource types and look at available operations, events, and attributes.

You can use Measurement and Automation Explorer to configure the settings for NI-VISA.

g. Wait until the installation is completed, and then reboot your PC.

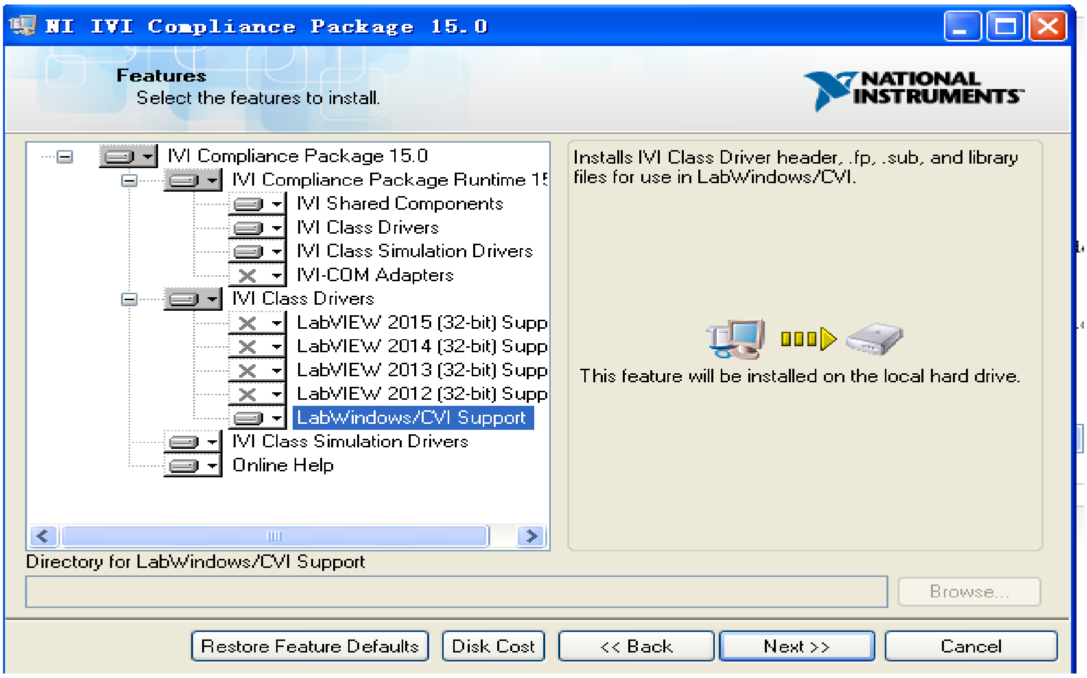
2.4 Install the IVI Compliance Package

The IVI Compliance Package contains the IVI class drivers and supported libraries for developing and leveraging IVI-based applications.

You can get the IVI Compliance Package from *https://www.ni.com/zh-cn/support/downloads/drivers/download.ivi-compliance-package.html#329444*

a. If the IVI Compliance Package is not installed, there is no IVI Drivers option in "My System".

b. Install the IVI Compliance Package (ICP).



NI IVI Compliance Package 15.0

Features
Select the features to install.

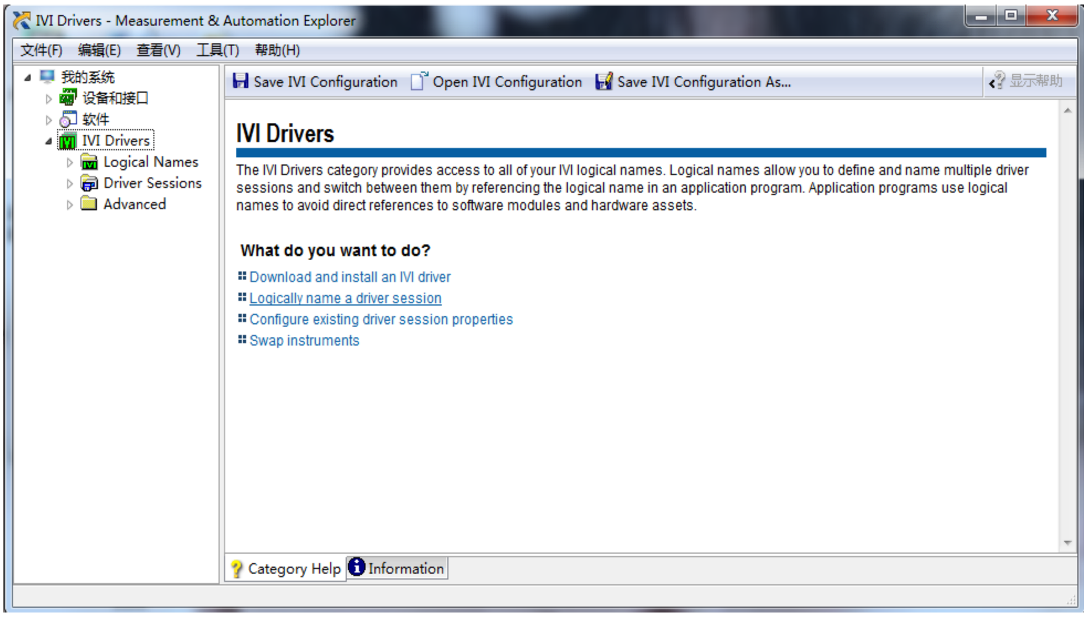
- IVI Compliance Package 15.0
  - IVI Compliance Package Runtime 15
    - IVI Shared Components
    - IVI Class Drivers
    - IVI Class Simulation Drivers
    - ✕ IVI-COM Adapters
  - IVI Class Drivers
    - ✕ LabVIEW 2015 (32-bit) Supp
    - ✕ LabVIEW 2014 (32-bit) Supp
    - ✕ LabVIEW 2013 (32-bit) Supp
    - ✕ LabVIEW 2012 (32-bit) Supp
    - LabWindows/CVI Support
  - IVI Class Simulation Drivers
  - Online Help

Installs IVI Class Driver header, .fp, .sub, and library files for use in LabWindows/CVI.

This feature will be installed on the local hard drive.

Directory for LabWindows/CVI Support

c. Restart your computer after the installation. After the reboot, the IVI Drivers option appears.



IVI Drivers

The IVI Drivers category provides access to all of your IVI logical names. Logical names allow you to define and name multiple driver sessions and switch between them by referencing the logical name in an application program. Application programs use logical names to avoid direct references to software modules and hardware assets.

What do you want to do?

- Download and install an IVI driver
- Logically name a driver session
- Configure existing driver session properties
- Swap instruments

2.5 SSA IVI-C Driver Package List

The SSA IVI-C driver package provides three kinds of files: ssa.dll file, ssa.h file and ssa.lib file.

File	Description
ssa.dll/ssa_64.dll	A dynamic link library file, including variables, functions, and data interfaces for various attributes.
ssa.lib/ssa_64.lib	An import library file, including the symbolic name and optional identification number of each exported function in the ssa.dll file.
ssa.h	A header file, including declarations of variables, functions, and data interfaces.

You include the ssa.h when programming the Siglent oscilloscope with the IVI driver, and load the ssa.dll dynamic file or ssa.lib import library file into your own project.

You will find an example that show you how to use these files at the end of this document.

3 Introduction to IVI

IVI (Interchangeable Virtual Instruments) is a new generation of instrument driver technology specifications introduced by the IVI Foundation. IVI can realize the interchangeability with the instrument, the instrument simulation, and the instrument state tracking and buffer function. All references to IVI drivers in this document refer to IVI-C drivers that are created using NI tools and that rely on the IVI Engine.

3.1 IVI Data Type

There are six data types for the attributes of the IVI Engine: ViInt32, ViReal64, ViString, ViBoolean, ViSession and ViAddr.

Table 1 Data Type

Data Type	**Description**
ViInt32	32-bit signed integer
ViReal64	64-bit floating-point number
ViString	String type
ViBoolean	Boolean value
Visession	A VISA session handle
ViAddr	Logical address type

3.2 Access IVI Attribute

User-callable functions are typically implemented by manipulating attributes. You can call ssa_SetAttribute or ssa_GetAttribute functions.

3.2.1 SetAttribute Function Group

3.2.1.1 ssa_SetAttributeViInt32 (ViSession vi, ViConstString channelName, ViAttr attributeId, ViInt32 value)

Example: When you want to set the sweep mode, you can call the SetAttribute function to change the sweep mode.

ssa_SetAttributeViInt32(session, VI_NULL, SSA_ATTR_SWEEP_MODE,2)	
session	The instrument handle.
2	Set the scan mode to FFT.

3.2.1.2 ssa_SetAttributeViReal64 (ViSession vi, ViConstString channelName, ViAttr attributeId, ViReal64 value)

Example: When you want to set the start frequrncy, you can call SetAttribute or GetAttribute function to change or obtain the start frequency value.

ssa_SetAttributeViReal64(session,VI_NULL,SSA_ATTR_FREQUENCY_START,1000);	
session	The instrument handle.
1000	Set the starting frequency to 1khz.

3.2.1.3 ssa_SetAttributeViString (ViSession vi, ViConstString channelName, ViAttr attributeId, ViConstString value)

Example: When you want to set the marker label trace, you can call SetAttribute or GetAttribute function to change or obtain the active marker label trace.

ssa_SetAttributeViString(session,VI_NULL,SSA_ATTR_MARKER_TRACE,"TRACE1") ;	
session	The instrument handle.
“TRACE1”	set active marker mark trace 1.

3.2.1.4 ssa_SetAttributeViBoolean (ViSession vi, ViConstString channelName, ViAttr attributeId, ViBoolean value)

Example: When you want to set frequency counter on or off, you can call SetAttribute or GetAttribute function to change or obtain the state of the frequency counter.

ssa_SetAttributeViBoolean(session,VI_NULL, *SSA_ATTR_MARKER_FREQUENCY_COUNTER_ENABLED, VI_TRUE);*	
session	The instrument handle.
VI_TRUE	Open frequency counter switch.

3.2.2 GetAttribute Function Group

3.2.2.1 ssa_GetAttributeViInt32 (ViSession vi, ViConstString channelName, ViAttr attributeId, ViInt32 *value)

Example: When you want to set the trace type, you can call SetAttribute or GetAttribute function to change or obtain the trace type.

ssa_GetAttributeViInt32(session, VI_NULL, SSA_ATTR_TRACE_TYPE, &value32);	
session	The instrument handle.
value32	A ViInt32 type variable which is used to store the returned value of the active trace type.

3.2.2.2 ssa_GetAttributeViReal64 (ViSession vi, ViConstString channelName, ViAttr attributeId, ViReal64 *value)

Example: When you want to get the start frequency, you can call GetAttribute function to get the start frequency value.

ssa_GetAttributeViReal64(session,VI_NULL, SSA_ATTR_FREQUENCY_START, &value64);	
session	The instrument handle.
value64	A ViReal64 type variable which is used to store the returned value of the start frequency.

3.2.2.3 ssa_GetAttributeViString (ViSession vi, ViConstString channelName, ViAttr attributeId, ViInt32 bufSize, ViChar value[])

Example: When you want to get the active trace, you can call GetAttribute function to get the active trace.

ssa_GetAttributeViString(session,VI_NULL,SSA_ATTR_ACTIVE_TRACE, buffersize,str);	
session	The instrument handle.
buffersize	A ViInt32 type variable.
str	A ViString type variable which is used to store the returned value.

3.2.2.4 ssa_GetAttributeViBoolean (ViSession vi, ViConstString channelName, ViAttr attributeId, ViBoolean *value)

Example: When you want to get the frequency counter state, you can call GetAttribute function to get the frequency counter state.

ssa_GetAttributeViBoolean(session,VI_NULL, SSA_ATTR_MARKER_FREQUENCY_COUNTER_ENABLED, &boolean);	
session	The instrument handle.
boolean	A ViBoolean type variable which is used to store the freq count state returned value.

4 Attributes

system		Attribute
Basic	1.	SSA_ATTR_AMPLITUDE_UNITS
	2.	SSA_ATTR_ATTENUATION
	3.	SSA_ATTR_ATTENUATION_AUTO
	4.	SSA_ATTR_DETECTOR_TYPE
	5.	SSA_ATTR_DETECTOR_TYPE_AUTO
	6.	SSA_ATTR_FREQUENCY_START
	7.	SSA_ATTR_FREQUENCY_STOP
	8.	SSA_ATTR_FREQUENCY_OFFSET
	9.	SSA_ATTR_INPUT_IMPEDANCE
	10.	SSA_ATTR_NUMBER_OF_SWEEPS
	11.	SSA_ATTR_REFERENCE_LEVEL
	12.	SSA_ATTR_REFERENCE_LEVEL_OFFSET
	13.	SSA_ATTR_RESOLUTION_BANDWIDTH
	14.	SSA_ATTR_RESOLUTION_BANDWIDTH_AUTO
	15.	SSA_ATTR_SWEEP_MODE_CONTINUOUS
	16.	SSA_ATTR_SWEEP_TIME
	17.	SSA_ATTR_SWEEP_TIME_AUTO
	18.	SSA_ATTR_SWEEP_MODE
	19.	SSA_ATTR_VERTICAL_SCALE
	20.	SSA_ATTR_VIDEO_BANDWIDTH
	21.	SSA_ATTR_VIDEO_BANDWIDTH_AUTO
	22.	SSA_ATTR_FREQUENCY_SPAN_MODE
	23.	SSA_ATTR_FREQUENCY_SPAN
	24.	SSA_ATTR_CENTER_FREQUENCY
	25.	SSA_ATTR_INSTRUMENT_MODE
	26.	SSA_ATTR_MEASUREMENT_TYPE
	27.	SSA_ATTR_AVERAGE_COUNT
	28.	SSA_ATTR_AVERAGE_ENABLE
	29.	SSA_ATTR_AVERAGE_TYPE
Trace	1.	SSA_ATTR_TRACE_SIZE
	2.	SSA_ATTR_TRACE_TYPE
	3.	SSA_ATTR_TRACE_MATH_TYPE
	4.	SSA_ATTR_ACTIVE_TRACE
TG	1.	SSA_ATTR_TG_NORMAILIZE_REFERENCE_POSITION
	2.	SSA_ATTR_TG_NORMALIZE_REFERENCE_LEVEL
	3.	SSA_ATTR_TG_NORMALIZE_ENABLE
	4.	SSA_ATTR_TG_OUTPUT_AMPLITUDE
	5.	SSA_ATTR_TG_OUTPUT_AMPLITUDE_OFFSET
	6.	SSA_ATTR_TG_OUTPUT_AMPLITUDE_ENABLE

CHP	1.	SSA_ATTR_CHP_CHANNEL_SPAN
	2.	SSA_ATTR_CHP_CENTER_FREQUENCY
	3.	SSA_ATTR_CHP_INTEGRATION_BANDWIDTH
ACPR	1.	SSA_ATTR_ACPR_MAIN_CHANNEL_INTERGRATION_BANDWIDTH
	2.	SSA_ATTR_ACPR_CENTER_FREQUENCY
OBW	1.	SSA_ATTR_OBW_POWER_LEVEL
	2.	SSA_ATTR_OBW_POWER_PERCENTAGE
	3.	SSA_ATTR_OBW_METHOD
Trigger	1.	SSA_ATTR_TRIGGER_SOURCE
	2.	SSA_ATTR_VIDEO_TRIGGER_LEVEL
	3.	SSA_ATTR_VIDEO_TRIGGER_SLOPE
	4.	SSA_ATTR_EXTERNAL_TRIGGER_SLOPE
Marker	1.	SSA_ATTR_ACTIVE_MARKER
	2.	SSA_ATTR_MARKER_AMPLITUDE
	3.	SSA_ATTR_MARKER_ENABLED
	4.	SSA_ATTR_MARKER_FREQUENCY_COUNTER_ENABLED
	5.	SSA_ATTR_MARKER_POSITION
	6.	SSA_ATTR_MARKER_THRESHOLD
	7.	SSA_ATTR_MARKER_TRACE
	8.	SSA_ATTR_MARKER_X_READOUT
	9.	SSA_ATTR_MARKER_FUNCTION
	10.	SSA_ATTR_MARKER_TYPE
	11.	SSA_ATTR_MARKER_INSTRUMENT_SETTING
	12.	SSA_ATTR_MARKER_PEAK_SEARCH
	13.	SSA_ATTR_PEAK_SEARCH_TYPE
	14.	SSA_ATTR_PEAK_EXCURSION
	15.	SSA_ATTR_MARKER_CONTINUOUS_PEAKING_ENABLE
	16.	SSA_ATTR_SIGNAL_TRACK_ENABLED
	17.	SSA_ATTR_MARKER_DEMODULATION_DELAY_TIME
	18.	SSA_ATTR_MARKER_DEMODULATION_SPEAKER_VOLUME
	19.	SSA_ATTR_MARKER_DEMODULATION_FUNCTION

4.1 Base Attributes

4.1.1 Amplitude Units

Attributes Defines	SSA_ATTR_AMPLITUDE_UNITS
Data Type	ViInt32
Access	R/W

Common Control Functions	ssaAttrAmplitudeUnits_ReadCallback ssaAttrAmplitudeUnits_WriteCallback
High Level Functions	ssa_ConfigureLevel
Description	Specifies the amplitude units for input, output and display amplitude.

Value Range

Enumeration	Attribute Value Defines	value
dBm	SSA_VAL_AMPLITUDE_UNITS_DBM	1
dBmV	SSA_VAL_AMPLITUDE_UNITS_DBMV	2
dBuV	SSA_VAL_AMPLITUDE_UNITS_DBUV	3
Volt	SSA_VAL_AMPLITUDE_UNITS_VOLT	4
Watt	SSA_VAL_AMPLITUDE_UNITS_WATT	5

4.1.2 Attenuation

Attributes Defines	SSA_ATTR_ATTENUATION
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureLevel
Description	Specifies the input attenuation (in positive dB).
Value Range	Depends on the maximum attenuation

4.1.3 Attenuation Auto

Attributes Defines	SSA_ATTR_ATTENUATION_AUTO

Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureLevel
Description	If set to True, attenuation is automatically selected. If set to False, attenuation is manually selected.
Value Range	0|1

4.1.4 Detector Type

Attributes Defines	SSA_ATTR_DETECTOR_TYPE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaAttrDetectorType_ReadCallback ssaAttrDetectorType_WriteCallback
High Level Functions	ssa_ConfigureAcquisition
Description	Specifies the detection method used to capture and process the signal. This governs the data acquisition for a particular sweep, but does not have any control over how multiple sweeps are processed.

Value Range

Enumeration	Attribute Value Defines	value
Average	SSA_VAL_DETECTOR_TYPE_AVERAGE	2
Maximum Peak	SSA_VAL_DETECTOR_TYPE_MAX_PEAK	3
Minimum Peak	SSA_VAL_DETECTOR_TYPE_MIN_PEAK	4
Sample	SSA_VAL_DETECTOR_TYPE_SAMPLE	5
Normal	SSA_VAL_DETECTOR_TYPE_NORMAL	6

4.1.5 Detector Type Auto

Attributes Defines	SSA_ATTR_DETECTOR_TYPE_AUTO
Data Type	ViBoolean
Access	R/W
Common Control Functions	None
High Level Functions	ssa_ConfigureLevel
Description	If set to True, the detector type is automatically selected.The relationship between Trace Type and Detector Type is not defined by the specification when the Detector Type Auto is set to True. If set to False, the detector type is manually selected.
Value Range	0|1

4.1.6 Frequency Start

Attributes Defines	SSA_ATTR_FREQUENCY_START
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaAttrFrequencyStart_ReadCallback ssaAttrFrequencyStart_WriteCallback
High Level Functions	ssa_ConfigureFrequencyStartStop ssa_ConfigureFrequencyCenterSpan
Description	Specifies the left edge of the frequency domain in Hertz. This is used in conjunction with the Frequency Stop attribute to define the frequency domain. If the Frequency Start attribute value is equal to the Frequency Stop attribute value then the spectrum analyzer's horizontal attributes are in time-domain.
Value Range	Depends on the maximum bandwidth

4.1.7 Frequency Stop

Attributes Defines	SSA_ATTR_FREQUENCY_ STOP
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaAttrFrequencyStop_ReadCallback ssaAttrFrequencyStop_WriteCallback
High Level Functions	ssa_ConfigureFrequencyStart Stop ssa_ConfigureFrequencyCenterSpan
Description	Specifies the right edge of the frequency domain in Hertz. This is used in conjunction with the Frequency Start attribute to define the frequency domain. If the Frequency Start attribute value is equal to the Frequency Stop attribute value then the spectrum analyzer's horizontal attributes are in time-domain.
Value Range	Depends on the maximum bandwidth

4.1.8 Frequency Offset

Attributes Defines	SSA_ATTR_FREQUENCY_OFFSET
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaAttrFrequencyOffset_ReadCallback ssaAttrFrequencyOffset_WriteCallback
High Level Functions	ssa_ConfigureFrequencyOffset
Description	Specifies an offset value, in Hertz, that is added to the frequency readout. The offset is used to compensate for external frequency conversion. This changes the driver's Frequency Start and Frequency Stop attributes. The equations relating the affected values are: Frequency Start = Actual Start Frequency + Frequency Offset Frequency Stop = Actual Stop Frequency + Frequency Offset Marker Position = Actual Marker Frequency + Frequency Offset
Value Range	-100G~100G

4.1.9 Input Impedance

Attributes Defines	SSA_ATTR_INPUT_IMPEDANCE
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaAttrInputImpedance_ReadCallback ssaAttrInputImpedance_WriteCallback
High Level Functions	ssa_ ConfigureLevel
Description	Specifies the value of input impedance, in ohms, expected at the active input port. This is typically 50 ohms or 75 ohms.
Value Range	50/75

4.1.10 Number Of Sweeps

Attributes Defines	SSA_ATTR_NUMBER_OF_SWEEPS
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaViInt32_ReadCallback ssaViInt32_WriteCallback
High Level Functions	ssa_ConfigureAcquisition
Description	This attribute defines the number of sweeps. This attribute value has no effect if the Trace Type attribute is set to the value Clear Write.
Value Range	1~999

4.1.11 Reference Level

Attributes Defines	SSA_ATTR_REFERENCE_LEVEL
Data Type	ViReal64

Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ ConfigureLevel
Description	The calibrated vertical position of the captured data used as a reference for amplitude measurements. This is typically set to a value slightly higher than the highest expected signal level. The units are determined by the Amplitude Units attribute.
Value Range	Depends on the max reflevel and min reflevel.

4.1.12 Reference Level Offset

Attributes Defines	SSA_ATTR_REFERENCE_LEVEL_OFFSET
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureLevel
Description	Specifies an offset for the Reference Level attribute. This value is used to adjust the reference level for external signal gain or loss. A positive value corresponds to a gain while a negative number corresponds to a loss. The value is in dB.
Value Range	-100dB~100dB

4.1.13 Resolution Bandwidth

Attributes Defines	SSA_ATTR_RESOLUTION_BANDWIDTH
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback

	ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureSweepCoupling
Description	Specifies the width of the IF filter in Hertz.
Value Range	1Hz/3Hz/10Hz/30Hz/100Hz/300Hz/1kHz/3kHz/10kHz/30kHz/100 kHz/300kHz/1MHz/3MHz/10M

4.1.14 Resolution Bandwidth Auto

Attributes Defines	SSA_ATTR_RESOLUTION_BANDWIDTH_AUTO
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureSweepCoupling
Description	If set to True, the resolution bandwidth is automatically selected. If set to False, the resolution bandwidth is manually selected.
Value Range	0|1

4.1.15 Sweep Mode Continuous

Attributes Defines	SSA_ATTR_SWEEP_MODE_CONTINUOUS
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureAcquisition
Description	If set to True, the sweep mode is continuous　If set to False, the sweep mode is not continuous..

Value Range	0|1

4.1.16 Sweep Time

Attributes Defines	SSA_ATTR_SWEEP_TIME
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureSweepCoupling
Description	Specifies the length of time to sweep from the left edge to the right edge of the current domain.

4.1.17 Sweep Time Auto

Attributes Defines	SSA_ATTR_SWEEP_TIME_AUTO
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureSweepCoupling
Description	If set to True, the sweep time is automatically selected　If set to False, the sweep time is manually selected..
Value Range	0|1

4.1.18 Sweep Mode

Attributes Defines	SSA_ATTR_SWEEP_MODE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaAttrSweepMode_ReadCallback ssaAttrSweepMode_WriteCallback
High Level Functions	none
Description	Sets the sweep mode.

Value Range:

Enumeration	Attribute Value Defines	value
Sweep	SSA_VAL_SWEEP_MODE_SWEEP	1
FFT	SSA_VAL_SWEEP_MODE_FFT	2

4.1.19 Vertical Scale

Attributes Defines	SSA_ATTR_VERTICAL_SCALE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureAcquisition
Description	Specifies the vertical scale of the measurement hardware (use of log amplifiers versus linear amplifiers)

Value Range:

Enumeration	Attribute Value Defines	value

LIN	SSA_VAL_VERTICAL_SCALE_LINEAR	1
LOG	SSA_VAL_VERTICAL_SCALE_LOGARITHMIC	2

4.1.20 Video Bandwidth

Attributes Defines	SSA_ATTR_VIDEO_BANDWIDTH
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureSweepCoupling
Description	Specifies the video bandwidth of the post-detection filter in Hertz.
Value Range	1Hz/3Hz/10Hz/30Hz/100Hz/300Hz/1kHz/3kHz/10kHz/30kHz/100kHz/300 kHz/1MHz/3MHz/10M

4.1.21 Video Bandwidth Auto

Attributes Defines	SSA_ATTR_VIDEO_BANDWIDTH_AUTO
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureSweepCoupling
Description	If set to True, the video bandwidth is automatically selected. If set to False, the video bandwidth is manually selected.
Value Range	0|1

4.1.22 Span Mode

Attributes Defines	SSA_ATTR_FREQUENCY_SPAN_MODE
Data Type	ViInt32
Access	WO
Common Control Functions	ssaAttrFrequencySpanMode_WriteCallback
High Level Functions	ssa_ConfigureFrequencySpanMode
Description	Sets the frequency span to full scale, zero span or the previous span setting.

Value Range:

Enumeration	Attribute Value Defines	value
Full span	SSA_VAL_FREQUENCY_SPAN_MODE_FULL	1
Zero span	SSA_VAL_FREQUENCY_SPAN_MODE_ZERO	2
Last span	SSA_VAL_FREQUENCY_SPAN_MODE_PREVIOUS	3

4.1.23 Frequency Span

Attributes Defines	SSA_ATTR_FREQUENCY_SPAN
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaAttrFrequencySpan_ReadCallback ssaAttrFrequencySpan_WriteCallback
High Level Functions	ssa_ConfigureFrequencyCenterSpan
Description	This function configures the frequency range defining the span. Frequency Start = CenterFrequency - Span / 2 Frequency Stop = CenterFrequency + Span / 2
Value Range	Depends on the maximum bandwidth

4.1.24 Frequency Center

Attributes Defines	SSA_ATTR_CENTER_FREQUENCY
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaAttrCenterFrequency_ReadCallback ssaAttrCenterFrequency_ReadCallback
High Level Functions	ssa_ConfigureFrequencyCenterSpan
Description	this function configures the frequency range defining the center frequency. Frequency Start = CenterFrequency - Span / 2 Frequency Stop = CenterFrequency + Span / 2
Value Range	Depends on the maximum bandwidth

4.1.25 Instrument Mode

Attributes Defines	SSA_ATTR_INSTRUMENT_MODE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureMeasurementType
Description	This function selects the measurement type.

Value Range:

Enumeration	Attribute Value Defines	value
SA	SSA_VAL_INSTRUMENT_MODE_SPECTRUM_ANALYZER	1

4.1.26 Measurement Type

Attributes Defines	SSA_ATTR_MEASUREMENT_TYPE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaAttrMeasurementType_ReadCallback ssaAttrMeasurementType_WriteCallback
High Level Functions	ssa_ConfigureMeasurementType
Description	This function selects the measurement type. Please set the Instrument Mode to Measurement Type to SSA_VAL_INSTRUMENT_MODE_SPECTRUM_ANALYZER when configuring the Measurement Type.

Value Range:

Enumeration	Attribute Value Defines	value
Swept Sa	SSA_VAL_MEASUREMENT_TYPE_SA	0
Channel Power	SSA_VAL_MEASUREMENT_TYPE_CHP	1
ACPR	SSA_VAL_MEASUREMENT_TYPE_ACPR	2
Occupied BW	SSA_VAL_MEASUREMENT_TYPE_OBW	3
T-POWer	SSA_VAL_MEASUREMENT_TYPE_TPOWER	4
TOI	SSA_VAL_MEASUREMENT_TYPE_TOI	5
SPECtrum Monitor	SSA_VAL_MEASUREMENT_TYPE_SM	6
CNR	SSA_VAL_MEASUREMENT_TYPE_CNR	7
Harmonic	SSA_VAL_MEASUREMENT_TYPE_HARM	8

4.1.27 Average Count

Attributes Defines	SSA_ATTR_AVERAGE_COUNT

Data Type	ViInt32
Access	R/W
Common Control Functions	ssaViInt32_ReadCallback ssaViInt32_WriteCallback
High Level Functions	ssa_ConfigureAverage
Description	Set average times.
Value Range	1~999

4.1.28 Average Enable

Attributes Defines	SSA_ATTR_AVERAGE_ENABLE
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureAverage
Description	Turn on/off the average switch.
Value Range	0|1

4.1.29 Average Type

Attributes Defines	SSA_ATTR_AVERAGE_TYPE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureAverage

Description	Set the average type.

Value Range:

Enumeration	Attribute Value Defines	value
Log power	SSA_VAL_AVERAGE_TYPE_LOGPOWER	1
Power	SSA_VAL_AVERAGE_TYPE_POWER	2
Voltage	SSA_VAL_AVERAGE_TYPE_VOLTAGE	3

4.2 Trace

4.2.1 Trace Size

Attributes Defines	SSA_ATTR_TRACE_SIZE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaViInt32_ReadCallback ssaViInt32_WriteCallback
High Level Functions	ssa_QueryTraceSize
Description	Returns the number of points in the trace array.
Value Range	Depends on the maximum number of points

4.2.2 Trace Type

Attributes Defines	SSA_ATTR_TRACE_TYPE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaAttrTraceType_ReadCallback ssaAttrTraceType_WriteCallback

High Level Functions	ssa_ConfigureTraceType
Description	Specifies the representation of the acquired data.

Value Range

Enumeration	Attribute Value Defines	value
WRITe	SSA_VAL_TRACE_TYPE_CLEAR_WRITE	1
MAXHold	SSA_VAL_TRACE_TYPE_MAX_HOLD	2
MINHold	SSA_VAL_TRACE_TYPE_MIN_HOLD	3
VIEW	SSA_VAL_TRACE_TYPE_VIEW	5
BLANk	SSA_VAL_TRACE_TYPE_STORE	6

4.2.3 Trace Math Type

Attributes Defines	SSA_ATTR_TRACE_MATH_TYPE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureTrace
Description	Sets the mathtype of the trace.

Value Range

Enumeration	Attribute Value Defines	value
Off	SSA_VAL_TRACE_MATH_TYPE_OFF	1
Power diff	SSA_VAL_TRACE_MATH_TYPE_POWER_DIFF	2
Power sum	SSA_VAL_TRACE_MATH_TYPE_POWER_SUM	3

Log offset	SSA_VAL_TRACE_MATH_TYPE_LOG_OFFSET	4
Log diff	SSA_VAL_TRACE_MATH_TYPE_LOG_DIFF	5

4.2.4 ActiveTrace

Attributes Defines	SSA_ATTR_ACTIVE_TRACE
Data Type	ViString
Access	R/W
Common Control Functions	ssaAttrActiveTrace_ReadCallback ssaAttrActiveTrace_WriteCallback
High Level Functions	ssa_SetActiveTrace ssa_GetActiveTrace ssa_ConfigureTraceType ssa_QueryTraceSize
Description	selects one of the available traces, and makes it the active trace.
Value Range	TRACE1/ TRACE2/ TRACE3/ TRACE4 TRACE5/ TRACE6(only ssa5000x)

4.3 TG

4.3.1 Normalize ref position

Attributes Defines	SSA_ATTR_TG_NORMAILIZE_REFERENCE_POSITION
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaViInt32_ReadCallback ssaViInt32_WriteCallback
High Level Functions	ssa_ConfigureNormalize
Description	Set the normalized reference position.

Value Range	0%~100%

4.3.2 Normalize Reference Level

Attributes Defines	SSA_ATTR_TG_NORMALIZE_REFERENCE_LEVEL
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaViInt32_ReadCallback ssaViInt32_WriteCallback
High Level Functions	ssa_ConfigureNormalize
Description	Set the normalized reference level
Value Range	-200dB ~ 200dB

4.3.3 Normalize Enable

Attributes Defines	SSA_ATTR_TG_NORMALIZE_ENABLE
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureNormalize
Description	Set the normalized reference level
Value Range	0|1

4.3.4 Output Power

Attributes Defines	SSA_ATTR_TG_OUTPUT_AMPLITUDE

Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureTrackingGenerator
Description	Set the TG output power.
Value Range	-40dB~0dB(offset = 0)

4.3.5 Output Power Offset

Attributes Defines	SSA_ATTR_TG_OUTPUT_AMPLITUDE_OFFSET
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureTrackingGenerator
Description	Set the TG output power offset
Value Range	-200dB ~ 200dB

4.3.6 Output Enabled

Attributes Defines	SSA_ATTR_TG_OUTPUT_AMPLITUDE_ENABLE
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureTrackingGenerator

Description	Set the TG output switch.
Value Range	0|1

4.4 CHP

4.4.1 Channel Span

Attributes Defines	SSA_ATTR_CHP_CHANNEL_SPAN
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureCHP
Description	Set the CHP span
Value Range	Depends on the maximum bandwidth

4.4.2 Center Frequency

Attributes Defines	SSA_ATTR_CHP_CENTER_FREQUENCY
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureCHP
Description	Set the CHP center frequency.
Value Range	Depends on the maximum bandwidth

4.4.3 Integration Bandwidth

Attributes Defines	SSA_ATTR_CHP_INTEGRATION_BANDWIDTH
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureCHP
Description	Set the CHP integral bandwidth
Value Range	Depends on the maximum bandwidth

4.5 ACPR

4.5.1 Main Intergration Bandwidth

Attributes Defines	SSA_ATTR_ACPR_MAIN_CHANNEL_INTERGRATION_BANDWIDTH
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureACPR
Description	Set the bandwidth of the ACPR main channel
Value Range	Depends on the maximum bandwidth

4.5.2 Center Frequency

Attributes Defines	SSA_ATTR_ACPR_CENTER_FREQUENCY
Data Type	ViReal64

Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureACPR
Description	Set the ACPR center frequency.
Value Range	Depends on the maximum bandwidth

4.6 OBW

4.6.1 OBW Power Level

Attributes Defines	SSA_ATTR_OBW_POWER_LEVEL
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureOBW
Description	Set the OBW power x dB.
Value Range	-100dB~100dB

4.6.2 OBW Power Percentage

Attributes Defines	SSA_ATTR_OBW_POWER_PERCENTAGE
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback

High Level Functions	ssa_ConfigureOBW
Description	Set the OBW power percentage.
Value Range	0%~99.99%

4.6.3 OBW Method

Attributes Defines	SSA_ATTR_OBW_METHOD
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureOBW
Description	Set the OBW method.

Value Range

Enumeration	Attribute Value Defines	value
Percent	SSA_VAL_OBW_METHOD_PERCENT	1
dBC	SSA_VAL_OBW_METHOD_DBC	2

4.7 Trigger

4.7.1 Trigger Source

Attributes Defines	SSA_ATTR_TRIGGER_SOURCE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback

	ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureTriggerSource
Description	Specifies the source of the trigger signal that causes the analyzer to leave the Wait-For-Trigger state.

Value Range

Enumeration	Attribute Value Defines	value
External	SSA_VAL_TRIGGER_SOURCE_EXTERNAL	1
Free	SSA_VAL_TRIGGER_SOURCE_IMMEDIATE	2
Video	SSA_VAL_TRIGGER_SOURCE_VIDEO	5

4.7.2 Video Trigger Level

Attributes Defines	SSA_ATTR_VIDEO_TRIGGER_LEVEL
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureVideoTrigger
Description	Specifies the level that the video signal shall reach to trigger the acquisition. The units are specified by the Amplitude Units attribute.
Value Range	-300dBm~50dBm

4.7.3 Video Trigger Slope

Attributes Defines	SSA_ATTR_VIDEO_TRIGGER_SLOPE
Data Type	ViInt32
Access	R/W

Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureVideoTrigger
Description	Specifies which slope of the video signal triggers the acquisition.

Value Range

Enumeration	Attribute Value Defines	value
Positive	SSA_VAL_VIDEO_TRIGGER_SLOPE_POSITIVE	1
Negative	SSA_VAL_VIDEO_TRIGGER_SLOPE_NEGATIVE	2

4.7.4 External Trigger Slope

Attributes Defines	SSA_ATTR_EXTERNAL_TRIGGER_SLOPE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureExternalTrigger
Description	Specifies which slope of the external trigger signal triggers the acquisition.

Value Range

Enumeration	Attribute Value Defines	value
Positive	SSA_VAL_VIDEO_TRIGGER_SLOPE_POSITIVE	1
Negative	SSA_VAL_VIDEO_TRIGGER_SLOPE_NEGATIVE	2

4.8 Marker

4.8.1 Active Marker

Attributes Defines	SSA_ATTR_ACTIVE_MARKER
Data Type	ViString
Access	R/W
Common Control Functions	ssaAttrActiveMarker_ReadCallback ssaAttrActiveMarker_WriteCallback
High Level Functions	ssa_SetActiveMarker ssa_GetActiveMarker
Description	Specifies the marker which is currently active. The values for this attribute correspond to the Marker repeated capability. If the driver defines a qualified Marker name, this attribute returns the qualified name
Value Range	MARKER1,MARKER2,MARKER3,MARKER4,MARKER5,MARKER6, MARKER7,MARKER8

Value Range

Enumeration	Attribute Value Defines	value
Percent	SSA_VAL_OBW_METHOD_PERCENT	1
dBC	SSA_VAL_OBW_METHOD_DBC	2

4.8.2 Marker Amplitude

Attributes Defines	SSA_ATTR_MARKER_AMPLITUDE
Data Type	ViReal64
Access	RO
Common Control Functions	ssaAttrMarkerAmplitude_ReadCallback

High Level Functions	ssa_QueryMarker
Description	Returns the amplitude of the active marker. The units are specified by the Amplitude Units attribute, except when the Marker Type attribute is set to Delta. Then the units are dB. If the Marker Enabled attribute is set to False, any attempt to read this attribute returns the Marker Not Enabled error.

4.8.3 Marker Enabled

Attributes Defines	SSA_ATTR_MARKER_ENABLED
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureMarkerEnabled
Description	If set to True , the active marker is enabled. When False, the active marker is disabled.
Value Range	0|1

4.8.4 Marker Frequency Counter Enabled

Attributes Defines	SSA_ATTR_MARKER_FREQUENCY_COUNTER_ENABLED
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaAttrMarkerFrequencyCounterEnabled_ReadCallback ssaAttrMarkerFrequencyCounterEnabled_WriteCallback
High Level Functions	ssa_ConfigureMarkerFrequencyCounter
Description	Enables/disables the marker frequency counter for greater marker measurement accuracy. If set to True, the marker frequency counter is enabled. If set to False, the marker frequency counter is disabled. This attribute returns the Marker Not Enabled error if the Marker Enabled attribute is set to False.

Value Range	0|1

4.8.5 Marker Position

Attributes Defines	SSA_ATTR_MARKER_POSITION
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaAttrMarkerPosition_ReadCallback ssaAttrMarkerPosition_WriteCallback
High Level Functions	ssa_MoveMarker ssa_QueryMarker
Description	Specifies the frequency in Hertz or time position in seconds of the active marker (depending on the mode in which the analyzer is operating, frequency or time-domain). This attribute returns the Marker Not Enabled error if the active marker is not enabled.
Value Range	Depends on the bandwidth.

4.8.6 Marker Threshold

Attributes Defines	SSA_ATTR_MARKER_THRESHOLD
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaAttrMarkerThreshold_ReadCallback ssaAttrMarkerThreshold_WriteCallback
High Level Functions	ssa_ConfigureMarkerSearch
Description	Specifies the lower limit of the search domain vertical range for the Marker Search function.
Value Range	-200dBm~200dBm

4.8.7 Marker Trace

Attributes Defines	SSA_ATTR_MARKER_TRACE
Data Type	ViString
Access	R/W
Common Control Functions	ssaAttrMarkerTrace_ReadCallback ssaAttrMarkerTrace_WriteCallback
High Level Functions	ssa_ConfigureMarkerEnabled
Description	Specifies the Trace for the active marker.
Value Range	TRACE1,TRACE2,TRACE3,TRACE4,TRACE5,TRACE6

4.8.8 Marker x readout

Attributes Defines	SSA_ATTR_MARKER_X_READOUT
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	none
Description	Set marker x readout type.

Value Range

Enumeration	Attribute Value Defines	value
Frequency	SSA_VAL_MARKER_X_READOUT_FREQUENCY	1
Time	SSA_VAL_MARKER_X_READOUT_TIME	2
Period	SSA_VAL_MARKER_X_READOUT_PERIOD	3

4.8.9 Marker Function

Attributes Defines	SSA_ATTR_MARKER_FUNCTION
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureMarkerType
Description	Selects the type of markers that you want to activate. Notes: The user must call ssa_SetActiveMarker function ahead to specify the active marker before calling this function.

Value Range

Enumeration	Attribute Value Defines	value
Noisy	SSA_VAL_MARKER_FUNCTION_NOISE	1
Ndb	SSA_VAL_MARKER_FUNCTION_NDB	2
Off	SSA_VAL_MARKER_FUNCTION_OFF	3

4.8.10 Marker type

Attributes Defines	SSA_ATTR_MARKER_FUNCTION
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaAttrMarkerType_ReadCallback ssaAttrMarkerType_WriteCallback
High Level Functions	ssa_ConfigureMarkerType ssa_QueryMarkerType

	ssa_MakeMarkerDelta
Description	Selects the marker type.

Value Range

Enumeration	Attribute Value Defines	value
Normal	SSA_VAL_MARKER_TYPE_NORMAL	1
Delta	SSA_VAL_MARKER_TYPE_DELTA	2

4.8.11 Marker to

Attributes Defines	SSA_ATTR_MARKER_INSTRUMENT_SETTING
Data Type	ViInt32
Access	W
Common Control Functions	ssaAttrMarkerInstrumentSetting_WriteCallback
High Level Functions	ssa_SetInstrumentFromMarker
Description	Uses the Marker Position attributes to configure the spectrum analyzer setting specified by the Instrument Setting parameter

Value Range

Enumeration	Attribute Value Defines	value
To center	SSA_VAL_INSTRUMENT_SETTING_FREQUENCY_CENTER	1
To start	SSA_VAL_INSTRUMENT_SETTING_FREQUENCY_START	2
To stop	SSA_VAL_INSTRUMENT_SETTING_FREQUENCY_STOP	3

4.8.12 Other Peak Search

Attributes Defines	SSA_ATTR_MARKER_PEAK_SEARCH
Data Type	ViInt32
Access	W
Common Control Functions	ssaAttrMarkerPeakSearch_WriteCallback
High Level Functions	ssa_MarkerSearch
Description	Specifies the type of marker search and performs the search. This function returns the Marker Not Enabled error if the Marker Enabled attribute is set to False.

Value Range

Enumeration	Attribute Value Defines	value
Peak	SSA_VAL_MARKER_SEARCH_HIGHEST	1
Left	SSA_VAL_MARKER_SEARCH_NEXT_PEAK_LEFT	2
Right peak	SSA_VAL_MARKER_SEARCH_NEXT_PEAK_RIGHT	3
Next peak	SSA_VAL_MARKER_SEARCH_NEXT_PEAK	4

4.8.13 Peak Search type

Attributes Defines	SSA_ATTR_PEAK_SEARCH_TYPE
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureMarkerPeakSearch
Description	Set peak mode as max or min.

Value Range

Enumeration	Attribute Value Defines	value
Max	SSA_VAL_PEAK_SEARCH_MODE_MAXIMUM	1
Min	SSA_VAL_PEAK_SEARCH_MODE_MINIMUM	2

4.8.14 Peak Excursion

Attributes Defines	SSA_ATTR_PEAK_EXCURSION
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureMarkerSearch
Description	Specifies the minimum amplitude variation of the signal in dB that the Marker Search function can identify as a peak.
Value Range	0dBm~200dBm

4.8.15 Continuous Peak

Attributes Defines	SSA_ATTR_MARKER_CONTINUOUS_PEAKING_ENABLE
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaAttrMarkerContinuousPeakingEnable_ReadCallback ssaAttrMarkerContinuousPeakingEnable_WriteCallback
High Level Functions	ssa_ConfigureMarkerPeakSearch
Description	Set the continuous peak or not.

Value Range	1|0

4.8.16 Signal Track Enabled

Attributes Defines	SSA_ATTR_SIGNAL_TRACK_ENABLED
Data Type	ViBoolean
Access	R/W
Common Control Functions	ssaViBoolean_ReadCallback ssaViBoolean_WriteCallback
High Level Functions	ssa_ConfigureSignalTrackEnabled
Description	If set to True, the spectrum analyzer centers the signal after each sweep. This process invalidates the Frequency Start and Frequency Stop attributes. If set to False, the spectrum analyzer does not center the signal after each sweep. Operations on this attribute return the Marker Not Enabled error if the active marker is not enabled. Note: Signal tracking can only be enabled on one marker at any given time. The driver is responsible for enforcing this policy.
Value Range	0|1

4.8.17 Demod Delay Time

Attributes Defines	SSA_ATTR_MARKER_DEMODULATION_DELAY_TIME
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureDemodulation
Description	Set the demodulation time.
Value Range	5ms~1ks

4.8.18 Demod Speaker Volume

Attributes Defines	SSA_ATTR_MARKER_DEMODULATION_SPEAKER_VOLUME
Data Type	ViReal64
Access	R/W
Common Control Functions	ssaViReal64_ReadCallback ssaViReal64_WriteCallback
High Level Functions	ssa_ConfigureDemodulation
Description	Set the demodulation volume.
Value Range	1~10

4.8.19 Demodulation Mode

Attributes Defines	SSA_ATTR_MARKER_DEMODULATION_FUNCTION
Data Type	ViInt32
Access	R/W
Common Control Functions	ssaEnum_ReadCallback ssaEnum_WriteCallback
High Level Functions	ssa_ConfigureDemodulation
Description	Set demodulation mode.

Value Range

Enumeration	Attribute Value Defines	value
AM	SSA_VAL_MARKER_DEMODULATION_FUNCTION_AM	1
FM	SSA_VAL_MARKER_DEMODULATION_FUNCTION_FM	2

5 High Level Functions

5.1 Basic

5.1.1 Abort

Description

This function aborts a previously initiated measurement and returns the spectrum analyzer to the idle state.

This function does not check instrument status.

C Function Prototype

```
ssa_Abort (ViSession vi)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession

5.1.2 AcquisitionStatus

Description

This function determines and returns the status of an acquisition.

C Function Prototype

```
ssa_AcquisitionStatus (ViSession Vi, ViInt32 *Status);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession

Outputs	Description	Base Type
Status (C/COM)	Returns the acquisition status.	ViInt32

Defined Values for Status Parameter

Name	Identifier	Description
Acquisition Complete	SSA_VAL_ACQUISITION_STATUS_COMPLETE	The spectrum analyzer has completed the acquisition.
Acquisition In Progress	SSA_VAL_ACQUISITION_STATUS _IN_PROGRESS	The spectrum analyzer is still acquiring data.
Acquisition Status Unknown	SSA_VAL_ACQUISITION_STATUS_UNKNOWN	The spectrum analyzer cannot determine the status of the acquisition.

5.1.3 ConfigureAcquisition

Description

This function configures the acquisition attributes of the spectrum analyzer.

C Function Prototype

```
ssa_ConfigureAcquisition (ViSession Vi,
                             ViBoolean SweepModeContinuous,
                             ViInt32 NumberOfSweeps,
                             ViBoolean DetectorTypeAuto,
                             ViInt32 DetectorType,
                             ViInt32 VerticalScale);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`SweepModeContinuous`	Enables or disables continuous sweeping. The driver uses this value to set the Sweep Mode Continuous attribute. See the attribute description for more details.	`ViBoolean`
`NumberOfSweeps`	Specifies the number of sweeps to take. The driver uses this value to set the Number Of Sweeps attribute. See the attribute description for more details.	`ViInt32`

`DetectorTypeAuto`	Enables or Disables the auto detector. The driver uses this value to set the Detector Type Auto attribute. See the attribute description for more details.	`ViBoolean`
`DetectorType`	Specifies the method of capturing and processing signal data. The driver uses this value to set the Detector Type attribute. See the attribute description for more details.	`ViInt32`
`VerticalScale`	Specifies the vertical scale. The driver uses this value to set the Vertical Scale attribute. See the attribute description for more details.	`ViInt32`

5.1.4 ConfigureFrequencyCenterSpan

Description

This function configures the frequency range defining the center frequency and the frequency span. If the span corresponds to zero Hertz, then the spectrum analyzer operates in time-domain mode. Otherwise, the spectrum analyzer operates in frequency-domain mode,

This function modifies the Frequency Start and Frequency Stop attributes as follows:

Frequency Start = CenterFrequency – Span / 2
Frequency Stop = CenterFrequency + Span / 2

C Function Prototype

```
ssa_ConfigureFrequencyCenterSpan (ViSession Vi,
                                  ViReal64 CenterFrequency,
                                  ViReal64 Span);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`CenterFrequency`	Specifies the center frequency of the frequency sweep. The units are Hertz.	`ViReal64`
`Span`	Specifies the frequency span of the frequency sweep. The units are Hertz.	`ViReal64`

5.1.5 ConfigureFrequencyOffset

Description

This function configures the Frequency Offset attribute of the spectrum analyzer. This function affects the setting of the spectrum analyzer's absolute frequencies, such as start, stop, center, and marker. It does not affect values such as span and delta marker, which are the difference of frequencies.

C Function Prototype

```
ssa_ConfigureFrequencyOffset (ViSession Vi,
                                  ViReal64 FrequencyOffset);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`FrequencyOffset`	Specifies the frequency offset. The driver uses this value to set the Frequency Offset attribute. See the attribute description for more details. The units are Hertz.	`ViReal64`

5.1.6 ConfigureFrequencyStartStop

Description

This function configures the frequency range defining its start frequency and its stop frequency. If the start frequency is equal to the stop frequency, then the spectrum analyzer operates in time-domain mode. Otherwise, the spectrum analyzer operates in frequency-domain mode.

C Function Prototype

```
ssa_ConfigureFrequencyStartStop (ViSession Vi,
                                   ViReal64 StartFrequency,
                                   ViReal64 StopFrequency);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`StartFrequency`	Specifies the start frequency of the frequency sweep (in Hertz). The driver uses this value to set the Frequency Start attribute. See the attribute description for more details.	`ViReal64`
`StopFrequency`	Specifies the stop frequency of the frequency sweep (in Hertz). The driver uses this value to set the Frequency Stop attribute. See the attribute description for more details.	`ViReal64`

5.1.7 ConfigureLevel

Description

This function configures the vertical attributes of the spectrum analyzer. This corresponds to the Amplitude Units, Input Attenuation, Input Impedance, Reference Level, and Reference Level Offset attributes.

C Function Prototype

```
ssa_ConfigureLevel (ViSession Vi,
                    ViInt32 AmplitudeUnits,
                    ViReal64 InputImpedance,
                    ViReal64 ReferenceLevel,
                    ViReal64 ReferenceLevelOffset,
                    ViBoolean AttenuationAuto,
                    ViReal64 Attenuation);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`AmplitudeUnits`	Specifies the amplitude units for input, output and display. The driver uses this value to set the Amplitude Units attribute. See the attribute description for more details.	`ViInt32`
`InputImpedance`	Specifies the input impedance. The driver uses this value to set the Input Impedance attribute. See the attribute description for more details.	`ViReal64`
`ReferenceLevel`	Specifies the amplitude value of the reference level. The driver uses this value to set the Reference Level attribute. See the attribute description for more details.	`ViReal64`
`ReferenceLevelOffset`	Specifies the offset value to the reference level. The driver uses this value to set the Reference Level Offset attribute. See the attribute description for more details.	`ViReal64`
`AttenuationAuto`	Enables or disables auto attenuation. The driver uses this value to set the Attenuation Auto attribute. See the attribute description for more details.	`ViBoolean`
`Attenuation`	Specifies the attenuation level. If AttenuationAuto is True then this parameter is ignored. The driver uses this value to set the Attenuation attribute. See the attribute description for more details.	`ViReal64`

Defined Values for AmplitudeUnits Parameter

Name	Identifier	value
dBm	SSA_VAL_AMPLITUDE_UNITS_DBM	1
dBmV	SSA_VAL_AMPLITUDE_UNITS_DBMV	2
dBuV	SSA_VAL_AMPLITUDE_UNITS_DBUV	3
Volt	SSA_VAL_AMPLITUDE_UNITS_VOLT	4
Watt	SSA_VAL_AMPLITUDE_UNITS_WATT	5

5.1.8 ConfigureSweepCoupling

Description

This function configures the coupling and sweeping attributes.

C Function Prototype

```
ssa_ConfigureSweepCoupling (ViSession Vi,
                            ViBoolean ResolutionBandwidthAuto,
                            ViReal64 ResolutionBandwidth,
                            ViBoolean VideoBandwidthAuto,
                            ViReal64 VideoBandwidth,
                            ViBoolean SweepTimeAuto,
                            ViReal64 SweepTime);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession
ResolutionBandwidthAuto	Enables or disables resolution bandwidth auto coupling. The driver uses this value to set the Resolution Bandwidth Auto attribute. See the attribute description for more details.	ViBoolean

`ResolutionBandwidth`	Specifies the measurement resolution bandwidth in Hertz. This value is ignored when ResolutionBandwidthAuto is True. The driver uses this value to set the Resolution Bandwidth attribute. See the attribute description for more details.	`ViReal64`
`VideoBandwidthAuto`	Enables or disables video bandwidth auto coupling. The driver uses this value to set the Video Bandwidth Auto attribute. See the attribute description for more details.	`ViBoolean`
`VideoBandwidth`	Specifies the video bandwidth of the post-detection filter in Hertz. This value is ignored when VideoBandwidthAuto is True. The driver uses this value to set the Video Bandwidth attribute. See the attribute description for more details.	`ViReal64`
`SweepTimeAuto`	Enables or disables sweep time auto coupling. The driver uses this value to set the Sweep Time Auto attribute. See the attribute description for more details.	`ViBoolean`
`SweepTime`	Specifies the length of time to sweep from the left edge to the right edge of the current domain.	`ViReal64`

5.1.9 ConfigureTraceType

Description

This function configures the Trace Type attribute.

C Function Prototype

```
ssa_ConfigureTraceType (ViSession Vi,
                        ViConstString TraceName,
                        ViInt32 TraceType);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`TraceName`	Specifies the trace name.	`ViConstString`

`TraceType`	Specifies the type of trace. The driver uses this value to set the Trace Type attribute. See the attribute description for more details.	`ViInt32`

Defined Values for TraceType Parameter

Name	Identifier	value
WRITe	SSA_VAL_TRACE_TYPE_CLEAR_WRITE	1
MAXHold	SSA_VAL_TRACE_TYPE_MAX_HOLD	2
MINHold	SSA_VAL_TRACE_TYPE_MIN_HOLD	3
VIEW	SSA_VAL_TRACE_TYPE_VIEW	5
BLANk	SSA_VAL_TRACE_TYPE_STORE	6

5.1.10 FetchYTrace

Description

This function returns the trace the spectrum analyzer acquires. The trace is from a previously initiated acquisition. The user calls the Initiate function to start an acquisition. The user calls the Acquisition Status function to determine when the acquisition is complete.

The user may call the Read Y Trace function instead of the Initiate function. This function starts an acquisition, waits for the acquisition to complete, and returns the trace in one function call.

The Amplitude array returns data that represents the amplitude of the signals obtained by sweeping from the start frequency to the stop frequency (in frequency domain, in time domain the amplitude array is ordered from beginning of sweep to end). The Amplitude Units attribute determines the units of the points in the Amplitude array.

This function does not check the instrument status. The user calls the Error Query function at the conclusion of the sequence to check the instrument status.

C Function Prototype

```
ssa_FetchYTrace (ViSession Vi,
                          ViConstString TraceName,
                          ViInt32 ArrayLength,
                          ViInt32 *ActualPoints,
                          ViReal64 Amplitude[]);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession
TraceName	Specifies the trace to return.	ViConstString
ArrayLength	Specifies the number of array points requested.	ViInt32

Outputs	Description	Base Type
ActualPoints	Specified the number of points actually returned in the array.	ViInt32
Amplitude[] (C/COM)	Specifies a user allocated (IVI-C) or driver-allocated (IVI-COM) buffer into which the trace amplitudes are stored.	ViReal64

5.1.11 GetTraceName

Description

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

C Function Prototype

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

Parameters

Inputs	Description	Base Type
Vi	Unique identifier for an IVI session	ViSession
Index	A one-based index that defines which name to return.	ViInt32
NameBufferSize	Specifies the number of bytes in the ViChar array referenced by the Name parameter.	ViInt32

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

5.1.12 Initiate

Description

This function initiates an acquisition. After calling this function, the spectrum analyzer leaves the idle state.

This function does not check the instrument status. The user calls the Acquisition Status function to determine when the acquisition is complete.

C Function Prototype

```
ssa_Initiate (ViSession Vi);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`

5.1.13 QueryTraceSize

Description

This function queries the read-only Trace Size attribute.

C Function Prototype

```
ssa_QueryTraceSize (ViSession Vi,
                      ViConstString TraceName,
                      ViInt32 *TraceSize);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle	`ViSession`
`TraceName`	Specifies the Trace name.	`ViConstString`

Outputs	Description	Base Type
`TraceSize`	Returns the size of the Trace.	`ViInt32`

5.1.14 ReadYTrace

Description

This function initiates a signal acquisition based on the present instrument configuration. It then waits for the acquisition to complete, and returns the trace as an array of amplitude values. The amplitude array returns data that represent the amplitude of the signals obtained by sweeping from the start frequency to the stop frequency (in frequency domain, in time domain the amplitude array is ordered from beginning of sweep to end). The Amplitude Units attribute determines the units of the points in the amplitude array. This function resets the sweep count.

If the spectrum analyzer did not complete the acquisition within the time period the user specified with the `MaxTime` parameter, the function returns the Max Time Exceeded error.

C Function Prototype

```
ssa_ReadYTrace (ViSession Vi,
                          ViConstString TraceName,
                          ViInt32 MaxTimeMilliseconds,
                          ViInt32 ArrayLength,
                          ViInt32 *ActualPoints,
                          ViReal64 Amplitude[]);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`TraceName`	Specifies the trace to return.	`ViConstString`
`MaxTimeMilliseconds (C/COM)`	Specifies the maximum length of time allowed for the function to complete in milliseconds.	`ViInt32`
`ArrayLength`	Specifies the number of points in the `Amplitude` array .	`ViInt32`

Outputs	Description	Base Type
`ActualPoints`	Specifies the number of points actually returned in the `Amplitude` array.	`ViInt32`
`Amplitude[] (C/COM)`	Specifies a user allocated (IVI-C) or driver-allocated (IVI-COM) buffer into which the trace amplitudes are stored.	`ViReal64`

5.1.15 ConfigureNormalize

Description

Configures normalize settings.

C Function Prototype

```
        ssa_ConfigureNormalize (ViSession vi,
                                ViBoolean NormalizeEnable,
                                ViInt32 NormalizeReferenceLevel,
                                ViInt32 NormalizeReferencePosition)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
NormalizeEnable	The normalized switch.	ViBoolean
NormalizeReferenceLevel	Normalized reference level.	ViInt32
NormalizeReferencePosition	Normalized reference position.	ViInt32

5.1.16 ConfigureTrackingGenerator

Description

This function configures the tracking generator, including the output power, attenuation, amplitude offset, power sweep and output power tracking

C Function Prototype

```
        ssa_ConfigureTrackingGenerator (ViSession vi,

                                                ViBoolean OutputEnabled,

                                                ViReal64 OutputPower,

                                                ViReal64 OutputPowerOffset)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
OutputEnabled	TG output switch	ViBoolean
OutputPower	TG output power	ViReal64
OutputPowerOffset	TG output power offset	ViReal64

5.1.17 ConfigureFrequencySpanMode

Description

Sets the frequency span to full scale, zero span or the previous span setting.

C Function Prototype

```
        ssa_ConfigureFrequencySpanMode (ViSession vi,

                                                 ViInt32 FrequencySpanMode)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
FrequencySpanMode	Span mode	ViInt32

Defined Values for FrequencySpanMode Parameter

Name	Identifier	value
Full span	SSA_VAL_FREQUENCY_SPAN_MODE_FULL	1
Zero span	SSA_VAL_FREQUENCY_SPAN_MODE_ZERO	2
Last span	SSA_VAL_FREQUENCY_SPAN_MODE_PREVIOUS	3

5.1.18 ConfigureVBWRBWRatio

Description

This function configures the ratio of the video bandwidth to the resolution bandwidth.

C Function Prototype

```
ssa_ConfigureVBWRBWRatio (ViSession vi,
                                ViBoolean RatioAuto,
                                ViReal64 Ratio)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
RatioAuto	RBW /VBW automatic mode switch	ViBoolean
Ratio	RBW /VBW value	ViReal64

5.1.19 ConfigureAverage

Description

Configures average settings.

C Function Prototype

```
ssa_ConfigureAverage (ViSession vi,
                           ViBoolean AverageEnable,
                           ViInt32 AverageCount,
                           ViInt32 AverageType,
                           ViBoolean AverageDurationEnable,
                           ViReal64 AverageDuration)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession

`AverageEnable`	The average switch	`ViBoolean`
`AverageCount`	The average number	`ViInt32`
`AverageType`	The average type	`ViInt32`
`AverageDurationEnable`	Does not support	`ViBoolean`
`AverageDuration`	Does not support	`ViReal64`

Defined Values for AverageType Parameter

Name	Identifier	value
Log power	SSA_VAL_AVERAGE_TYPE_LOGPOWER	1
Power	SSA_VAL_AVERAGE_TYPE_POWER	2
Voltage	SSA_VAL_AVERAGE_TYPE_VOLTAGE	3

5.1.20 RestartTraceAverage

Description

Configures average settings.

C Function Prototype

```
          ssa_RestartTraceAverage (ViSession vi)
```

Parameters

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

5.1.21 ConfigureMeasurementType

Description

This function selects the measurement type.

Please set the Instrument Mode to Measurement Type to SSA_VAL_INSTRUMENT_MODE_SPECTRUM_ANALYZER when configuring the Measurement

Type.

C Function Prototype

```
        ssa_ConfigureMeasurementType (ViSession vi,
                                                           ViInt32 InstrumentMode,
                                                           ViInt32 MeasurementType)
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle	`ViSession`
`InstrumentMode`	Mode type	`ViInt32`
`MeasurementType`	Measurement type	`ViInt32`

Defined Values for MeasurementType Parameter

Name	Identifier	value
Swept Sa	SSA_VAL_MEASUREMENT_TYPE_SA	0
Channel Power	SSA_VAL_MEASUREMENT_TYPE_CHP	1
ACPR	SSA_VAL_MEASUREMENT_TYPE_ACPR	2
Occupied BW	SSA_VAL_MEASUREMENT_TYPE_OBW	3
T-POWer	SSA_VAL_MEASUREMENT_TYPE_TPOWER	4
TOI	SSA_VAL_MEASUREMENT_TYPE_TOI	5
SPECtrum Monitor	SSA_VAL_MEASUREMENT_TYPE_SM	6
CNR	SSA_VAL_MEASUREMENT_TYPE_CNR	7
Harmonic	SSA_VAL_MEASUREMENT_TYPE_HARM	8

5.1.22 ConfigureACPR

Description

This function configures the Adjacent Channel Power measurement.

C Function Prototype

```
        ssa_ConfigureACPR (ViSession vi,

                                             ViReal64 CenterFrequency,

                                             ViReal64 MainIntergrationBandwidth)
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle	`ViSession`
`CenterFrequency`	Set the center frequency	`ViReal64`
`MainIntergrationBandwidth`	Set the integral bandwidth of the main channel power	`ViReal64`

5.1.23 ReadMeasurementACPR

Description

This function initiates a ACPR measurement acquisition based on the present instrument configuration. It then waits for the acquisition to complete, and returns the ACPR measurement.

Note:

If the spectrum analyzer did not complete the acquisition within the time period the user specified with the maxTime parameter, the function returns Max Time Exceeded error.

C Function Prototype

```
ssa_ReadMeasurementACPR (ViSession vi,
                                ViInt32 MaxTime,
                                ViReal64* LowACP,
                                ViReal64* LowACPR,
                                ViReal64* UpperACP,
                                ViReal64* UpperACPR,
                                ViReal64* MainChannelPower)
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`MaxTime`	Specifies the maximum length of time allowed for the function to complete.	`ViInt32`

Outputs	Description	Base Type
`LowACP`	Lower frequency adjacent channel power	`ViReal64`
`LowACPR`	Lower frequency adjacent channel power ratio	`ViReal64`
`UpperACP`	Higher frequency adjacent channel power	`ViReal64`
`UpperACPR`	Higher frequency adjacent channel power ratio	`ViReal64`
`MainChannelPower`	Main channel power	`ViReal64`

5.1.24 ConfigureCHP

Description

This function configures the channel power settings.

C Function Prototype

```
        ssa_ConfigureCHP (ViSession vi,

                                    ViReal64 CenterFrequency,

                                    ViReal64 IntegrationBandwidth,

                                    ViReal64 ChannelSpan)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession
CenterFrequency	Set the center frequency	ViReal64
IntegrationBandwidth	Setting integral bandwidth	ViReal64
ChannelSpan	Setting span	ViReal64

5.1.25 ReadMeasurementCHP

Description

This function initiates a CHP measurement acquisition based on the present instrument configuration. It then waits for the acquisition to complete, and returns the CHP measurement.

C Function Prototype

```
        ssa_ReadMeasurementCHP (ViSession vi,

                                        ViInt32 MaxTime,

                                        ViReal64* MainChannelPower,

                                        ViReal64* PowerDensity
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession

MaxTime	Specifies the maximum length of time allowed for the function to complete.	ViInt32

Outputs	Description	Base Type
MainChannelPower	Obtain the main channel power	ViReal64
PowerDensity	Obtain the power spectral density of the main channel	ViReal64

5.1.26 ConfigureOBW

Description

This function configures the OBW measurement method, OBW power percentage and the dBc value.

C Function Prototype

```
        ssa_ConfigureOBW (ViSession vi,
                                      ViInt32 OBWMethod,
                                      ViReal64 OBWPowerPercentage,
                                      ViReal64 OBWPowerLevel)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
OBWMethod	Calculation method of occupied power	ViInt32
OBWPowerPercentage	Percentage of occupied power	ViReal64
OBWPowerLevel	Occupied power level	ViReal64

Defined Values for OBWMethod Parameter

Name	Identifier	value
Percent	SSA_VAL_OBW_METHOD_PERCENT	1
dBC	SSA_VAL_OBW_METHOD_DBC	2

5.1.27 ReadMeasurementOBW

Description

This function initiates a OBW measurement acquisition based on the present instrument configuration. It then waits for the acquisitionto complete, and returns the OBW measurement

C Function Prototype

```
ssa_ReadMeasurementOBW (ViSession vi,
                                 ViInt32 MaxTime,
                                 ViReal64* OccupiedBandwidth,
                                 ViReal64* BandwidthCentroid,
                                 ViReal64* XDbBandwidth)
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`MaxTime`	Specifies the maximum length of time allowed for the function to complete.	`ViInt32`

Outputs	Description	Base Type
`OccupiedBandwidth`	occupied bandwidth	`ViReal64`
`BandwidthCentroid`	occupied bandwidth center	`ViReal64`
`XDbBandwidth`	occupied bandwidth x dB	`ViReal64`

5.1.28 ConfigureDemodulation

Description

Configures marker demodulation settings.

C Function Prototype

```
    ssa_ConfigureDemodulation (ViSession vi,
                                                    ViInt32 DemodulationFunction,
                                                    ViReal64 SpeakerVolume,
                                                    ViReal64 DelayTime)
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle	`ViSession`
`DemodulationFunction`	Demodulation type	`ViInt32`
`SpeakerVolume`	Demodulation volume	`ViReal64`
`DelayTime`	Demodulation time	`ViReal64`

Defined Values for DemodulationFunction Parameter

Name	Identifier	value
AM	SSA_VAL_MARKER_DEMODULATION_FUNCTION_AM	1
FM	SSA_VAL_MARKER_DEMODULATION_FUNCTION_FM	2

5.2 Trace

5.2.1 SetActiveTrace

Description

This function selects one of the available traces, and makes it the active trace.

C Function Prototype

```
        ssa_SetActiveTrace (ViSession vi,

                                           ViConstString ActiveTrace)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
ActiveTrace	Trace name like “TRACE1”	ViConstString

5.2.2 ConfigureTrace

Description

Configures trace settings. If TraceMathFunctionEnable is set to VI_FALSE,TraceMathType will have no function.

C Function Prototype

```
         ssa_ConfigureTrace (ViSession vi,

                                          ViBoolean TraceMathFunctionEnable,

                                          ViInt32 TraceMathType)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
TraceMathFunctionEnable	Trace mathematical function switch	ViBoolean
TraceMathType	Trace mathematical calculation type	ViInt32

Defined Values for TraceMathType Parameter

Name	Identifier	value

Off	SSA_VAL_TRACE_MATH_TYPE_OFF	1
Power diff	SSA_VAL_TRACE_MATH_TYPE_POWER_DIFF	2
Power sum	SSA_VAL_TRACE_MATH_TYPE_POWER_SUM	3
Log offset	SSA_VAL_TRACE_MATH_TYPE_LOG_OFFSET	4
Log diff	SSA_VAL_TRACE_MATH_TYPE_LOG_DIFF	5

5.2.3 CopyTrace

Description

Copies the data array from one trace into another trace. Any data in the Destination Trace is deleted.

C Function Prototype

```
ssa_CopyTrace (ViSession vi,

                          ViConstString DestinationTrace,

                          ViConstString SourceTrace)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle	ViSession
DestinationTrace	Target trace	ViConstString
SourceTrace	Source of trace	ViConstString

5.2.4 ExchangeTraces

Description

Exchanges the data arrays of two traces.

C Function Prototype

```
        ssa_ExchangeTraces (ViSession vi,

                                        ViConstString Trace1,

                                        ViConstString Trace2)
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle	`ViSession`
`Trace1`	Source of trace	`ViConstString`
`Trace2`	Target trace	`ViConstString`

5.3 Marker

5.3.1 ConfigureMarkerEnabled

Description

This function enables the active marker on the specified Trace.

C Function Prototype

```
        ssa_ConfigureMarkerEnabled (ViSession Vi,
                                                ViBoolean MarkerEnabled,
                                                ViConstString MarkerTraceName);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`MarkerEnabled`	Enables or disables the active marker. The driver uses this value to set the Marker Enabled attribute. See the attribute description for more details.	`ViBoolean`
`MarkerTraceName`	Specifies the trace name. The driver uses this value to set the Marker Trace attribute. See the attribute description for more details.	`ViConstString`

5.3.2 ConfigureMarkerFrequencyCounter

Description

This function sets the marker frequency counter resolution and enables or disables the marker frequency counter.

C Function Prototype

```
ssa_ConfigureMarkerFrequencyCounter (ViSession Vi,
                                                ViBoolean Enabled,
                                                ViReal64 Resolution);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`Enabled`	Enables or disables the marker frequency counter. The driver uses this value to set the Marker Frequency Counter Enabled attribute. See the attribute description for more details.	`ViBoolean`
`Resolution`	Specifies the frequency counter resolution in Hertz. This value is ignored when Enabled is False. The driver uses this value to set the Marker Frequency Counter Resolution attribute. See the attribute description for more details.	`ViReal64`

5.3.3 ConfigureMarkerSearch

Description

This function configures the Peak Excursion and Marker Threshold attribute values.

C Function Prototype

```
ssa_ConfigureMarkerSearch (ViSession Vi,
                                   ViReal64 PeakExcursion,
                                   ViReal64 MarkerThreshold);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`

PeakExcursion	Minimum amplitude variation of the signal that the marker can recognize as a peak in dB. The driver uses this value to set the Peak Excursion attribute. See the attribute description for more details.	ViReal64
MarkerThreshold	Minimum amplitude below which a peak will not be detected. The driver uses this value to set the Marker Threshold attribute. See the attribute description for more details.	ViReal64

5.3.4 ConfigureMarkerPeakSearch

Description

Configures peak search settings

C Function Prototype

```
ssa_ConfigureMarkerPeakSearch (ViSession vi,

                        ViBoolean MarkerContinuousPeakingEnable,

                         ViInt32 PeakSearchMode)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession
MarkerContinuousPeakingEnable	Peak continuous switch.	ViReal64
PeakSearchMode	Max peak or min peak	ViReal64

5.3.5 ConfigureSignalTrackEnabled

Description

If set to True , the active marker is enabled. When False, the active marker is disabled.

C Function Prototype

```
ssa_ConfigureSignalTrackEnabled (ViSession Vi,
                                        ViBoolean SignalTrackEnabled);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`SignalTrackEnabled`	If set to True , the active marker is enabled. When False, the active marker is disabled. The driver uses this value to set the Signal Track Enabled attribute. See the attribute description for more details.	`ViBoolean`

5.3.6 DisableAllMarkers

Description

This function disables all markers.

C Function Prototype

```
ssa_DisableAllMarkers (ViSession Vi);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`

5.3.7 GetMarkerName

Description

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

C Function ProtoType

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

Parameters

Inputs	Description	Base Type
`Vi`	Unique identifier for an IVI session	`ViSession`
`Index`	An index	`ViInt32`

`NameBufferSize`	Specifies the number of bytes in the `ViChar` array referenced by the `Name` parameter.	`ViInt32`

Outputs	Description	Base Type
`Name (C/COM)`	Specifies the buffer into which the function returns the name that corresponds to the index the user specifies. The caller may pass `VI_NULL` for this parameter if the `NameBufferSize` parameter is 0.	`ViChar[]`

5.3.8 MarkerSearch

Description

This function specifies the type of marker search and performs the search. This function returns the Marker Not Enabled error if the Marker Enabled attribute is set to False.

C Function Prototype

```
ssa_MarkerSearch (ViSession Vi,
                                        ViInt32 SearchType);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`SearchType`	Specifies the type of marker search.	`ViInt32`

Defined Values for SearchType Parameter

Name	Attribute Value Defines	value
Highest	SSA_VAL_MARKER_SEARCH_HIGHEST	1
Left peak	SSA_VAL_MARKER_SEARCH_NEXT_PEAK_LEFT	2
Right peak	SSA_VAL_MARKER_SEARCH_NEXT_PEAK_RIGHT	3
Next peak	SSA_VAL_MARKER_SEARCH_NEXT_PEAK	4

5.3.9 MoveMarker

Description

This function specifies the frequency in Hertz or time position in seconds of the specified horizontal position.

C Function Prototype

```
ssa_MoveMarker (ViSession Vi,
                          ViReal64 MarkerPosition);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession
MarkerPosition	Horizontal position (Hertz or seconds). The driver uses this value to set the Marker Position attribute. See the attribute description for more details.	ViReal64

5.3.10 QueryMarker

Description

This function returns the horizontal position and the amplitude level of the active marker.

C Function Prototype

```
ssa_QueryMarker(ViSession Vi,
                          ViReal64 *MarkerPosition,
                          ViReal64 *MarkerAmplitude);
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession

Outputs	Description	Base Type
MarkerPosition	The frequency in Hertz or time position in seconds of the active marker (depending on the mode in which the analyzer is operating, frequency or time-domain). See the Marker Position attribute description for more details.	ViReal64
MarkerAmplitude	The amplitude of the active marker. The units are specified by the Amplitude Units attribute, except when the Marker Type attribute is set to Delta. Then the units are dB. See the Marker Amplitude attribute description for more details.	ViReal64

5.3.11 SetActiveMarker

Description

This function selects one of the available markers, and makes it the active marker.

C Function Prototype

```
ssa_SetActiveMarker (ViSession Vi,
                                        ViConstString ActiveMarker);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`ActiveMarker`	Marker to be selected. The driver uses this value to set the Active Marker attribute. See the attribute description for more details.	`ViConstString`

5.3.12 SetInstrumentFromMarker

Description

This function uses the Marker Position or Marker Amplitude attributes to configure the spectrum analyzer setting specified by the InstrumentSetting parameter.

This function may set the Frequency Start, Frequency Stop, or Reference Level attributes.

If the Marker Enabled attribute is set to False, this function returns the Marker Not Enabled error. If the Marker Type attribute is not Delta and the `InstrumentSetting` parameter is Frequency Span, the function returns the Delta Marker Not Enabled error.

C Function Prototype

```
ssa_SetInstrumentFromMarker (ViSession Vi,
                                        ViInt32 InstrumentSetting);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`InstrumentSetting`	Specifies the instrument setting to be set from the marker position.	`ViInt32`

Defined Values for InstrumentSetting Parameter

Name	Attribute Value Defines	value
Marker to center	SSA_VAL_INSTRUMENT_SETTING_FREQUENCY_CENTER	1
Marker to start	SSA_VAL_INSTRUMENT_SETTING_FREQUENCY_START	2
Marker to stop	SSA_VAL_INSTRUMENT_SETTING_FREQUENCY_STOP	3

5.3.13 ConfigureMarkerType

Description

This function selects the type of markers that you want to activate.

Notes:

The user must call ssa_SetActiveMarker function ahead to specify the active marker before calling this function.

C Function Prototype

```
      ssa_ConfigureMarkerType (ViSession vi,

                                                    ViInt32 MarkerType,

                                                    ViInt32 MarkerFunction)
```

Parameters

Inputs	Description	Base Type
Vi	Instrument handle.	ViSession
MarkerType	Marker type	ViInt32
MarkerFunction	Marker function	ViInt32

Defined Values for MarkerType Parameter

Name	Identifier	value
Normal	SSA_VAL_MARKER_TYPE_NORMAL	1
Delta	SSA_VAL_MARKER_TYPE_DELTA	2

Defined Values for MarkerFunction Parameter

Name	Identifier	value
Noisy	SSA_VAL_MARKER_FUNCTION_NOISE	1
Ndb	SSA_VAL_MARKER_FUNCTION_NDB	2
Off	SSA_VAL_MARKER_FUNCTION_OFF	3

5.3.14 QueryMarkerType

Description

This function returns the type of the active marker.

C Function Prototype

```
        ssa_QueryMarkerType (ViSession vi,
                                               ViInt32* MarkerType)
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`

Outputs	Description	Base Type
`MarkerType`	Get marker type	`ViInt32`

5.3.15 MakeMarkerDelta

Description

This function specifies whether the active marker is a delta marker. If the current active marker is not enabled then this function enables the active marker.

C Function Prototype

```
            ssa_MakeMarkerDelta (ViSession vi,
                                                    ViBoolean DeltaMarker)
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`DeltaMarker`	Set the cursor to delta	**`ViBoolean`**

5.4 Trigger

5.4.1 Configure Trigger Source

Description

This function specifies the trigger source that causes the spectrum analyzer to leave the *Wait-for-Trigger* state.

C Function Prototype

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

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`TriggerSource`	Specifies the trigger source that causes the analyzer to leave the *Wait-For-Trigger* state. The driver uses this value to set the Trigger Source attribute. See the attribute description for more details.	`ViInt32`

5.4.2 Configure Video Trigger

Description

This function specifies at which level and slope of the video signal, acquisition is triggered. This is applicable when the Trigger Source attribute is set to Video.

C Function Prototype

```
ViStatus IviSpecAn_ConfigureVideoTrigger (ViSession Vi,
                                         ViReal64 VideoTriggerLevel,
                                         ViInt32 VideoTriggerSlope);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`VideoTriggerLevel`	Specifies the level that the video signal shall reach to trigger the acquisition. The driver uses this value to set the Video Trigger Level attribute. See the attribute description for more details.	`ViReal64`

`VideoTriggerSlope`	Specifies which slope of the video signal triggers the acquisition. The driver uses this value to set the Video Trigger Slope attribute. See the attribute description for more details.	`ViInt32`

5.4.3 Configure External Trigger

Description

This function specifies at which level and slope of the external trigger signal, acquisition is triggered. This is applicable when the Trigger Source attribute is set to External.

C Function Prototype

```
ViStatus IviSpecAn_ConfigureExternalTrigger (ViSession Vi,
                                     ViReal64 ExternalTriggerLevel,
                                     ViInt32 ExternalTriggerSlope);
```

Parameters

Inputs	Description	Base Type
`Vi`	Instrument handle.	`ViSession`
`ExternalTriggerLevel`	Specifies the level, in volts, that the external trigger signal shall reach to trigger the acquisition. The driver uses this value to set the External Trigger Level attribute. See the attribute description for more details.	`ViReal64`
`ExternalTriggerSlope`	Specifies which slope of the external trigger signal triggers the acquisition. The driver uses this value to set the External Trigger Slope attribute. See the attribute description for more details.	`ViInt32`