AV4051系列信号/频谱分析仪
驱动库函数手册

中电科仪器仪表有限公司

二零一六年五月

目录

第一节 功能函数说明 8

1.1 环境搭建 8

1.2 使用说明 8

第二节 驱动库函数 9

Init 子系统 9

AVSA_init 9

AVSA_InitWithOptions 9

AVSA_close 10

ANALOGDEMOD 子系统 11

- AVSA_ANALOGDEMOD_FREQUENCY_CENTER 11
- AVSA_AnalogDemod_ AmplitudeReferenceLevelConfigure 11
- AVSA_AnalogDemod_ AmplitudeReferenceLevelQuery 11
- AVSA_AnalogDemod_ AmplitudeAttenuationConfigure 12
- AVSA_AnalogDemod_ SpanConfigure 12
- AVSA_AnalogDemod_ RFBandWidthConfigure 13
- AVSA_AnalogDemod_ AFBandWidthConfigure 13
- AVSA_AnalogDemod_ SweepPointsConfigure 14
- AVSA_AnalogDemod_ MeasurementAverageCountConfigure 14
- AVSA_AnalogDemod_ MeasurementAverageCountQuery 14
- AVSA_AnalogDemod_ MeasurementAverageModeConfigure 15
- AVSA_AnalogDemod_ MeasurementAverageModeQuery 15
- AVSA_AnalogDemod_ MeasurementHighPassConfigure 15
- AVSA_AnalogDemod_ MeasurementLowPassConfigure 16
- AVSA_AnalogDemod_ MeasurementDEMPhasisConfigure 16
- AVSA_AnalogDemod_ MeasurementControlContinuoushConfigure 17

BASIC 子系统 17

- AVSA_BASIC_FREQUENCY_CENTER 18
- AVSA_Basic_AmplitudeSetReferenceLevel 18
- AVSA_Basic_AmplitudeGetReferenceLevel 18
- AVSA_Basic_AmplitudeConfigureAttenuation 19
- AVSA_Basic_SpanConfigure 19
- AVSA_Basic_BandWidthConfigure 20
- AVSA_Basic_SweepPointsConfigure 20
- AVSA_Basic_MeasurementAverageCountConfigure 21
- AVSA_Basic_MeasurementAverageCountQuery 21
- AVSA_Basic_MeasurementAverageModeConfigure 21
- AVSA_Basic_MeasurementAverageModeQuery 22

- AVSA_Basic_MeasurementSweepTimeConfigure 22
- AVSA_Basic_MeasurementSweepTimeQuery 23
- AVSA_Basic_MeasurementSweepLengthConfigure 23
- AVSA_Basic_MeasurementSweepLengthQuery 24
- AVSA_Basic_MeasurementControlContinuoushConfigure 24

Configuration 子系统 24

- AVSA_ConfigureInstModeType 24
- AVSA_ ConfigureFrequencyStartStop 25
- AVSA_ ConfigureFrequencyCenterSpan 25
- AVSA_ ConfigureFrequencyOffset 26
- AVSA_ ConfigureSweepCoupling 26
- AVSA_ ConfigureAcquisition 27
- AVSA_ ConfigureLevel 28
- AVSA_ GetTraceName 29
- AVSA_ ConfigureTraceType 29
- AVSA_ AddTraces 30
- AVSA_ ExchangeTraces 31
- AVSA_ CopyTrace 32
- AVSA_ SubtractTraces 32
- AVSA_ ConfigureMarkerEnabled 33
- AVSA_ ConfigureMarkerFrequencyCounterEnabled 34
- AVSA_ ConfigureMarkerFrequencyCounterResolution 34
- AVSA_ ConfigureMarkerSearch 35
- AVSA_ ConfigureSignalTrackEnabled 35
- AVSA_ DisableAllMarkers 36
- AVSA_ MarkerSearch 36
- AVSA_ MoveMarker 37
- AVSA_ QueryMarker 37
- AVSA_ SetActiveMarker 37
- AVSA_ SetInstrumentFromMarker 38
- AVSA_ QueryMarkerType 38
- AVSA_ MakeMarkerDelta 39
- AVSA_ GetMarkerName 39
- AVSA_ QueryReferenceMarker 40
- AVSA_ ConfigureTriggerSource 40
- AVSA_ ConfigureExternalTrigger 41
- AVSA_ ConfigureExternal2Trigger 41
- AVSA_ ConfigureVideoTrigger 42

SA 子系统.....42
- AVSA_ConfigureMeasurementType 42
- AVSA_ ConfigureCoupleAll 43
- AVSA_ ConfigureBandRatio.....43
- AVSA_ GetBandRatio.....44
- AVSA_ ConfigureVideoRatio.....44
- AVSA_ GetVideoRatio 45
- AVSA_ ConfigureFilterShape.....45
- AVSA_ GetFilterShape.....46
- AVSA_ ConfigureFilterBand.....46
- AVSA_ GetFilterBand.....47
- AVSA_ ConfigureFrequencyStep.....47
- AVSA_ GetFrequencyStep.....47
- AVSA_ ConfigureFrequencySpan.....48
- AVSA_ GetFrequencySpan.....48
- AVSA_ ConfigureFrequencyFullSpan 49
- AVSA_ ConfigureFrequencyPrevSpan 49
- AVSA_ ConfigureFrequencyZeroSpan.....49
- AVSA_ ConfigureSweepType 50
- AVSA_ GetSweepType.....50
- AVSA_ ConfigureSweepTypeRules 51
- AVSA_ GetSweepTypeRules 51
- AVSA_ ConfigureFFTWidth.....52
- AVSA_ GetFFTWidth.....52
- AVSA_ ConfigureNDB.....53
- AVSA_ GetNDB.....53
- AVSA_ ConfigurePeakSort.....54
- AVSA_ GetPeakSort.....54
- AVSA _ ConfigurePeakReadOut.....55
- AVSA_ GetPeakReadOut.....55
- AVSA_ ConfigurePeakTrack.....56
- AVSA_ GetPeakTrack.....56
- AVSA_ ConfigurePeakPeak.....57
- AVSA_ ConfigureDispalyGrid 57
- AVSA_ GetDispalyGrid 57
- AVSA_ ConfigureDispalyLine 58
- AVSA_ GetDispalyLine 58
- AVSA_ ConfigureDispalyAnnotation 59

- AVSA_ GetDispalyAnnotation....59
- AVSA_ ConfigureDispalyFREQuency....60
- AVSA_ GetDispalyFREQuency....60
- AVSA_ ConfigureLimitLineDIspaly....60
- AVSA_ GetLimitLineDIspaly....61
- AVSA_ ConfigureLimitLineType....61
- AVSA_ GetLimitLineType....62
- AVSA_ConfigureLimitLineTrace....62
- AVSA_GetLimitLineTrace....63
- AVSA_ ConfigureLimitLineProperty....63
- AVSA_ GetLimitLineProperty....64
- AVSA_ ConfigureLimitLineMargin....65
- AVSA_ GetLimitLineMargin....66
- AVSA_ ConfigureLimitLineTestSwitch....66
- AVSA_ GetLimitLineTestSwitch....67
- AVSA_ ConfigureLimitLineControlDomain....67
- AVSA_ GetLimitLineControlDomain....68
- AVSA_ DeleteAllLimitLine....68
- AVSA_ SaveLimitLine....68
- AVSA_ LoadLimitLine....69
- AVSA_ ConfigureCouplingType....69
- AVSA_ GetCouplingType....70
- AVSA_ ConfigureFEED....70
- AVSA_ GetFEED....71
- AVSA_ ConfigureFEEDAreference....71
- AVSA_ GetFEEDAreference....72
- AVSA_ ConfigureROSCillatorSource....72
- AVSA_ GetROSCillatorSource....73
- AVSA_ ConfigureAUXOutput....73
- AVSA_ GetAUXOutput....74
- AVSA_ ConfigureSIF....74
- AVSA_ GetSIF....74
- AVSA_ ConfigureAIF....75
- AVSA_ GetAIF....75
- AVSA_ ConfigureIFAUTOGainSwitch....75
- AVSA_ GetIFAUTOGainSwitch....76
- AVSA_ ConfigureIFAUTOGainLevel....76
- AVSA_ GetIFAUTOGainLevel....77

- AVSA_ ConfigureMarkerCouple 77
- AVSA_ GetMarkerCouple 78
- AVSA_ ConfigureMarkerCountSwitch 78
- AVSA_ GetMarkerCountSwitch 79
- AVSA_ ConfigureAudioDemulaitonType 79
- AVSA_ GetAudioDemulaitonType 80
- AVSA_ ConfigureAudioDemulaitonMode 80
- AVSA_ GetAudioDemulaitonMode 81
- AVSA_ ConfigureAudioDemulaitonTime 81
- AVSA_ GetAudioDemulaitonTime 82
- AVSA_ ConfigureBurstTrigger 82
- AVSA_ GetBurstTrigger 82
- AVSA_ ConfigureFRAMTrigger 83
- AVSA_ GetFRAMTrigger 84
- AVSA_ ConfigureFRAMTriggerPeriod 85
- AVSA_ GetFRAMTriggerPeriod 85
- AVSA_ ConfigureFRAMTriggerOffset 85
- AVSA_ GetFRAMTriggerOffset 86
- AVSA_ ConfigureFRAMTriggerHoldOff 86
- AVSA_ GetFRAMTriggerHoldOff 87
- AVSA_ ConfigureVideoTriggerDelay 87
- AVSA_ GetVideoTriggerDelay 87
- AVSA_ ConfigureExternal1TriggerDelay 88
- AVSA_ GetExternal1TriggerDelay 88
- AVSA_ ConfigureExternal2TriggerDelay 89
- AVSA_ GetExternal2TriggerDelay 89
- AVSA_ ConfigureBurstTriggerDelay 90
- AVSA_ GetBurstTriggerDelay 90
- AVSA_ ConfigureADCDither 91
- AVSA_ GetADCDither 91
- AVSA_ ConfigureAverageCount 92
- AVSA_ GetAverageCount 92
- AVSA_ ConfigureAverageType 93
- AVSA_ GetAverageType 93
- AVSA_ ConfigurePhaseNoiseOpt 94
- AVSA_ GetPhaseNoiseOpt 94
- AVSA_ ConfigurePreselectorSwitch 95
- AVSA_ GetPreselectorSwitch 95

- AVSA_ ConfigureMeasureControl....95

Measurement 子系统....96

- AVSA_ Initiate....96
- AVSA_ Abort....96
- AVSA_ ReadYTrace....97
- AVSA_ FetchYTrace....97

PNoise 子系统....98

- AVSA_ PNoise_FrequencyCarrier....98
- AVSA_ PNoise_FrequencyCarrierSearchExecute....98
- AVSA_ PNoise_FrequencyOffset....99
- AVSA_ PNoise_LogPlotConfigure....99
- AVSA_ PNoise_MarkerConfigure....99
- AVSA_ PNoise_MarkerQuery....100

Utility Functions 子系统....100

- AVSA_ reset....101
- AVSA_ ResetWithDefaults....101
- AVSA_revision_query....101
- AVSA_ InvalidateAllAttributes....102
- AVSA_error_query....102
- AVSA_ GetError....102
- AVSA_ ClearError....103
- AVSA_error_ message....103
- AVSA_ GetNextInterchangeWarning....104
- AVSA_ ResetInterchangeCheck....104
- AVSA_ ClearInterchangeWarnings....105

第一节 功能函数说明

1.1 环境搭建

该驱动函数提供对 AVSA 的编程支持；本驱动程序是 IVI-C 版本因此需要安装的组件或程序为（需要按先后顺序）:IVISharedComponents1.5.1.msi(或以上版本)、ICP410(或以上版本)。注：该驱动函数的运行需要 VISA 和 IVI 库。

运行环境：该驱动函数可运行在 Windows xp 及 win7，32 位系统上。可在 labwindows 和 VC 中编程调用运行。

1.2 使用说明

为了成功使用该模块，必须满足如下条件：

对于 GPIB 仪器驱动：

1、仪器连接了 GPIB。

2、初始化函数中的 GPIB 地址必须与仪器的 GPIB 地址相匹配。

对于局域网(LAN)

1、仪器连接了 LAN

2、初始化函数中的逻辑地址必须与仪器的逻辑地址相匹配。

该驱动程序针对 AVSA 提供了包括通信句柄初始化、仪器复位、信号源状态设置、查询、通信句柄关闭等相关函数。

对于参数为数值型的函数，该类函数仅接受相应的数值，不接受数值对应的单位，因此对于函数中参数的数值都采用国际标准单位（如：时间单位为秒 s，频率单位为 Hz，功率单位为 dBm）；函数说明中为了方便参数范围说明通常都带有单位，如有关时间的单位为 10us~2s，因此在设置数据时应将范围转化为以 s 为单位的数值，即实际的数据范围为 0.00001~2。对于查询函数的返回的值也是上述标准单位下对应的数值。在查询函数中若返回值没有进行详细的说明可参考对应的设置函数中参数的说明。

对于函数的调用，调用成功时函数的返回值为 0；当设置的参数超限时，通常函数不进行相应的设置，函数的返回值为 BFFA0010（即-1074135024）；当函数调用失败时函数的返回值为一负值（非-1074135024）。

AVSA 驱动函数的详细说明如下。

第二节 驱动库函数

Init 子系统

该子系统函数用来初始化和关闭远程控制的通信句柄。子系统函数信息如下：

AVSA_init

【方法原型】

ViStatus _VI_FUNC AVSA_init (ViRsrc resourceName,
ViBoolean IDQuery,
ViBoolean resetDevice,
ViSession *newVi)

【功能】

初始化信号源，创建设备句柄；实现仪器的远程连接，将仪器复位到某一状态。

【参数】

方向	参数名	说明
[in]	resourceName	仪器的 VISA 资源名。
[in]	IDQuery	指示是否查询仪器的 ID 号。设为 VI_FALSE
[in]	resetDevice	指示是否对仪器进行复位操作。设为 VI_FALSE(连接时不复位仪器)VI_TRUE(连接时复位仪器)
[out]	newVi	会话句柄

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

AVSA_InitWithOptions

【方法原型】

ViStatus _VI_FUNC AVSA_InitWithOptions (ViRsrc resourceName,
ViBoolean IDQuery,
ViBoolean resetDevice,
ViConstString optionString,
ViSession *newVi)

【功能】

初始化信号源，创建设备句柄；实现仪器的远程连接，将仪器复位到某一状态。

【参数】

方向	参数名	说明
[in]	resourceName	仪器的 VISA 资源名。
[in]	IDQuery	指示是否查询仪器的 ID 号。设为 VI_FALSE
[in]	resetDevice	指示是否对仪器进行复位操作。设为 VI_FALSE(连接时不复位仪器)VI_TRUE(连接时复位仪器)。
[in]	optionString	初始化使用的可选字符串，该变量用于设置一个会话某些属性的初始状态。 属性如与对应的缺省值如下 RangeCheck VI_TRUE QueryInstrStatus VI_FALSE Cache VI_TRUE Simulate VI_FALSE RecordCoercions VI_FALSE ViBoolean 属性的有效值如下: True: 1, True, or VI_TRUE False: 0, False, or VI_FALSE 因此如果该参数设置为 NULL 时，缺省值为 "Simulate=0,RangeCheck=1,QueryInstrStatus=0,Cache=1，RecordCoercions=0 "。通常使用默认的初始值。
[out]	newVi	会话句柄

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

AVSA_close

【方法原型】

ViStatus _VI_FUNC AVSA_close (ViSession vi)

【功能】

关闭通信连接，将仪器返回到本地状态。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

ANALOGDEMOD 子系统

- AVSA_ANALOGDEMOD_FREQUENCY_CENTER

【方法原型】

ViStatus _VI_FUNC AVSA_ ANALOGDEMOD _FREQUENCY_CENTER (ViSession vi,
ViReal64 CenterFrequency)

【功能】

模拟解调模式下设置中心频率。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	CenterFrequency	中心频率。工厂预设：1GHz。设置范围：最小值为 0Hz，最大值为信号分析仪最高频率上限。默认单位：Hz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ AmplitudeReferenceLevelConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_AmplitudeReferenceLevelConfigure (ViSession vi,
ViReal64 ReferenceLevel);

【功能】

模拟解调模式下设置参考电平。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	ReferenceLevel	参考电平。工厂预设：0dBm。设置范围：最小值-170dBm，最大值为 30dBm。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ AmplitudeReferenceLevelQuery

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_AmplitudeReferenceLevelQuery (ViSession vi,
ViReal64 *ReferenceLevel)

【功能】

模拟解调模式下查询参考电平。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	ReferenceLevel	参考电平。范围：最小值-170dBm，最大值为 30dBm。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ AmplitudeAttenuationConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_AmplitudeAttenuationConfigure (ViSession vi,
ViBoolean AttenuationAuto,
ViReal64 Attenuation)

【功能】

模拟解调模式下设置衰减器。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AttenuationAuto	衰减器自动手动开关。工厂预设：自动。 1 自动 0 手动
[in]	Attenuation	衰减。设置范围：最小值 0dB，最大值为 70dB。 默认单位：dB。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ SpanConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_SpanConfigure (ViSession vi,
ViReal64 Span)

【功能】

模拟解调模式下设置的测量带宽。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Span	测量带宽。设置范围：最小值为 10Hz，最大值为 10Mhz。

		默认单位：Hz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ RFBandWidthConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_RFBandWidthConfigure (ViSession vi,
ViBoolean RFBandWidthAuto,
ViReal64 RFBandWidth)

【功能】

模拟解调模式下调整 RF 频谱图的分辨率带宽。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	RFBandWidthAuto	自动手动开关。工厂预设：自动 1 自动 0 手动
[in]	RFBandWidth	RF 频谱图的分辨率带宽。设置范围：最小值为 1Hz，最大值为 3MHz。默认单位：Hz

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ AFBandWidthConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_AFBandWidthConfigure (ViSession vi,
ViBoolean AFBandWidthAuto,
ViReal64 AFBandWidth)

【功能】

模拟解调模式下调整 AF 频谱图的分辨率带宽。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AFBandWidthAuto	自动手动开关。工厂预设：自动 1 自动 0 手动
[in]	AFBandWidth	AF 频谱图的分辨率带宽。设置范围：最小值为 1Hz，最大值为 3MHz。默认单位：Hz

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ SweepPointsConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_SweepPointsConfigure (ViSession vi,
ViInt32 SweepPoints)

【功能】

模拟解调模式下设置扫描点数。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	SweepPoints	扫描点数。工厂预设：1001。设置范围：101-30001。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ MeasurementAverageCountConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_MeasurementAverageCountConfigure (ViSession vi,
ViInt32 AverageCount)

【功能】

模拟解调模式下设置平均次数。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AverageCount	平均次数。工厂预设：100。设置范围：1-10000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ MeasurementAverageCountQuery

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_MeasurementAverageCountQuery (ViSession vi,
ViInt32 *AverageCount)

【功能】

模拟解调模式下查询平均次数 。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AverageCount	平均次数。范围：1-10000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ MeasurementAverageModeConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_MeasurementAverageModeConfigure (ViSession vi, ViInt32 AverageMode)

【功能】

模拟解调模式下设置平均模式。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AverageMode	平均模式。工厂预设：指数模式。 1 指数模式 2 重复模式

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ MeasurementAverageModeQuery

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_MeasurementAverageModeQuery (ViSession vi, ViInt32 *AverageMode)

【功能】

模拟解调模式下测量结果的平均方式。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AverageMode	平均模式。工厂预设：指数模式。 1 指数模式 2 重复模式

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ MeasurementHighPassConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_MeasurementHighPassConfigure (ViSession vi,
ViInt32 HighPassState)

【功能】

模拟解调模式下设置高通滤波器。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	HighPassState	高通滤波器类型。工厂预设：关。 1 OFF 设置为关闭 2 20Hz 设置为 20Hz 3 50Hz 设置为 50Hz 4 300Hz 设置为 300Hz

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ MeasurementLowPassConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_MeasurementLowPassConfigure (ViSession vi,
ViInt32 LowPassState)

【功能】

模拟解调模式下设置低通滤波器。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LowPassState	低通滤波器类型。工厂预设：关。 1 OFF 设置为关闭 2 300Hz 设置为 300Hz 3 3kHz 设置为 3kHz 4 15kHz 设置为 15kHz 5 30kHz 设置为 30kHz 6 80kHz 设置为 80kHz 7 300kHz 设置为 300kHz

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ MeasurementDEMPhasisConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_MeasurementDEMPhasisConfigure (ViSession vi,
ViInt32 DEMPhasisState)

【功能】

模拟解调模式下设置去加重滤波器。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	DEMPhasisState	去加重滤波器类型。工厂预设：关。 1 OFF 设置为关闭 2 25us 设置为 25us 3 50us 设置为 50us 4 75us 设置为 75us 5 750us 设置为 750us

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_AnalogDemod_ MeasurementControlContinuoushConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_AnalogDemod_MeasurementControlContinuoushConfigure (
ViSession vi,
ViBoolean
ContinuousSwtich)

【功能】

模拟解调模式下设置连续还是单次测量 。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	ContinuousSwtich	连续、单次开关。工厂预设：连续测量。 1 连续 0 单次

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

BASIC 子系统

该子系统函数用于 IQ 模式下操作。

- AVSA_BASIC_FREQUENCY_CENTER

【方法原型】

ViStatus _VI_FUNC AVSA_BASIC_FREQUENCY_CENTER (ViSession vi,
ViReal64 CenterFrequency)

【功能】

IQ 模式下设置中心频率。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	CenterFrequency	中心频率。工厂预设：1GHz。设置范围：最小值为 0Hz，最大值为信号分析仪最高频率上限。默认单位：Hz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_AmplitudeSetReferenceLevel

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_AmplitudeSetReferenceLevel (ViSession vi,
ViInt32 WindowNumber,
ViReal64 ReferenceLevel)

【功能】

IQ 模式下设置参考电平。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	WindowNumber	窗口 ID。
[in]	ReferenceLevel	参考电平。工厂预设：0dBm。设置范围：最小值-170dBm，最大值为 30dBm。默认单位：依据纵轴刻度单位设置。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_AmplitudeGetReferenceLevel

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_AmplitudeGetReferenceLevel (ViSession vi,
ViInt32 WindowNumber,
ViReal64 *ReferenceLevel)

【功能】

IQ 模式下获取参考电平。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	WindowNumber	窗口 ID。
[out]	ReferenceLevel	参考电平。工厂预设：0dBm。设置范围：最小值-170dBm，最大值为 30dBm。默认单位：依据纵轴刻度单位设置。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_AmplitudeConfigureAttenuation

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_AmplitudeConfigureAttenuation (ViSession vi,
ViBoolean AttenuationAuto,
ViReal64 Attenuation)

【功能】

IQ 模式下设置衰减器。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AttenuationAuto	衰减器自动手动开关。工厂预设：自动。 1 自动 0 手动
[in]	Attenuation	衰减。设置范围：最小值 0dB，最大值为 70dB。 默认单位：dB。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_SpanConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_SpanConfigure (ViSession vi,
ViReal64 Span)

【功能】

IQ 模式下设置测量带宽。

【参数】

方向	参数名	说明

[in]	vi	会话句柄。
[in]	Span	测量带宽。工厂预设：最高频率上限-起始频率。设置范围：最小值为 10Hz，最大和仪器配置的选件有关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_BandWidthConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_BandWidthConfigure (ViSession vi,
ViBoolean BandWidthAuto,
ViReal64 BandWidth)

【功能】

IQ 模式下设置带宽。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	BandWidthAuto	带宽自动手动开关。工厂预设：自动。 1 自动 0 手动
[in]	BandWidth	带宽。设置范围：最小值为 1Hz，最大值为 3MHz。 默认单位：Hz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_SweepPointsConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_SweepPointsConfigure (ViSession vi,
ViInt32 SweepPoints)

【功能】

IQ 模式下设置扫描点数。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	SweepPoints	扫描点数。工厂预设：1001。设置范围：101-30001。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_MeasurementAverageCountConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_MeasurementAverageCountConfigure (ViSession vi,
ViInt32 AverageCount)

【功能】

IQ 模式下设置平均次数。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AverageCount	平均次数。工厂预设：100。设置范围：1-10000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_MeasurementAverageCountQuery

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_MeasurementAverageCountQuery (ViSession vi,
ViInt32 *AverageCount)

【功能】

IQ 模式下查询平均次数。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AverageCount	平均次数。范围：1-10000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_MeasurementAverageModeConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_MeasurementAverageModeConfigure (ViSession vi,
ViInt32 AverageMode)

【功能】

IQ 模式下设置平均模式。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

[in]	AverageMode	平均模式。工厂预设：指数模式。 AVSA_VAL_BASIC_MEASUREMENTSET_AVERAGE_MOED_EXP 1　　指数模式 AVSA_VAL_BASIC_MEASUREMENTSET_AVERAGE_MOED_REP 2　　重复模式

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_MeasurementAverageModeQuery

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_MeasurementAverageModeQuery (ViSession vi,
ViInt32 *AverageMode)

【功能】

IQ 模式下查询平均模式。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AverageMode	平均模式。工厂预设：指数模式。 AVSA_VAL_BASIC_MEASUREMENTSET_AVERAGE_MOED_EXP 1　　指数模式 AVSA_VAL_BASIC_MEASUREMENTSET_AVERAGE_MOED_REP 2　　重复模式

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_MeasurementSweepTimeConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_MeasurementSweepTimeConfigure (ViSession vi,
ViBoolean SweepTimeAuto,
ViReal64 SweepTime)

【功能】

IQ 模式下设置扫描时间开关及扫描时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	SweepTimeAuto	扫描时间自动手动开关。工厂预设：自动

		1　自动　0　手动
[in]	SweepTime	扫描时间。工厂预设：测量时间自动 37.76μs。设置范围：最小值:16/(1.25*测量带宽)；最大值：8000000/(1.25*测量带宽)。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_MeasurementSweepTimeQuery

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_MeasurementSweepTimeQuery (ViSession vi,
ViBoolean *SweepTimeAuto,
ViReal64 *SweepTime)

【功能】

IQ 模式下查询扫描时间开关及扫描时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	SweepTimeAuto	扫描时间自动手动开关。工厂预设：自动 1　自动　0　手动
[out]	SweepTime	扫描时间。范围：最小值:16/(1.25*测量带宽)；最大值：8000000/(1.25*测量带宽)。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_MeasurementSweepLengthConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_MeasurementSweepLengthConfigure (ViSession vi,
ViInt32 SweepLength)

【功能】

IQ 模式下设置扫描长度。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	SweepLength	扫描长度。工厂预设：472。设置范围：16-8000000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_MeasurementSweepLengthQuery

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_MeasurementSweepLengthQuery (ViSession vi,
ViInt32 *SweepLength)

【功能】

IQ 模式下查询扫描长度。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	SweepLength	扫描长度。范围：16-8000000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_Basic_MeasurementControlContinuoushConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_Basic_MeasurementControlContinuoushConfigure (ViSession vi,
ViBoolean ContinuousSwtich)

【功能】

IQ 模式下设置单次扫描、连续扫描。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	ContinuousSwtich	1 连续扫描 0 单次扫描

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

Configuration 子系统

该子系统函数用于设置输出射频信号的开关。

- AVSA_ConfigureInstModeType

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureInstModeType (ViSession vi,
ViInt32 instrumentModeType)

【功能】

设置仪器的当前工作模式类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	instrumentModeType	参数： AVSA_VAL_INSTMODE_TYPE_SA 频谱分析 AVSA_VAL_INSTMODE_TYPE_BASIC IQ 分析 AVSA_VAL_INSTMODE_TYPE_PNOISE 相位噪声 AVSA_VAL_INSTMODE_TYPE_ADEMOD 模拟解调

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFrequencyStartStop

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFrequencyStartStop (ViSession vi,
ViReal64 StartFrequency,
ViReal64 StopFrequency)

【功能】

设置仪器的起始、终止频率。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	StartFrequency	最小值为-100MHz，最大值为信号分析仪最高频率上限。
[in]	StopFrequency	最小值为 0Hz，最大值为信号分析仪最高频率上限。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFrequencyCenterSpan

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFrequencyCenterSpan (ViSession vi,
ViReal64 CenterFrequency,
ViReal64 Span)

【功能】

设置中心频率及频宽。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	CenterFrequency	中心频率。最小值为 0Hz，最大值为信号分析仪最高频率上限。
[in]	Span	频宽。最小值为 0Hz，最大值为信号分析仪最高频率上限。 Frequency Start = CenterFrequency - span/2 Frequency Stop = CenterFrequency + span/2

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFrequencyOffset

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFrequencyOffset (ViSession vi,
ViReal64 FrequencyOffset)

【功能】

设置频率偏移。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FrequencyOffset	默认单位 Hz。最小值-500GHz，最大值+500GHz

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureSweepCoupling

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureSweepCoupling (ViSession vi,
ViBoolean ResolutionBandwidthAuto,
ViReal64 ResolutionBandwidth,
ViBoolean VideoBandwidthAuto,
ViReal64 VideoBandwidth,
ViBoolean SweepTimeAuto,
ViReal64 SweepTime);

【功能】

设置扫描相关属性，分辨率带宽自动手动开关、分辨率带宽、视频带宽自动手动开关、视频带、扫描时间自动手动开关、扫描时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	ResolutionBandwidthAuto	分辨率带宽自动手动开关。1　自动 ，0　手动。
[in]	ResolutionBandwidth	分辨率带宽。默认单位 Hz。最小值为 1Hz，最大值为 20MHz
[in]	VideoBandwidthAuto	视频带宽自动手动开关。1　自动，0　手动。
[in]	VideoBandwidth	视频带宽。默认单位 Hz。最小值为 1Hz，最大值为 20MHz
[in]	SweepTimeAuto	扫描时间自动手动开关。1　自动，0　自动。
[in]	SweepTime	扫描时间。默认单位s。非零频宽下最小值为1ms，最大值为6000s；零频宽下最小值为1us，最大值为6000s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureAcquisition

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureAcquisition (ViSession vi,
ViBoolean SweepModeContinuous,
ViInt32 NumberOfSweeps,
ViBoolean DetectorTypeAuto,
ViInt32 DetectorType,
ViInt32 VerticleScale)

【功能】

设置仪器的获取属性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	SweepModeContinuous	设置连续扫描或单次扫描。1 连续，0 单次。
[in]	NumberOfSweeps	扫描点数。工厂预设：1001。设置范围：101-30001。
[in]	DetectorTypeAuto	轨迹检波方式自动手动开关。
[in]	DetectorType	轨迹检波方式。 正常　1 平均(视频)　2

		正峰值 3 取样 4 负峰值 5 平均(电压) 6 平均(功率) 7
[in]	VerticleScale	刻度类型。 LINEAR 1 LOGARITHMIC 2

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureLevel

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureLevel (ViSession vi,
ViInt32 AmplitudeUnits,
ViReal64 InputImpedance,
ViReal64 ReferenceLevel,
ViReal64 ReferenceLevelOffset,
ViBoolean AttenuationAuto,
ViReal64 Attenuation);

【功能】

设置纵轴属性，如 Y 轴刻度单位。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AmplitudeUnits	Y 轴刻度单位 dBm AVSA_VAL_ AMPLITUDE _UNITS_DBM dBmV AVSA_VAL_AMPLITUDE_UNITS_DBMV dBμV AVSA _VAL_AMPLITUDE_UNITS_DBUV V AVSA _VAL_AMPLITUDE_UNITS_VOLT W AVSA_VAL_AMPLITUDE_UNITS_WATT dBmA AVSA_VAL_AMPLITUDE_UNITS_DBMA A AVSA_VAL_AMPLITUDE_UNITS_AMPERE dBμA AVSA _VAL_AMPLITUDE_UNITS_DBUA
[in]	InputImpedance	设置的功率值，支持单位为 dB，取值范围为 -135dB~30dB
[in]	ReferenceLevel	参考电平。工厂预设：0dBm。设置范围：最小值-170dBm，

		最大值为30dBm。默认单位：依据纵轴刻度单位设置。
[in]	ReferenceLevelOffset	参考电平偏移。工厂预设：0dB。设置范围：最小值 -1000dB，最大值为1000dB。
[in]	AttenuationAuto	机械衰减器自动手动开关。默认自动。自动 1 ，手动 0。
[in]	Attenuation	机械衰减。设置范围：最小值 0dB，最大值为 70dB。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetTraceName

【方法原型】

ViStatus _VI_FUNC AVSA_GetTraceName (ViSession vi,
ViInt32 index,
ViInt32 nameBufferSize,
ViChar name[]);

【功能】

获取轨迹名称。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	index	参数。1 到 6。
[in]	nameBufferSize	缓存大小。
[in]	name	轨迹名称。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureTraceType

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureTraceType (ViSession vi,
ViConstString TraceName,
ViInt32 TraceType)

【功能】

设置轨迹类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	TraceName	轨迹名称。

		TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6
[in]	TraceType	轨迹类型。 WRITe 1 AVERage 2 MAXHold 3 MINHold 4

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ AddTraces

【方法原型】

ViStatus _VI_FUNC AVSA_AddTraces (ViSession vi,
ViConstString DestinationTrace,
ViConstString Trace1,
ViConstString Trace2)

【功能】

轨迹相加。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	DestinationTrace	目标轨迹。 TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6
[in]	Trace1	待相加轨迹。 TRACE1 TRACE2 TRACE3 TRACE4

		TRACE5 TRACE6
[in]	Trace2	待相加轨迹。 TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ExchangeTraces

【方法原型】

ViStatus _VI_FUNC AVSA_ExchangeTraces (ViSession vi,
ViConstString Trace1,
ViConstString Trace2)

【功能】

轨迹交换。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Trace1	待交换轨迹号。 TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6
[in]	Trace1	待交换轨迹号。 TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ CopyTrace

【方法原型】

ViStatus _VI_FUNC AVSA_CopyTrace (ViSession vi,
ViConstString DestinationTrace,
ViConstString SourceTrace)

【功能】

轨迹拷贝。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	DestinationTrace	目标轨迹号。 TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6
[in]	SourceTrace	源轨迹号。 TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ SubtractTraces

【方法原型】

ViStatus _VI_FUNC AVSA_SubtractTraces (ViSession vi,
ViConstString DestinationTrace,
ViConstString Trace1,
ViConstString Trace2);

【功能】

轨迹相减。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	DestinationTrace	目标轨迹号。 TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6
[in]	Trace1	被减轨迹号。 TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6
[in]	Trace2	减轨迹号。 TRACE1 TRACE2 TRACE3 TRACE4 TRACE5 TRACE6

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureMarkerEnabled

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureMarkerEnabled (ViSession vi,
ViInt32 MarkerNumber,
ViInt32 MarkerType)

【功能】

设置标记类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	MarkerNumber	标记号。取值范围：1 到 12。
[in]	MarkerType	标记类型。 正常频标　1 差值频标　2 固定频标　3 关闭频标　4

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureMarkerFrequencyCounterEnabled

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureMarkerFrequencyCounterEnabled (ViSession vi,
int MarkerNumber,
ViBoolean MarkerFrequencyCounter);

【功能】

设置标记计数功能。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	MarkerNumber	标记号。取值范围：1 到 12。
[in]	MarkerFrequencyCounter	标记计数功能开关。1　开，0　关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureMarkerFrequencyCounterResolution

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureMarkerFrequencyCounterResolution (ViSession vi,
int MarkerNumber,
ViReal64 FrequencyCounterResolution);

【功能】

设置标记计数门控时间。该函数当标记功能中的标记计数功能打开时，才起作用。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

[in]	MarkerNumber	标记号。取值范围：1 到 12。
[in]	FrequencyCounterResolution	标记计数门控时间。默认单位：s。设置范围：最小值为1.0E-3s，最大值为8s。取值分辨率：100ns。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureMarkerSearch

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureMarkerSearch (ViSession vi,
ViReal64 PeakExcursion,
ViReal64 MarkerThreshold);

【功能】

设置峰值偏移及峰值门限。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	PeakExcursion	峰值偏移。工厂预设：默认为关，表示峰值定义值为 6dB。 设置范围：最小值为 0dB，最大值为 100dB。
[in]	MarkerThreshold	峰值门限。工厂预设：默认为关，表示所有峰值均有效。 设置范围：最小值为-200dBm，最大值为参考电平。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureSignalTrackEnabled

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureSignalTrackEnabled (ViSession vi,
ViBoolean SignalTrackEnabled)

【功能】

设置信号跟踪。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	SignalTrackEnabled	信号跟踪标志。1　开，0　关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ DisableAllMarkers

【方法原型】

ViStatus _VI_FUNC AVSA_DisableAllMarkers(ViSession vi)

【功能】

关闭所有频标。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ MarkerSearch

【方法原型】

ViStatus _VI_FUNC AVSA_MarkerSearch (ViSession vi,
int MarkerNumber,
ViInt32 MarkerSearch);

【功能】

标记搜索。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	MarkerNumber	标记号。取值范围：1 到 12。
[in]	MarkerSearch	标记搜索类型 AVSA_VAL_MARKER_SEARCH_HIGHEST 峰值 AVSA_VAL_MARKER_SEARCH_NEXT_PEAK 次峰值 AVSA_VAL_MARKER_SEARCH_NEXT_PEAK_LEFT 左邻峰值 AVSA_VAL_MARKER_SEARCH_NEXT_PEAK_RIGHT 右邻峰值 AVSA_VAL_MARKER_SEARCH_MINIMUM 最小值

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ MoveMarker

ViStatus _VI_FUNC AVSA_MoveMarker (ViSession vi,
ViInt32 MarkerNumber,
ViReal64 MarkerPosition)

【功能】

移动频标到横轴某位置。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	MarkerNumber	标记号。取值范围：1 到 12。
[in]	MarkerPosition	标记 x 轴位置。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ QueryMarker

【方法原型】

ViStatus _VI_FUNC AVSA_QueryMarker (ViSession vi,
ViInt32 MarkerNumber,
ViReal64 *MarkerPosition,
ViReal64 *MarkerAmplitude)

【功能】

查询频标 X 轴、Y 轴读数。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	MarkerNumber	标记号。取值范围：1 到 12。
[out]	MarkerPosition	标记 x 轴位置。
[out]	MarkerAmplitude	标记 y 轴位置。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ SetActiveMarker

【方法原型】

ViStatus _VI_FUNC AVSA_SetActiveMarker (ViSession vi,
ViConstString ActiveMarker)

【功能】

设置当前激活频标。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	ActiveMarker	当前激活频标号。1 到 12。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ SetInstrumentFromMarker

【方法原型】

ViStatus _VI_FUNC AVSA_SetInstrumentFromMarker (ViSession vi,
ViInt32 InstrumentSetting)

【功能】

设置标记移动。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	InstrumentSetting	标记移动类型 AVSA_VAL_INSTRUMENT_SETTING_FREQUENCY_START 起始频率 AVSA_VAL_INSTRUMENT_SETTING_FREQUENCY_STOP 终止频率 AVSA_VAL_INSTRUMENT_SETTING_FREQUENCY_CENTER 中心频率 AVSA_VAL_INSTRUMENT_SETTING_FREQUENCY_SPAN 频宽 AVSA_VAL_INSTRUMENT_SETTING_REFERENCE_LEVEL Y 轴单位

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ QueryMarkerType

【方法原型】

ViStatus _VI_FUNC AVSA_QueryMarkerType (ViSession vi,
ViInt32 *MarkerType)

【功能】

获取标记类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	MarkerType	标记类型。 1　正常标记 2　差值标记 3　固定标记 4　标记关闭

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ MakeMarkerDelta

【方法原型】

ViStatus _VI_FUNC AVSA_MakeMarkerDelta (ViSession vi,
ViInt32 MarkerNumber,
ViBoolean DeltaMarker)

【功能】

设置为差值频标或正常频标。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	MarkerNumber	频标 ID
[in]	DeltaMarker	设置差值标记标志。1　设置为差值频标，0　设置为正常频标。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetMarkerName

【方法原型】

ViStatus _VI_FUNC AVSA_GetMarkerName (ViSession vi,
ViInt32 Index,
ViInt32 NameBufferSize,
ViChar Name[]);

【功能】

查询频标名称。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Index	参数。1 到 12。
[out]	NameBufferSize	缓存大小。
[out]	Name	频标名称。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ QueryReferenceMarker

【方法原型】

```
ViStatus _VI_FUNC AVSA_QueryReferenceMarker (ViSession vi,
                                             ViInt32 MarkerNumber,
                                             ViReal64 *ReferenceMarkerAmplitude,
                                             ViReal64 *ReferenceMarkerPosition)
```

【功能】

查询参考频标在 Y 轴和 X 轴的位置。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	MarkerNumber	频标号。1 到 12。
[out]	ReferenceMarkerAmplitude	参考标记 Y 轴位置。
[out]	ReferenceMarkerPosition	参考标记 X 轴位置。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureTriggerSource

【方法原型】

```
ViStatus _VI_FUNC AVSA_ConfigureTriggerSource (ViSession vi,
                                               ViInt32 TriggerSource);
```

【功能】

设置触发源。

【参数】

方向	参数名	说明

[in]	vi	会话句柄。
[in]	TriggerSource	触发源。 1 EXTernal1 2 EXTernal2 3 IMMediate 4 LINE 5 FRAMe 6 RFBurst 7 VIDeo

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureExternalTrigger

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureExternalTrigger (ViSession vi,
ViReal64 ExternalTriggerLevel,
ViInt32 ExternalTriggerSlope);

【功能】

设置外部触发源 1 的触发电平及触发极性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	ExternalTriggerLevel	触发电平。工厂预设：1.2V。设置范围：-5V-5Vs。默认单位：V。
[in]	ExternalTriggerSlope	触发极性。 AVSA_VAL_TRIGGER_SLOPE_POSITIVE 正 AVSA_VAL_TRIGGER_SLOPE_NEGATIVE 负

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureExternal2Trigger

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureExternal2Trigger (ViSession vi,
ViReal64 External2TriggerLevel,
ViInt32 External2TriggerSlope);

【功能】

设置外部触发 2 的触发电平及触发极性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	External2TriggerLevel	触发电平。工厂预设：1.2V。设置范围：-5V-5Vs。 默认单位：V。
[in]	External2TriggerLevel	触发极性。 AVSA_VAL_TRIGGER_SLOPE_POSITIVE 正 AVSA_VAL_TRIGGER_SLOPE_NEGATIVE 负

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureVideoTrigger

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureVideoTrigger (ViSession vi,
ViReal64 VideoTriggerLevel,
ViInt32 VideoTriggerSlope)

【功能】

设置视频触发的触发电平及触发极性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	VideoTriggerLevel	视频触发电平。工厂预设：-25dBm。设置范围： -270dBm-30dBm。 默认单位：dBm
[in]	VideoTriggerSlope	视频触发极性。 AVSA_VAL_TRIGGER_SLOPE_POSITIVE 正 AVSA_VAL_TRIGGER_SLOPE_NEGATIVE 负

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

SA 子系统

- AVSA_ConfigureMeasurementType

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureMeasurementType (ViSession vi,

ViInt32 measurementType)

【功能】

设置频谱分析模式下当前的测量类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	measurementType	参数： AVSA_VAL_MEASURE_TYPE_SA 扫频分析 AVSA_VAL_MEASURE_TYPE_OBW 占用带宽 AVSA_VAL_MEASURE_TYPE_CHP 信道功率 AVSA_VAL_MEASURE_TYPE_ACP 邻道功率 AVSA_VAL_MEASURE_TYPE_PST 功率统计 AVSA_VAL_MEASURE_TYPE_BP 突发功率 AVSA_VAL_MEASURE_TYPE_SPUR 杂散发射 AVSA_VAL_MEASURE_TYPE_SEM 频谱发射模板 AVSA_VAL_MEASURE_TYPE_TOI 三阶交调 AVSA_VAL_MEASURE_TYPE_HARM 谐波失真

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureCoupleAll

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureCoupleAll (ViSession vi);

【功能】

关联所有。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureBandRatio

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureBandRatio (ViSession vi,

ViBoolean BandRatioAuto,

ViInt32 BandRatio);

【功能】

设置当前频宽和分辨率带宽的比率及自动手动开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	BandRatioAuto	频宽和分辨率带宽的比率自动手动开关。工厂预设：自动。1 自动， 0 手动。
[in]	BandRatio	频宽和分辨率带宽的比率。工厂预设：频宽/RBW 等于 106。设置范围：最小值为 2，最大值为 10000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetBandRatio

【方法原型】

ViStatus _VI_FUNC AVSA_GetBandRatio (ViSession vi,
ViBoolean *BandRatioAuto,
ViInt32 *BandRatio)

【功能】

查询当前频宽和分辨率带宽的比率及自动手动开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	BandRatioAuto	频宽和分辨率带宽的比率自动手动开关。
[out]	BandRatio	频宽和分辨率带宽的比率。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureVideoRatio

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureVideoRatio (ViSession vi,
ViBoolean videoRatioAuto,
ViReal64 videoRatio)

【功能】

设置当前视频带宽和分辨率带宽的比值及自动手动开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	videoRatioAuto	当前视频带宽和分辨率带宽的比值自动手动开关。工厂预设：自动。1　自动，　0　手动。
[in]	videoRatio	当前视频带宽和分辨率带宽的比值。工厂预设：VBW/RBW 等于 1。设置范围：最小值为 0.0001，最大值为 300000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetVideoRatio

【方法原型】

ViStatus _VI_FUNC AVSA_GetVideoRatio (ViSession vi,
ViBoolean *videoRatioAuto,
ViReal64 *videoRatio);

【功能】

查询当前视频带宽和分辨率带宽的比值。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	videoRatioAuto	当前视频带宽和分辨率带宽的比值自动手动开关。
[out]	videoRatio	当前视频带宽和分辨率带宽的比值。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFilterShape

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFilterShape (ViSession vi,
ViInt32 FilterShape);

【功能】

设置滤波器类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FilterShape	滤波器类型。工厂预设：高斯。 AVSA_VAL_FILTER_SHAPE_GAUSSIAN　1

		AVSA_VAL_FILTER_SHAPE_FLATTOP 2

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFilterShape

【方法原型】

ViStatus _VI_FUNC AVSA_GetFilterShape (ViSession vi,
ViInt32 *FilterShape)

【功能】

查询滤波器类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FilterShape	滤波器类型。 AVSA_VAL_FILTER_SHAPE_GAUSSIAN 1 AVSA_VAL_FILTER_SHAPE_FLATTOP 2

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFilterBand

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFilterBand (ViSession vi,
ViInt32 FilterBand)

【功能】

设置滤波器带宽。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FilterBand	滤波器带宽。 AVSA_VAL_FILTER_BAND_DB3 1 AVSA_VAL_FILTER_BAND_DB6 2 AVSA_VAL_FILTER_BAND_IMPulse 3 AVSA_VAL_FILTER_BAND_NOISe 4

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFilterBand

【方法原型】

ViStatus _VI_FUNC AVSA_GetFilterBand (ViSession vi,
ViInt32 *FilterBand)

【功能】

查询滤波器带宽。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FilterBand	滤波器带宽。 AVSA_VAL_FILTER_BAND_DB3 1 AVSA_VAL_FILTER_BAND_DB6 2 AVSA_VAL_FILTER_BAND_IMPulse 3 AVSA_VAL_FILTER_BAND_NOISe 4

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFrequencyStep

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFrequencyStep (ViSession vi,
ViBoolean FrequencyStepAuto,
ViReal64 FrequcenyStep)

【功能】

设置频率步进。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FrequencyStepAuto	频率步进自动手动开关。1 自动，0 手动。
[in]	FrequcenyStep	频率步进。工厂预设：取决于信号分析仪最高频率上限和当前的测量模式。设置范围：最小值为 1Hz，最大值为信号分析仪最高频率上限。 默认单位：Hz

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFrequencyStep

【方法原型】

ViStatus _VI_FUNC AVSA_GetFrequencyStep (ViSession vi,
ViBoolean *FrequencyStepAuto,
ViReal64 *FrequcenyStep)

【功能】

查询频率步进。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FrequencyStepAuto	频率步进自动手动开关。1 自动，0 手动。
[out]	FrequcenyStep	频率步进。工厂预设：取决于信号分析仪最高频率上限和当前的测量模式。范围：最小值为 1Hz，最大值为信号分析仪最高频率上限。 默认单位：Hz

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFrequencySpan

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFrequencySpan (ViSession vi,
ViReal64 FrequencySPAN)

【功能】

设置频宽。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FrequencySPAN	频宽。工厂预设：最高频率上限-起始频率。设置范围：最小值 0Hz，非零频宽最小值为 10Hz，最大和仪器最高工作频率上限有关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFrequencySpan

【方法原型】

ViStatus _VI_FUNC AVSA_GetFrequencySpan (ViSession vi,
ViReal64 *FrequencySpan)

【功能】

查询频率频宽。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FrequencySpan	频宽。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFrequencyFullSpan

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFrequencyFullSpan (ViSession vi)

【功能】

设置信号分析仪为中心频率/频宽模式，同时将频宽置为最大值。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFrequencyPrevSpan

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFrequencyPrevSpan (ViSession vi)

【功能】

设置信号分析仪为中心频率/频宽模式，同时将频宽置为前一次的值。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFrequencyZeroSpan

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFrequencyZeroSpan (ViSession vi)

【功能】

将频宽设置为 0Hz。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureSweepType

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureSweepType (ViSession vi,
ViBoolean SweepTypeAuto,
ViInt32 SweepType)

【功能】

设置扫描类型及自动手动开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	SweepTypeAuto	扫描类型自动手动开关。 1 自动　0 手动。
[in]	SweepType	扫描类型 AVSA_VAL_SWEEP_TYPE_FFT　1 AVSA_VAL_SWEEP_TYPE_SWEEP　2

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetSweepType

【方法原型】

ViStatus _VI_FUNC AVSA_GetSweepType (ViSession vi,
ViBoolean *SweepTypeAuto,
ViInt32 *SweepType)

【功能】

查询扫描类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	SweepTypeAuto	扫描类型自动手动开关。 1 自动　0 手动。

[out	SweepType	扫描类型 AVSA_VAL_SWEEP_TYPE_FFT 1 AVSA_VAL_SWEEP_TYPE_SWEEP 2

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureSweepTypeRules

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureSweepTypeRules (ViSession vi,
ViBoolean SweepAutoRuleAuto,
ViInt32 SweepTypeAutoRule)

【功能】

设置扫描类型规则及自动手动开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	SweepAutoRuleAuto	扫描类型规则自动手动开关。 1 自动 0 手动。
[in]	SweepTypeAutoRule	扫描类型规则。工厂预设：扫描类型规则自动设置成“动态范围优先”规则。 AVSA_VAL_SWEEP_TYPE_AUTO_RULE_SPEED 1 AVSA_VAL_SWEEP_TYPE_AUTO_RULE_DRANGE 2

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetSweepTypeRules

【方法原型】

ViStatus _VI_FUNC AVSA_GetSweepTypeRules (ViSession vi,
ViBoolean *SweepTypeAutoRuleAuto,
ViInt32 *SweepTypeAutoRule)

【功能】

查询扫描类型规则。

【参数】

方向	参数名	说明

[in]	vi	会话句柄。
[out]	SweepAutoRuleAuto	扫描类型规则自动手动开关。 1 自动 0 手动。
[out]	SweepTypeAutoRule	扫描类型规则。工厂预设：扫描类型规则自动设置成“动态范围优先”规则。 AVSA_VAL_SWEEP_TYPE_AUTO_RULE_SPEED 1 AVSA_VAL_SWEEP_TYPE_AUTO_RULE_DRANGE 2

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFFTWidth

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFFTWidth (ViSession vi,
ViBoolean FFTWidthAuto,
ViReal64 FFTWidth)

【功能】

设置 FFT 宽度及自动手动开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FFTWidthAuto	FFT 宽度自动手动开关。工厂预设：FFT 宽度选择方式为自动。 1 自动 ，0 手动。
[in]	FFTWidth	FFT 宽度。自动将输入的数值处理为以下四档：[<100kHz]、[<3MHz]、[<13MHz]、[<=最大值]， 故可输入范围为 0Hz 到 40MHz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFFTWidth

【方法原型】

ViStatus _VI_FUNC AVSA_GetFFTWidth (ViSession vi,
ViBoolean *FFTWidthAuto,
ViReal64 *FFTWidth)

【功能】

查询 FFT 宽度。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FFTWidthAuto	FFT 宽度自动手动开关。
[out]	FFTWidth	FFT 宽度。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureNDB

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureNDB (ViSession vi,
ViBoolean NDBAuto,
ViReal64 NDB)

【功能】

打开或者关闭测量从信号峰值点下降 NdB 的带宽值的功能及自动手动开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	NDBAuto	NDB 自动手动开关。 1 自动 ，0 手动。
[in]	NDB	NDB 值。工厂预设：3.0。设置范围：最小值为 1.0，最大值为 60.0。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetNDB

【方法原型】

ViStatus _VI_FUNC AVSA_GetNDB (ViSession vi,
ViReal64 *NDB,
ViReal64 *NDBResult)

【功能】

查询 NDB 值及测试值。

【参数】

方向	参数名	说明

[in]	vi	会话句柄。
[out]	NDB	NDB 值。范围：最小值为 1.0，最大值为 60.0。
[out]	NDBResult	NDB 测试值。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigurePeakSort

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigurePeakSort (ViSession vi,
ViInt32 PeakSort)

【功能】

设置峰值表格中的峰值列表是按照频率递增原则排序还是按照幅度递减原则排序。。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	PeakSort	排序类型。 AVSA_VAL_PEAK_SORT_FREQUENCY 1 频率递增原则排序 AVSA_VAL_PEAK_SORT_AMPLITUDE 2 幅度递减原则排序

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetPeakSort

【方法原型】

ViStatus _VI_FUNC AVSA_GetPeakSort (ViSession vi,
ViInt32 *PeakSort)

【功能】

查询峰值表格中的峰值列表是按照频率递增原则排序还是按照幅度递减原则排序。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	PeakSort	排序类型。 AVSA_VAL_PEAK_SORT_FREQUENCY 1 频率递增原则排序 AVSA_VAL_PEAK_SORT_AMPLITUDE 2

		幅度递减原则排序

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA _ ConfigurePeakReadOut

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigurePeakReadOut (ViSession vi,
ViInt32 PeakReadout)

【功能】

设置峰值表格中的峰值列表仅显示高于限定线的峰值数据、还是低于限定线的峰值数据，还是所有峰值数据均显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	PeakReadout	显示依据。 AVSA_VAL_PEAK_READOUT_ALL 1 全部显示。 AVSA_VAL_PEAK_READOUT_GTDLINE 2 高于显示。 AVSA_VAL_PEAK_READOUT_LTDLINE 3 低于显示。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetPeakReadOut

【方法原型】

ViStatus _VI_FUNC AVSA_GetPeakReadOut (ViSession vi,
ViInt32 *PeakReadOut)

【功能】

查询峰值表格中的峰值列表仅显示高于限定线的峰值数据、还是低于限定线的峰值数据，还是所有峰值数据均显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	PeakReadOut	显示依据。 AVSA_VAL_PEAK_READOUT_ALL 1

		全部显示。 AVSA_VAL_PEAK_READOUT_GTDLINE 2 高于显示。 AVSA_VAL_PEAK_READOUT_LTDLINE 3 低于显示。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigurePeakTrack

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigurePeakTrack (ViSession vi,
ViInt32 Marker,
ViBoolean PeakTrackAuto)

【功能】

设置峰值追踪开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Marker	标记号。取值范围：1 到 12。
[in]	PeakTrackAuto	峰值跟踪开关。 1 开， 0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetPeakTrack

【方法原型】

ViStatus _VI_FUNC AVSA_GetPeakTrack (ViSession vi,
ViInt32 Marker,
ViBoolean *PeakTrackAuto)

【功能】

查询峰值追踪开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Marker	标记号。取值范围：1 到 12。
[out]	PeakTrackAuto	峰值跟踪开关。

		1 开， 0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigurePeakPeak

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigurePeakPeak (ViSession vi,
ViInt32 Marker)

【功能】

打开峰-峰值搜索功能。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Marker	标记号。取值范围：1 到 12。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureDispalyGrid

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureDispalyGrid (ViSession vi,
ViBoolean DisplayGridState)

【功能】

选择屏幕轨迹区的格线是否显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	DisplayGridState	显示开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetDispalyGrid

【方法原型】

ViStatus _VI_FUNC AVSA_GetDispalyGrid (ViSession vi,
ViBoolean *DisplayGridState)

【功能】

查询屏幕轨迹区的格线是否显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	DisplayGridState	显示开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureDispalyLine

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureDispalyLine (ViSession vi,
ViBoolean DisplayLineState,
ViReal64 DisplayLine)

【功能】

为开时，在屏幕上激活一条可调整的水平参考线。此线显示的是相对于参考电平的一个幅度值。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	DisplayLineState	水平指导线的开关。 1 开，0 关。
[in]	DisplayLine	水平指导线相对于参考电平的幅度值。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetDispalyLine

【方法原型】

ViStatus _VI_FUNC AVSA_GetDispalyLine (ViSession vi,
ViBoolean *DisplayLineState,
ViReal64 *DisplayLine)

【功能】

查询是否在屏幕上激活一条可调整的水平参考线。此线显示的是相对于参考电平的一个幅度值。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	DisplayLineState	水平指导线的开关。 1 开，0 关。
[out]	DisplayLine	水平指导线相对于参考电平的幅度值。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureDispalyAnnotation

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureDispalyAnnotation (ViSession vi,
ViBoolean DisplayAnnotationState)

【功能】

选择屏幕状态区的信息是否显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	DisplayAnnotationState	屏幕状态显示区显示开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetDispalyAnnotation

【方法原型】

ViStatus _VI_FUNC AVSA_GetDispalyAnnotation (ViSession vi,
ViBoolean *DisplayAnnotationState)

【功能】

查询屏幕状态区的信息是否显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	DisplayAnnotationState	屏幕状态显示区显示开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureDispalyFREQuency

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureDispalyFREQuency (ViSession vi,
ViBoolean DisplayFrequencyState)

【功能】

选择屏幕状态区的频率信息是否显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	DisplayFrequencyState	屏幕状态区的频率信息显示开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetDispalyFREQuency

【方法原型】

ViStatus _VI_FUNC AVSA_GetDispalyFREQuency (ViSession vi,
ViBoolean *DisplayFrequencyState)

【功能】

查询屏幕状态区的频率信息是否显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	DisplayFrequencyState	屏幕状态区的频率信息显示开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureLimitLineDIspaly

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureLimitLineDIspaly (ViSession vi,
ViInt32 limitLine,
ViBoolean displaySwtich)

【功能】

打开或关闭限定线显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	limitLine	限定线号。1 到 6。
[in]	displaySwtich	限定线显示开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetLimitLineDIspaly

【方法原型】

ViStatus _VI_FUNC AVSA_GetLimitLineDIspaly (ViSession vi,
ViInt32 LimitLine,
ViBoolean *DisplayState)

【功能】

查询是否打开或关闭限定线显示。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LimitLine	限定线号。1 到 6。
[out]	DisplayState	限定线显示开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureLimitLineType

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureLimitLineType (ViSession vi,
ViInt32 limitLine,
ViInt32 limitLineType)

【功能】

指定测试轨迹数据超过还是低于限定线数值为 FAIL。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	limitLine	限定线号。1 到 6。

[in]	limitLineType	测试轨迹数据超过还是低于限定线数值为 FAIL。 AVSA_VAL_LIMITLINE_TYPE_UPPER 1 高于。 AVSA_VAL_LIMITLINE_TYPE_LOWER 2 低于。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetLimitLineType

【方法原型】

ViStatus _VI_FUNC AVSA_GetLimitLineType (ViSession vi,
ViInt32 LimitLine,
ViInt32 *LimitLineType)

【功能】

指定测试轨迹数据超过还是低于限定线数值为 FAIL。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LimitLine	限定线号。1 到 6。
[out]	LimitLineType	测试轨迹数据超过还是低于限定线数值为 FAIL。 AVSA_VAL_LIMITLINE_TYPE_UPPER 1 高于。 AVSA_VAL_LIMITLINE_TYPE_LOWER 2 低于。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ConfigureLimitLineTrace

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureLimitLineTrace (ViSession vi,
ViInt32 LimitLine,
ViInt32 LimitLineTrace)

【功能】

设置当前选择的限定线应用于哪一条轨迹。

【参数】

方向	参数名	说明

[in]	vi	会话句柄。
[in]	LimitLine	限定线号。1 到 6。
[in]	LimitLineTrace	轨迹号 1 到 6.

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_GetLimitLineTrace

【方法原型】

ViStatus _VI_FUNC AVSA_GetLimitLineTrace (ViSession vi,
ViInt32 LimitLine,
ViInt32 *LimitLineTrace)

【功能】

查询当前选择的限定线应用于哪一条轨迹。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LimitLine	限定线号。1 到 6。
[out]	LimitLineTrace	轨迹号 1 到 6.

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureLimitLineProperty

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureLimitLineProperty (ViSession vi,
ViInt32 LimitLine,
ViInt32 Trace,
ViInt32 LimitLineType,
ViInt32 CnotrolInterpolate,
ViInt32 AmplitudeInterpolate,
ViBoolean CFRelativeSwitch,
ViBoolean YRelativeSwitch)

【功能】

设置限定线属性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

[in]	LimitLine	限定线号，1 到 6。
[in]	Trace	指定当前选择的限定线应用于哪一条轨迹。1 到 6.
[in]	LimitLineType	测试轨迹数据超过还是低于限定线数值为 FAIL。 AVSA_VAL_LIMITLINE_TYPE_UPPER 1 高于。 AVSA_VAL_LIMITLINE_TYPE_LOWER 2 低于。
[in]	CnotrolInterpolate	指定当前选择的限定线编辑时，频率轴依据何种插值规则进行数据的内插运算。工厂预设：线性。 1 对数 2 线性
[in]	AmplitudeInterpolate	指定当前选择的限定线编辑时，幅度轴依据何种插值规则进行数据的内插运算。工厂预设：对数。 1 对数 2 线性
[in]	CFRelativeSwitch	指定当前选择的限定线编辑时，其 X 轴数据分别是相对值还是绝对值。工厂预设：关。 1 相对，0 绝对
[in]	YRelativeSwitch	指定当前选择的限定线编辑时，其 Y 轴数据分别是相对值还是绝对值。工厂预设：关。 相对，0 绝对

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetLimitLineProperty

【方法原型】

```
ViStatus _VI_FUNC AVSA_GetLimitLineProperty (ViSession vi,
                                             ViInt32 LimitLine,
                                             ViInt32 *Trace,
                                             ViInt32 *TimitLineType,
                                             ViInt32 *CnotrolInterpolate,
                                             ViInt32 *AmplitudeInterpolate,
                                             ViBoolean *CFRelativeSwitch,
                                             ViBoolean *YRelativeSwitch)
```

【功能】

查询限定线属性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LimitLine	限定线号，1 到 6。
[out]	Trace	指定当前选择的限定线应用于哪一条轨迹。1 到 6.
[out]	TimitLineType	测试轨迹数据超过还是低于限定线数值为 FAIL。 AVSA_VAL_LIMITLINE_TYPE_UPPER 1 高于。 AVSA_VAL_LIMITLINE_TYPE_LOWER 2 低于。
[out]	CnotrolInterpolate	指定当前选择的限定线编辑时，频率轴依据何种插值规则进行数据的内插运算。工厂预设：线性。 1 对数 2 线性
[out]	AmplitudeInterpolate	指定当前选择的限定线编辑时，幅度轴依据何种插值规则进行数据的内插运算。工厂预设：对数。 1 对数 2 线性
[out]	CFRelativeSwitch	指定当前选择的限定线编辑时，其 X 轴数据分别是相对值还是绝对值。工厂预设：关。 1 相对，0 绝对
[out]	YRelativeSwitch	指定当前选择的限定线编辑时，其 Y 轴数据分别是相对值还是绝对值。工厂预设：关。 1 相对，0 绝对

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureLimitLineMargin

【方法原型】

```
ViStatus _VI_FUNC AVSA_ConfigureLimitLineMargin (ViSession vi,
                                                 ViInt32 LimitLine,
                                                 ViBoolean LimitLineMarginState,
                                                 ViReal64 LimitLineMargin)
```

【功能】

指定当前选择的限定线测试时，幅度数据超过限定线边界多少 dB 才表示超过限定。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

[in]	LimitLine	限定线号。1 到 6。
[in]	LimitLineMarginAuto	边界开关。工厂预设：关。 1 开，0 关。
[in]	LimitLineMargin	当前选择的限定线测试时，幅度数据超过限定线边界多少 dB 才表示超过限定。设置范围：-40dB 到 40dB。 默认单位：dB。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetLimitLineMargin

【方法原型】

ViStatus _VI_FUNC AVSA_GetLimitLineMargin (ViSession vi,
ViInt32 LimitLine,
ViBoolean *LimitLineMarginAuto,
ViReal64 *LimitLineMargin)

【功能】

查询指定当前选择的限定线测试时，幅度数据超过限定线边界多少 dB 才表示超过限定。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LimitLine	限定线号。1 到 6。
[out]	LimitLineMarginAuto	边界开关。工厂预设：关。 1 开，0 关。
[out]	LimitLineMargin	当前选择的限定线测试时，幅度数据超过限定线边界多少 dB 才表示超过限定。设置范围：-40dB 到 40dB。 默认单位：dB。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureLimitLineTestSwitch

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureLimitLineTestSwitch (ViSession vi,
ViBoolean LimitLineTestSwitch)

【功能】

指定是否开启限定线测试。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LimitLineTestSwitch	限定线测试开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetLimitLineTestSwitch

【方法原型】

ViStatus _VI_FUNC AVSA_GetLimitLineTestSwitch (ViSession vi,
ViBoolean *LimitLineTestSwitch)

【功能】

查询是否开启限定线测试。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	LimitLineTestSwitch	限定线测试开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureLimitLineControlDomain

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureLimitLineControlDomain (ViSession vi,
ViInt32 LimitLineControlDomainUnit)

【功能】

指定限定线横轴单位为频率还是时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LimitLineControlDomainUnit	限定线横轴单位。工厂预设：频率。 AVSA_VAL_LIMITLINE_CONTROL_ DOMAIN_UNIT_FREQUENCY 1 频率 AVSA_VAL_LIMITLINE_CONTROL_ DOMAIN_UNIT_TIME 2 时间

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetLimitLineControlDomain

【方法原型】

ViStatus _VI_FUNC AVSA_GetLimitLineControlDomain (ViSession vi,
ViInt32 *LimitLineControlDomainUnit)

【功能】

查询限定线横轴单位为频率还是时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	LimitLineControlDomainUnit	限定线横轴单位。工厂预设：频率。 AVSA_VAL_LIMITLINE_CONTROL_DOMAIN_UNIT_FREQUENCY 1 频率 AVSA_VAL_LIMITLINE_CONTROL_DOMAIN_UNIT_TIME 2 时间

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ DeleteAllLimitLine

【方法原型】

ViStatus _VI_FUNC AVSA_DeleteAllLimitLine (ViSession vi)

【功能】

将所有限定线数据清除。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ SaveLimitLine

【方法原型】

ViStatus _VI_FUNC AVSA_SaveLimitLine (ViSession vi,
ViInt32 LimitLine,
ViString FilePath)

【功能】

保存限定线到文件。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LimitLine	限定线号。1 或 2。
[in]	FilePath	保存文件名。文件后缀名必须是“.lim”，文件在仪器中保存于固定路径下，因此只写文件名即可。示例：”limitline.lim"

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ LoadLimitLine

【方法原型】

ViStatus _VI_FUNC AVSA_LoadLimitLine (ViSession vi,
ViInt32 LimitLine,
ViString FilePath)

【功能】

将保存的限定线文件数据应用于当前选择的限定线。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	LimitLine	限定线号。1 或 2。
[in]	FilePath	调用文件名。文件后缀名必须是“.lim”，文件在仪器中保存于固定路径下，因此只写文件名即可。示例：”limitline.lim"

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureCouplingType

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureCouplingType (ViSession vi,
ViInt32 CouplingType)

【功能】

选择信号分析仪射频输入端口的耦合方式为直流耦合(DC)还是交流耦合(AC)。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	CouplingType	分析仪射频输入端口的耦合方式。工厂预设：AC。 AVSA_VAL_INPUT_COUPLING_TYPE_AC 1 AC AVSA_VAL_INPUT_COUPLING_TYPE_DC 2 DC

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetCouplingType

【方法原型】

ViStatus _VI_FUNC AVSA_GetCouplingType (ViSession vi,
ViInt32 *CouplingType)

【功能】

查询信号分析仪射频输入端口的耦合方式为直流耦合(DC)还是交流耦合(AC)

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	CouplingType	分析仪射频输入端口的耦合方式。工厂预设：AC。 AVSA_VAL_INPUT_COUPLING_TYPE_AC 1 AC AVSA_VAL_INPUT_COUPLING_TYPE_DC 2 DC

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFEED

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFEED (ViSession vi,
ViInt32 FEED)

【功能】

选择输入端口类型为射频输入信号或者内部校准信号。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FEED	输入端口类型。工厂预设：射频。 AVSA_VAL_INPUT_FEED_RF 1 射频 AVSA_VAL_INPUT_FEED_AREF 2 内部校准

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFEED

【方法原型】

ViStatus _VI_FUNC AVSA_GetFEED (ViSession vi,
ViInt32 *FEED)

【功能】

查询输入端口类型为射频输入信号或者内部校准信号。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FEED	输入端口类型。工厂预设：射频。 AVSA_VAL_INPUT_FEED_RF 1 射频 AVSA_VAL_INPUT_FEED_AREF 2 内部校准

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFEEDAreference

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFEEDAreference (ViSession vi,
ViInt32 InputFEEDAreference)

【功能】

选择信号分析仪内部校准信号。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	InputFEEDAreference	信号分析仪内部校准信号. AVSA_VAL_INPUT_FEED_AREFERENCE_REF500 1 500MHz AVSA_VAL_INPUT_FEED_AREFERENCE_REFCOMB 2 梳状波 AVSA_VAL_INPUT_FEED_AREFERENCE_REF4800 3 4800MHz AVSA_VAL_INPUT_FEED_AREFERENCE_OFF 4 关闭

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFEEDAreference

【方法原型】

ViStatus _VI_FUNC AVSA_GetFEEDAreference (ViSession vi,
ViInt32 *InputFEEDAreference)

【功能】

查询信号分析仪内部校准信号。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	InputFEEDAreference	信号分析仪内部校准信号. AVSA_VAL_INPUT_FEED_AREFERENCE_REF500 1 500MHz AVSA_VAL_INPUT_FEED_AREFERENCE_REFCOMB 2 梳状波 AVSA_VAL_INPUT_FEED_AREFERENCE_REF4800 3 4800MHz AVSA_VAL_INPUT_FEED_AREFERENCE_OFF 4 关闭

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureROSCillatorSource

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureROSCillatorSource (ViSession vi,
ViInt32 ROSCillatorSource)

【功能】

选择内部频率参考或外部频率参考。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	ROSCillatorSource	选择内部频率参考或外部频率参考。工厂预设：内部。 AVSA_VAL_INPUT_ROSC_SOURCE_INT 1 内部 AVSA_VAL_INPUT_ROSC_SOURCE_EXT

		2　　外部

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetROSCillatorSource

【方法原型】

ViStatus _VI_FUNC AVSA_GetROSCillatorSource (ViSession vi,
ViInt32 *ROSCillatorSource)

【功能】

查询内部频率参考或外部频率参考。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	ROSCillatorSource	选择内部频率参考或外部频率参考。工厂预设：内部。 AVSA_VAL_INPUT_ROSC_SOURCE_INT 1　　内部 AVSA_VAL_INPUT_ROSC_SOURCE_EXT 2　　外部

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureAUXOutput

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureAUXOutput (ViSession vi,
ViInt32 AUXOutput)

【功能】

设置高中频输出开关。（只有在配置 H04 选件的情况下，才起作用）

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AUXOutput	高中频输出开关。工厂预设：关。 AVSA_VAL_AUX_OUTPUT_SIF　1　高频 AVSA_VAL_AUX_OUTPUT_AIF　2　中频 AVSA_VAL_AUX_OUTPUT_OFF　3　关闭

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetAUXOutput

【方法原型】

ViStatus _VI_FUNC AVSA_GetAUXOutput (ViSession vi,
ViInt32 *AUXOutput)

【功能】

查询高中频输出开关状态。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AUXOutput	高中频输出开关。工厂预设：关。 AVSA_VAL_AUX_OUTPUT_SIF 1 高频 AVSA_VAL_AUX_OUTPUT_AIF 2 中频 AVSA_VAL_AUX_OUTPUT_OFF 3 关闭

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureSIF

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureSIF (ViSession vi,
ViReal64 SIF)

【功能】

设置高频输出。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	SIF	高频输出。工厂预设：关。设置范围：275MHz-475MHz。 默认单位：Hz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetSIF

【方法原型】

ViStatus _VI_FUNC AVSA_GetSIF (ViSession vi,
ViReal64 *SIF)

【功能】

查询高频输出。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	SIF	高频输出。范围：275MHz-475MHz。默认单位：Hz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureAIF

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureAIF (ViSession vi,
ViReal64 AIF)

【功能】

设置中频输出。（只有在配置 H04 选件的情况下，才起作用）

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AIF	中频输出。工厂预设：70MHz。设置范围：10MHz-160MHz。默认单位：MHz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetAIF

【方法原型】

ViStatus _VI_FUNC AVSA_GetAIF (ViSession vi,
ViReal64 *AIF)

【功能】

查询中频输出。（只有在配置 H04 选件的情况下，才起作用）

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AIF	中频输出。范围：10MHz-160MHz。默认单位：MHz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureIFAUTOGainSwitch

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureIFAUTOGainSwitch (ViSession vi,

ViBoolean IFAUTOGainSwitch)

【功能】

打开或关闭中频输出的自动增益控制功能。（只有在配置 H03 选件的情况下，才起作用）

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	IFAUTOGainSwitch	自动增益控制开关。工厂预设：关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetIFAUTOGainSwitch

【方法原型】

ViStatus _VI_FUNC AVSA_GetIFAUTOGainSwitch (ViSession vi,

ViBoolean *IFAUTOGainSwitch)

【功能】

查询中频输出的自动增益控制功能。（只有在配置 H03 选件的情况下，才起作用）

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	IFAUTOGainSwitch	自动增益控制开关。1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureIFAUTOGainLevel

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureIFAUTOGainLevel (ViSession vi,

ViInt32 IFAUTOGainLevel)

【功能】

设置自动增益电平。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	IFAUTOGainLevel	自动增益电平。工厂预设：0dBm。

		AVSA_VAL_IF_AUTO_GAIN_LEVEL_5 1 5 dBm AVSA_VAL_IF_AUTO_GAIN_LEVEL_0 2 0 dBm AVSA_VAL_IF_AUTO_GAIN_LEVEL_NEG5 3 -5 dBm AVSA_VAL_IF_AUTO_GAIN_LEVEL_NEG10 4 -10 dBm

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetIFAUTOGainLevel

【方法原型】

ViStatus _VI_FUNC AVSA_GetIFAUTOGainLevel (ViSession vi,
ViInt32 *IFAUTOGainLevel)

【功能】

查询自动增益电平。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	IFAUTOGainLevel	自动增益电平。 AVSA_VAL_IF_AUTO_GAIN_LEVEL_5 1 5 dBm AVSA_VAL_IF_AUTO_GAIN_LEVEL_0 2 0 dBm AVSA_VAL_IF_AUTO_GAIN_LEVEL_NEG5 3 -5 dBm AVSA_VAL_IF_AUTO_GAIN_LEVEL_NEG10 4 -10 dBm

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureMarkerCouple

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureMarkerCouple (ViSession vi,
ViBoolean MarkerCouple)

【功能】

打开或关闭标记关联功能。

【参数】

方向	参数名	说明

[in]	vi	会话句柄。
[in]	MarkerCouple	标记关联功能开关。工厂预设：关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetMarkerCouple

【方法原型】

ViStatus _VI_FUNC AVSA_GetMarkerCouple (ViSession vi,
ViBoolean *MarkerCouple)

【功能】

查询标记关联功能打开或关闭。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	MarkerCouple	标记关联功能开关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureMarkerCountSwitch

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureMarkerCountSwitch (ViSession vi,
ViInt32 Marker,
ViBoolean MarkerCountAuto)

【功能】

设置频标计数功能开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Marker	标记号。取值范围：1 到 12。
[in]	MarkerCountAuto	频标计数功能开关。工厂预设：关。 1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetMarkerCountSwitch

【方法原型】

ViStatus _VI_FUNC AVSA_GetMarkerCountSwitch (ViSession vi,
ViInt32 Marker,
ViBoolean *MarkerCountSwitch)

【功能】

查询频标计数开关。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Marker	标记号。取值范围：1 到 12。
[out]	MarkerCountSwitch	频标计数功能开关。1 开，0 关。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureAudioDemulaitonType

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureAudioDemulaitonType (ViSession vi,
ViBoolean AudioDemulationSwitch,
ViInt32 AudioDemulationType)

【功能】

可以选择关闭解调或者进行 AM 解调、FM 解调。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AudioDemulationSwitch	音频解调开关。工厂预设：关闭。 1 开，0 关。
[in]	AudioDemulationType	音频解调类型。 AVSA_VAL_AUDIO_DEMULATION_TYPE_AM 1 AM 解调 AVSA_VAL_AUDIO_DEMULATION_TYPE_FM 2 FM 解调 AVSA_VAL_AUDIO_DEMULATION_TYPE_OFF 3 关闭

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetAudioDemulaitonType

【方法原型】

ViStatus _VI_FUNC AVSA_GetAudioDemulaitonType (ViSession vi,
ViBoolean *AudioDemulationSwitch,
ViInt32 *AudioDemulationType)

【功能】

查询解调开关状态及类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AudioDemulationSwitch	音频解调开关。 1 开，0 关。
[out]	AudioDemulationType	音频解调类型。 AVSA_VAL_AUDIO_DEMULATION_TYPE_AM 1 AM 解调 AVSA_VAL_AUDIO_DEMULATION_TYPE_FM 2 FM 解调 AVSA_VAL_AUDIO_DEMULATION_TYPE_PM 3 PM 解调 AVSA_VAL_AUDIO_DEMULATION_TYPE_OFF 4 关闭

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureAudioDemulaitonMode

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureAudioDemulaitonMode (ViSession vi,
ViInt32 AudioDemulationMode)

【功能】

选择解调方式为间歇或者连续。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AudioDemulationMode	解调方式。 AVSA_VAL_AUDIO_DEMULATION_MODE_INTER 1 间歇解调。

		AVSA_VAL_AUDIO_DEMULATION_MODE_CONT 2　　连续解调。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetAudioDemulaitonMode

【方法原型】

ViStatus _VI_FUNC AVSA_GetAudioDemulaitonMode (ViSession vi,
ViInt32 *AudioDemulationMode)

【功能】

查询解调方式。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AudioDemulationMode	解调方式。 AVSA_VAL_AUDIO_DEMULATION_MODE_INTER 1　　间歇解调。 AVSA_VAL_AUDIO_DEMULATION_MODE_CONT 2　　连续解调。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureAudioDemulaitonTime

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureAudioDemulaitonTime (ViSession vi,
ViReal64 AudioDemulaitonTime)

【功能】

控制音频解调的时间长度。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AudioDemulaitonTime	解调时间长度。工厂预设：1s。设置范围：最小值为 0.01s，最大值为 100s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetAudioDemulaitonTime

【方法原型】

ViStatus _VI_FUNC AVSA_GetAudioDemulaitonTime (ViSession vi,
ViReal64 *AudioDemulationTime)

【功能】

查询音频解调的时间长度。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AudioDemulationTime	解调时间长度。工厂预设：1s。设置范围：最小值为 0.01s，最大值为 100s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureBurstTrigger

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureBurstTrigger (ViSession vi,
ViReal64 BurstTriggerLevel,
ViInt32 BurstTriggerSlope)

【功能】

设置 Burst 触发电平及极性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	BurstTriggerLevel	Burst 触发电平。工厂预设：-20dBm。设置范围：-270dBm-30dBm。默认单位：dBm
[in]	BurstTriggerSlope	Burst 触发极性。工厂预设：正。 AVSA_VAL_TRIGGER_SLOPE_POSITIVE 1 正 AVSA_VAL_TRIGGER_SLOPE_NEGATIVE 2 负

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetBurstTrigger

【方法原型】

ViStatus _VI_FUNC AVSA_GetBurstTrigger (ViSession vi,

ViReal64 *BurstTriggerLevel,
ViInt32 *BurstTriggerSlope)

【功能】

查询 Burst 触发电平及极性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	BurstTriggerLevel	Burst 触发电平。范围：-270dBm-30dBm。默认单位：dBm
[out]	BurstTriggerSlope	Burst 触发极性。 AVSA_VAL_TRIGGER_SLOPE_POSITIVE 1 正 AVSA_VAL_TRIGGER_SLOPE_NEGATIVE 2 负

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFRAMTrigger

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFRAMTrigger (ViSession vi,
ViInt32 FRAMTriggerSYNC,
ViReal64 FRAMTriggerLevel,
ViInt32 FRAMTriggerSlope)

【功能】

设置 FRAM 触发电平及极性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FRAMTriggerSYNC	同步源。 AVSA_VAL_FRAM_TRIGGER_SYNC_EXTERNAL1 1 外部触发 1。 AVSA_VAL_FRAM_TRIGGER_SYNC_EXTERNAL2 2 外部触发 2。 AVSA_VAL_FRAM_TRIGGER_SYNC_RFBURST 3 Burst 触发。 AVSA_VAL_FRAM_TRIGGER_SYNC_OFF 4 关闭。
[in]	FRAMTriggerLevel	根据同步源不同该参数不同。 外部触发 1 及外部触发 2 时：工厂预设：1.2V。设置范

		围：-5V-5Vs。默认单位：V Burst 触发时： 工厂预设：1.2V。设置范围：-5V-5Vs。默认单位：V
[in]	FRAMTriggerSlope	触发极性。 AVSA_VAL_TRIGGER_SLOPE_POSITIVE 1 正 AVSA_VAL_TRIGGER_SLOPE_NEGATIVE 2 负

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFRAMTrigger

【方法原型】

ViStatus _VI_FUNC AVSA_GetFRAMTrigger (ViSession vi,
ViInt32 *FRAMTriggerSYNC,
ViReal64 *FRAMTriggerLevel,
ViInt32 *FRAMTriggerSlope)

【功能】

查询 FRAM 触发电平及极性。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FRAMTriggerSYNC	同步源。 AVSA_VAL_FRAM_TRIGGER_SYNC_EXTERNAL1 1 外部触发 1。 AVSA_VAL_FRAM_TRIGGER_SYNC_EXTERNAL2 2 外部触发 2。 AVSA_VAL_FRAM_TRIGGER_SYNC_RFBURST 3 Burst 触发。 AVSA_VAL_FRAM_TRIGGER_SYNC_OFF 4 关闭。
[out]	FRAMTriggerLevel	根据同步源不同该参数不同。 外部触发 1 及外部触发 2 时：工厂预设：1.2V。设置范围：-5V-5Vs。默认单位：V Burst 触发时： 工厂预设：1.2V。设置范围：-5V-5Vs。默认单位：V
[out]	FRAMTriggerSlope	触发极性。 AVSA_VAL_TRIGGER_SLOPE_POSITIVE 1 正 AVSA_VAL_TRIGGER_SLOPE_NEGATIVE 2 负

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFRAMTriggerPeriod

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFRAMTriggerPeriod (ViSession vi,
ViReal64 FRAMTriggerPeriod)

【功能】

设置触发模式为定时器触发的定时器周期。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FRAMTriggerPeriod	定时器触发的定时器周期。工厂预设：100ms。设置范围：100ns-20s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFRAMTriggerPeriod

【方法原型】

ViStatus _VI_FUNC AVSA_GetFRAMTriggerPeriod (ViSession vi,
ViReal64 *FRAMTriggerPeriod)

【功能】

查询触发模式为定时器触发的定时器周期。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FRAMTriggerPeriod	定时器触发的定时器周期范围：100ns-20s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFRAMTriggerOffset

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFRAMTriggerOffset (ViSession vi,
ViReal64 FRAMTriggerOffset)

【功能】

设置触发模式为定时器触发的定时器偏移，该值不能超过定时器周期。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FRAMTriggerOffset	定时器触发的定时器偏移。工厂预设：100ms。设置范围：10ns-20s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFRAMTriggerOffset

【方法原型】

ViStatus _VI_FUNC AVSA_GetFRAMTriggerOffset (ViSession vi,
ViReal64 *FRAMTriggerOffset)

【功能】

查询触发模式为定时器触发的定时器偏移。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FRAMTriggerOffset	定时器触发的定时器偏移。范围：10ns-20s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureFRAMTriggerHoldOff

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureFRAMTriggerHoldOff (ViSession vi,
ViReal64 FRAMTriggerHoldoff)

【功能】

设置触发模式为定时器触发的同步源阻断时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	FRAMTriggerHoldoff	同步源阻断时间。工厂预设：100ms。设置范围：100ns-20s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetFRAMTriggerHoldOff

【方法原型】

ViStatus _VI_FUNC AVSA_GetFRAMTriggerHoldOff (ViSession vi,
ViReal64 *FRAMTriggerHoldOff)

【功能】

查询触发模式为定时器触发的同步源阻断时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	FRAMTriggerHoldOff	同步源阻断时间。工厂预设：100ms。设置范围：100ns-20s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureVideoTriggerDelay

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureVideoTriggerDelay (ViSession vi,
ViBoolean VideoTriggerDelaySwitch,
ViReal64 VideoTriggerDelay)

【功能】

设置视频触发触发延迟开关并设置触发延迟时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	VideoTriggerDelaySwitch	视频触发触发延迟开关。工厂预设：关。1 开，0 关。
[in]	VideoTriggerDelay	触发延迟时间。设置范围：20ns-60s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetVideoTriggerDelay

【方法原型】

ViStatus _VI_FUNC AVSA_GetVideoTriggerDelay (ViSession vi,

ViBoolean *VideoTriggerDelaySwitch,
ViReal64 *VideoTriggerDelay)

【功能】

查询视频触发触发延迟开关并设置触发延迟时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	VideoTriggerDelaySwitch	视频触发触发延迟开关。 1 开，0 关。
[out]	VideoTriggerDelay	触发延迟时间。范围：20ns-60s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureExternal1TriggerDelay

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureExternal1TriggerDelay (ViSession vi,
ViBoolean External1TriggerDelaySwitch,
ViReal64 External1TriggerDelay)

【功能】

设置 External1 触发触发延迟开关并设置触发延迟时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	External1TriggerDelaySwitch	External1 触发触发延迟开关。工厂预设：关。 1 开，0 关。
[in]	External1TriggerDelay	触发延迟时间。范围：20ns-60s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetExternal1TriggerDelay

【方法原型】

ViStatus _VI_FUNC AVSA_GetExternal1TriggerDelay (ViSession vi,
ViBoolean *External1TriggerDelaySwitch,
ViReal64 *External1TriggerDelay)

【功能】

查询 External1 触发触发延迟开关并设置触发延迟时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	External1TriggerDelaySwitch	External1 触发触发延迟开关。 1 开，0 关。
[out]	External1TriggerDelay	触发延迟时间。范围：20ns-60s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureExternal2TriggerDelay

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureExternal2TriggerDelay (ViSession instrumentHandle,
ViBoolean External2TriggerDelaySwitch,
ViReal64 External2TriggerDelay);

【功能】

设置 External2 触发触发延迟开关并设置触发延迟时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	External2TriggerDelaySwitch	External2 触发触发延迟开关。工厂预设：关。 1 开，0 关。
[in]	External2TriggerDelay	触发延迟时间。范围：20ns-60s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetExternal2TriggerDelay

【方法原型】

ViStatus _VI_FUNC AVSA_GetExternal2TriggerDelay (ViSession vi,
ViBoolean *External2TriggerDelaySwitch,
ViReal64 *External2TriggerDelay)

【功能】

查询 External2 触发触发延迟开关并设置触发延迟时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	External2TriggerDelaySwitch	External2 触发触发延迟开关。

		1 开，0 关。
[out]	External2TriggerDelay	触发延迟时间。范围：20ns-60s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureBurstTriggerDelay

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureBurstTriggerDelay (ViSession vi,
ViBoolean BurstTriggerDelaySwitch,
ViReal64 BurstTriggerDelay)

【功能】

设置 Burst 触发触发延迟开关并设置触发延迟时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	BurstTriggerDelaySwitch	Burst 触发触发延迟开关。工厂预设：关。 1 开，0 关。
[in]	BurstTriggerDelay	触发延迟时间。范围：20ns-60s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetBurstTriggerDelay

【方法原型】

ViStatus _VI_FUNC AVSA_GetBurstTriggerDelay (ViSession vi,
ViBoolean *BurstTriggerDelaySwitch,
ViReal64 *BurstTriggerDelay)

【功能】

查询 Burst 触发触发延迟开关并设置触发延迟时间。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	BurstTriggerDelaySwitch	Burst 触发触发延迟开关。 1 开，0 关。
[out]	BurstTriggerDelay	触发延迟时间。范围：20ns-60s。默认单位：s。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureADCDither

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureADCDither (ViSession vi,
ViBoolean ADCDitherAuto,
ViInt32 ADCDither)

【功能】

用于控制信号分析仪中频采集电路的 ADC 抖动是否打开。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	ADCDitherAuto	中频采集电路的 ADC 抖动自动手动开关。工厂预设：自动。 1　自动，0　手动。
[in]	ADCDither	中频采集电路的 ADC 抖动。 AVSA_VAL_ADC_DITHER_OFF 1　　关。 AVSA_VAL_ADC_DITHER_ON 2　　开。 AVSA_VAL_ADC_DITHER_HIGH 3　　高。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetADCDither

【方法原型】

ViStatus _VI_FUNC AVSA_GetADCDither (ViSession vi,
ViBoolean *ADCDitherAuto,
ViInt32 *ADCDither)

【功能】

查询信号分析仪中频采集电路的 ADC 抖动是否打开等。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	ADCDitherAuto	中频采集电路的 ADC 抖动自动手动开关。1　自动，0　手动。
[out]	ADCDither	中频采集电路的 ADC 抖动。

		AVSA_VAL_ADC_DITHER_OFF 1　关。 AVSA_VAL_ADC_DITHER_ON 2　开。 AVSA_VAL_ADC_DITHER_HIGH 3　高。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureAverageCount

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureAverageCount (ViSession vi,
ViInt32 AverageCount)

【功能】

给定在进行轨迹平均处理时的平均次数。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AverageCount	平均次数。 工厂预设：10。设置范围：1-10000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetAverageCount

【方法原型】

ViStatus _VI_FUNC AVSA_GetAverageCount (ViSession vi,
ViInt32 *AverageCount)

【功能】

查询在进行轨迹平均处理时的平均次数。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AverageCount	平均次数。范围：1-10000。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureAverageType

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureAverageType (ViSession vi,
ViBoolean AverageTypeAuto,
ViInt32 AverageType)

【功能】

设置轨迹平均的平均类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	AverageTypeAuto	平均类型自动手动开关。工厂预设：自动。 1 自动，0 手动。
[in]	AverageType	平均类型。 AVSA_VAL_AVERAGE_TYPE_RMS 1 AVSA_VAL_AVERAGE_TYPE_LOG 2 AVSA_VAL_AVERAGE_TYPE_SCAL 3

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetAverageType

【方法原型】

ViStatus _VI_FUNC AVSA_GetAverageType (ViSession vi,
ViBoolean *AverageTypeAuto,
ViInt32 *AverageType)

【功能】

查询轨迹平均的平均类型。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	AverageTypeAuto	平均类型自动手动开关。1 自动，0 手动。
[out]	AverageType	平均类型。 AVSA_VAL_AVERAGE_TYPE_RMS 1 AVSA_VAL_AVERAGE_TYPE_LOG 2 AVSA_VAL_AVERAGE_TYPE_SCAL 3

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigurePhaseNoiseOpt

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigurePhaseNoiseOpt (ViSession vi,
ViBoolean PhaseNoiseOptAuto,
ViInt32 PhaseNoiseOptType)

【功能】

设置相噪优化。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	PhaseNoiseOptAuto	相噪优化自动手动开关。 1 自动，0 手动。
[in]	PhaseNoiseOptType	相噪优化类型。 1 频偏<150kHz 2 频偏>150kHz 3 快速调谐

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetPhaseNoiseOpt

【方法原型】

ViStatus _VI_FUNC AVSA_GetPhaseNoiseOpt (ViSession vi,
ViBoolean *PhaseNoiseOptAuto,
ViInt32 *PhaseNoiseOptType)

【功能】

查询相噪优化。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	PhaseNoiseOptAuto	相噪优化自动手动开关。 1 自动，0 手动。
[out]	PhaseNoiseOptType	相噪优化类型。 1 频偏<150kHz 2 频偏>150kHz 3 快速调谐

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigurePreselectorSwitch

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigurePreselectorSwitch (ViSession vi,
ViBoolean PreselectorSwitch)

【功能】

用于设置信号分析仪前端预选器是否旁路。（只有在配置 H36 选件的情况下，才起作用）

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	PreselectorSwitch	预选器开关。 1 预选，0 旁路。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetPreselectorSwitch

【方法原型】

ViStatus _VI_FUNC AVSA_GetPreselectorSwitch (ViSession vi,
ViBoolean *PreselectorSwitch)

【功能】

查询设置信号分析仪前端预选器是否旁路。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	PreselectorSwitch	预选器开关。 1 预选，0 旁路。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ConfigureMeasureControl

【方法原型】

ViStatus _VI_FUNC AVSA_ConfigureMeasureControl (ViSession vi,
ViInt32 MeasureControlType)

【功能】

暂停当前测量或继续进行测量。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	MeasureControlType	测量类型。 AVSA_VAL_MEASURE_CONTROL_TYPE_REST 1　重新测量。 AVSA_VAL_MEASURE_CONTROL_TYPE_PAUSE 2　暂停测量。 AVSA_VAL_MEASURE_CONTROL_TYPE_RESTART 3　恢复测量。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

Measurement 子系统

- AVSA_ Initiate

【方法原型】

ViStatus _VI_FUNC AVSA_Initiate (ViSession vi)

【功能】

立即进行测量，函数使仪器触发一次扫描。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ Abort

【方法原型】

ViStatus _VI_FUNC AVSA_Abort (ViSession vi)

【功能】

连续扫描时，等同于【重新扫描】。单次扫描时，它没有什么行动，idle 是空闲。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ReadYTrace

【方法原型】

ViStatus _VI_FUNC AVSA_ReadYTrace (ViSession vi,
ViConstString TraceName,
ViInt32 MaxTime,
ViInt32 ArrayLength,
ViInt32 *ActualPoints,
ViReal64 Amplitude[]);

【功能】

使仪器停止当前测量，开始一次新的测量，并在测量完成后，获取轨迹数据。

注意：1，数据缓存 Amplitude 大小应大于扫描点数，否则存在数据溢出的危险。

2，在检波方式为正常检波的情况下，每一个数据点为 2 个数据（最大值和最小值），数据缓存应为 2 倍的数据缓存大小。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	TraceName	轨迹名称。TRACE1，TRACE2 到 TRACE6.
[in]	MaxTime	读取超时时间。
[in]	ArrayLength	数组长度。即当前扫描点数。
[out]	ActualPoints	读取到的轨迹点数。
[out]	Amplitude	读取的轨迹数组。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ FetchYTrace

【方法原型】

ViStatus _VI_FUNC AVSA_FetchYTrace (ViSession vi,
ViConstString TraceName,
ViInt32 ArrayLength,
ViInt32 *ActualPoints,
ViReal64 Amplitude[])

【功能】

不改变仪器任何状态，仅是从仪器中获取轨迹数据。

注意：1，数据缓存 Amplitude 大小应大于扫描点数，否则存在数据溢出的危险。

2，AV4051 检波方式为正常检波的情况下，在每一个频点都保存了该店的最大值和最小值，远程控制取出的轨迹数据为每个频点的最大值。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	TraceName	轨迹名称。TRACE1，TRACE2 到 TRACE6.
[in]	ArrayLength	数组长度。即当前扫描点数。。
[out]	ActualPoints	读取到的轨迹点数。
[out]	Amplitude	读取的轨迹数组。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

PNoise 子系统

- AVSA_ PNoise_FrequencyCarrier

【方法原型】

ViStatus _VI_FUNC AVSA_PNoise_FrequencyCarrier (ViSession vi,
ViReal64 CarrierFrequency)

【功能】

相位噪声模式下设置载波频率。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	CarrierFrequency	载波频率。工厂预设：1GHz。设置范围：最小值为 1MHz，最大值为信号分析仪最高频率上限。默认单位：Hz

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ PNoise_FrequencyCarrierSearchExecute

【方法原型】

ViStatus _VI_FUNC AVSA_PNoise_FrequencyCarrierSearchExecute (ViSession vi);

【功能】

相位噪声模式下进行全频宽范围内的载波搜索，并在搜索结束后，将新的载波的频率置为当前载波频率。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ PNoise_FrequencyOffset

【方法原型】

ViStatus _VI_FUNC AVSA_PNoise_FrequencyOffset (ViSession vi,
ViReal64 StartOffset,
ViReal64 StopOffset)

【功能】

相位噪声模式下设置测量频偏。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	StartOffset	起始频偏。工厂预设：10Hz。设置范围： 10Hz-10MHz。
[in]	StopOffset	中止频偏。工厂预设：100MHz。设置范 100Hz-100MHz。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ PNoise_LogPlotConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_PNoise_LogPlotConfigure (ViSession vi)

【功能】

相位噪声模式下对数曲线测量。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ PNoise_MarkerConfigure

【方法原型】

ViStatus _VI_FUNC AVSA_PNoise_MarkerConfigure (ViSession vi,

ViInt32 Marker,
ViReal64 XFrequency,
ViInt32 XPosition);

【功能】

相位噪声模式下设置频标。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Marker	频标号。1 到 12.
[in]	XFrequency	X 横轴频率坐标。
[in]	XPosition	X 横轴位置。1 到 1001。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ PNoise_MarkerQuery

【方法原型】

ViStatus _VI_FUNC AVSA_PNoise_MarkerQuery (ViSession vi,
ViInt32 Marker,
ViReal64 *XFrequency,
ViReal64 *XPosition,
ViReal64 *YPhasisNoise)

【功能】

相位噪声模式下查询频标。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	Marker	频标号。1 到 12.
[out]	XFrequency	频标的频率读数
[out]	XPosition	频标的位置（点数）
[out]	YPhasisNoise	频标的相噪读数

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

Utility Functions 子系统

- AVSA_ reset

【方法原型】

ViStatus _VI_FUNC　AVSA_reset (ViSession vi);

【功能】

复位仪器状态。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ResetWithDefaults

【方法原型】

ViStatus _VI_FUNC　AVSA_ResetWithDefaults (ViSession vi);

【功能】

复位仪器，并按照用户配置为设置仪表为配置的参数。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_revision_query

【方法原型】

ViStatus _VI_FUNC　AVSA_revision_query (ViSession vi,
ViChar instrumentDriverRevision[],
ViChar firmwareRevision[]);

【功能】

返回仪器驱动版本和仪器硬件版本。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	instrumentDriverRevision	仪器驱动版本。
[out]	firmwareRevision	仪器硬件版本。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ InvalidateAllAttributes

【方法原型】

ViStatus _VI_FUNC　AVSA_InvalidateAllAttributes (ViSession vi);

【功能】

使所有属性的缓存值无效。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_error_query

【方法原型】

ViStatus _VI_FUNC　AVSA_error_query (ViSession vi,
ViInt32 *errorCode,
ViChar errorMessage[]);

【功能】

获取错误列表。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[out]	errorCode	错误号
[out]	errorMessage	错误信息

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetError

【方法原型】

ViStatus _VI_FUNC　AVSA_GetError (ViSession vi,
ViStatus *code,
ViInt32 bufferSize,

ViChar description[]);

【功能】

返回仪器已发生的错误。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	code	错误代码 ID
[in]	bufferSize	错误描述缓存大小
[in]	description	错误描述缓存

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ClearError

【方法原型】

ViStatus _VI_FUNC　AVSA_ClearError (ViSession vi);

【功能】

清除错误信息。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_error_ message

【方法原型】

ViStatus _VI_FUNC　AVSA_error_message (ViSession vi,
ViStatus errorCode,
ViChar errorMessage[256]);

【功能】

将返回的错误号翻译为用户可读的错误信息。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	errorCode	错误号
[in]	errorMessage	根据错误号翻译出的错误信息

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ GetNextInterchangeWarning

【方法原型】

ViStatus _VI_FUNC AVSA_GetNextInterchangeWarning (ViSession vi,
ViInt32 bufferSize,
ViChar warnString[]);

【功能】

返回互换性警告，互换性警告意味着该应用程序应用到不同的仪器时会导致不同的反应。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。
[in]	bufferSize	输入你规定的互换性警告参数的字节数
[in]	warnString	The buffer in which the function returns the description of the oldest value coercion of the instrument session. The buffer must be of type ViChar and have at least as many bytes as indicated in the Buffer Size parameter.

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ResetInterchangeCheck

【方法原型】

ViStatus _VI_FUNC AVSA_ResetInterchangeCheck (ViSession vi);

【功能】

该函数忽略之前的所有配置操作。通过在测试开始时调用该函数可以确定是否测试系统与之前的测试操作有关系。好处是：当创建一个包含多个测试模块的复杂的测试系统时，通常会将这些测试模块设计为可以以任何顺序运行。这样做需要确保每个测试模块完全配置其使用的仪器。如果不能那么仪器状态依赖于前一次执行的测试系统对仪器的设置。如果按照不通顺序进行测试那么仪器的反应和整个测试系统会发生改变。这种改变就是一种可互换性的问题。.

注意：该函数并不会清除之前记录的可互换性警告。

【参数】

方向	参数名	说明

[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。

- AVSA_ ClearInterchangeWarnings

【方法原型】

ViStatus _VI_FUNC AVSA_ClearInterchangeWarnings (ViSession vi);

【功能】

清除记录的可互换性警告信息。

【参数】

方向	参数名	说明
[in]	vi	会话句柄。

【返回值】

执行状态。正常返回 VI_SUCCESS，否则返回其它值。