# Development Hints and Best Practices for Using Instrument Drivers Application Note

### **Products:**

| All Instrument Drivers

This document answers frequently asked questions regarding Rohde & Schwarz instrument drivers and their programming environments.



## Table of Contents

1	Preface 4
2	Frequently Asked Questions (FAQ)5
2.1	What is an instrument driver?5
2.2	Why should I use an instrument driver instead of SCPI commands?5
2.3	VISA and physical interfaces7
2.4	What are attribute based instrument drivers?7
2.5	What is the RsSpecAn instrument driver?8
2.6	Are R&S instrument drivers compatible with the VXIPnP driver standard?8
2.7	Where to find the VXIPnP installation directory?8
2.8	How to install R&S instrument drivers on Windows?8
3	Best practice for software development10
3.1	Getting started10
3.2	How to start to develop my own application?10
3.3	What is the purpose of the "Driver Attribute Help"?11
3.4	How to combine instrument manuals and the driver's compressed help files?.12
3.5	How to integrate R&S instrument drivers into Microsoft Visual Studio 200x12
3.5.1	C# and instrument drivers: How to get started?13
3.5.2	VB.NET and instrument drivers: How to get started?14
3.5.3	C++ and instrument drivers: General hints15
3.5.4	C++ and SCPI commands: How to get started?18
4	General programming hints
4.1	How to disable option checking?20
4.2	How to disable error/status checking?20
4.3	How to disable range checking?21
4.4	How to speed up the driver execution?21
4.5	How to use attributes in LabVIEW?21
4.5.1	Repeated capabilities24
4.6	How to use attributes in LabWindows/CVI?25
4.6.1	Repeated capabilities28
4.7	How to use attributes in Agilent VEE?29

5	RsSpecAn specific hints	30
5.1	How to run a scan measurement with the R&S <sup>®</sup> ESL EMI Test Receiver a RsSpecAn instrument driver?	and the 30
6	Linux instrument drivers	31
6.1	Supported distributions	31
6.2	How to view "Compressed HTLM Help Files" (chm) in Linux	31
6.3	How to install R&S instruments driver?	32
7	Resources	33

## 1 Preface

The aim of this paper is to provide information regarding Rohde & Schwarz (R&S) instrument drivers. This paper shall help application engineers, as well as software developers to easily get an understanding of advanced techniques to develop test and measurement (T&M) application by utilizing R&S instrument drivers. Furthermore the used nomenclature for R&S instrument drivers will be explained.

It does not describe any programming languages nor, how to develop software in any programming language.

Microsoft<sup>®</sup> and Windows<sup>®</sup> are U.S. registered trademarks of Microsoft Corporation.

National Instrument<sup>®</sup>, LabVIEW<sup>®</sup>, LabWindows/CVI<sup>®</sup> are U.S. registered trademarks of National Instrument.

MATLAB<sup>®</sup> is a trademark of The MathWorks, Inc.

Rohde & Schwarz<sup>®</sup> is a registered trademark of Rohde & Schwarz GmbH & Co. KG.

What is an instrument driver?

## 2 Frequently Asked Questions (FAQ)

## 2.1 What is an instrument driver?

An instrument driver, in the context of T&M application development, is a set of software routines, which simplify the remote control of your measurement instrument. Through the input and output (I/O) abstraction via hiding the SCPI<sup>1</sup> commands a much more easier way of programming is possible. As buzzword *command completion*<sup>2</sup> of integrated development environments (IDE) can be mentioned. This feature is not available for SCPI commands.

R&S instrument drivers are based on Virtual Instrument Software Architecture (VISA<sup>3</sup>) which enables an physical independent connection between the host PC and the instrument.

Generally instrument drivers allow greater system flexibility and accelerate development time for instance by the possibility of reusing software. Major advantage is the reduction of the development time for any kind T&M application due to the fact of using a high level programming language in contrast to implement an application with SCPI commands. For instance all string formatting which is necessary when using SCPI commands is already done by the instrument driver.

# 2.2 Why should I use an instrument driver instead of SCPI commands?

Writing an application can take a lot of time. But writing the application is not the end of the story, since the T&M instrument must also be driven. That is not so simple with complex equipment, since just the description of the command set may comprise several hundred pages. Applying yourself to this task can be time consuming. Which is why Rohde & Schwarz provides ready-to-use software instrument drivers, relieving designers of these efforts to great extent. It is no longer necessary to search through manuals looking for commands; this has already be done when developing the instrument driver.

<sup>&</sup>lt;sup>1</sup> Standard Commands for Programmable Instruments (SCPI) is a command set for remote controlling T&M instruments. Further information is available on http://www.scpiconsortium.org/

<sup>&</sup>lt;sup>2</sup> "Command completion" of current integrated development environments offers the possibility to safe time and avoid typing errors during programming (e.g. in C++ or C#) <sup>3</sup> Virtual Instrument Software Architecture (VISA) as Host PC software a well defined hardware abstraction layer and I/O programming interface for various interfaces (e.g. General Purpose Interface Bus (GPIB) or Ethernet)

Why should I use an instrument driver instead of SCPI commands?

#### Drawbacks of programming with SCPI commands:

For simple applications like setting frequency/RF level and reading out simple measurement values (e.g. for digital mulitmeters) SCPI programming can be sufficient. However for more sophisticated applications this approach reaches quite fast its limits.

#### Example: Simple spectrum analysis

Even for simple spectrum analysis it starts to be quite complicated and error prone using SCPI commands. For instance, all data formatting of a read out data stream has to be done manually when using SCPI commands. This formatting is already done by the instrument driver, so it is possible to easily process the read out data within the T&M application.

As one can imagine, for more complex T&M application, like mobile standard generation (e.g. for Wideband Code Division Multiple Access (WCDMA) Signals) or its measurement, using SCPI commands is quite time consuming. Therefore setting up your instrumentation using instrument drivers avoids a lot of string formatting and other data parsing. Moreover the local grouping of functions within the provided compressed HTML help file (chm file) offers you all configuration possibilities at a glance.

#### Development, maintenance and software reuse

Instrument drivers speed up your application development process. This can be achieved due to the fact that debugging and error diagnostics is well supported by the instrument driver. So it is possible to easily get rid of wrong configurations as well as not understood commands. This is done by extensively using the instruments error reporting functionality. For further details please have a look in chapter 4.

During the lifetime of T&M applications requirements can change, also for this reason instrument drivers can help you modifying a application. The logical grouping of your instruments functionality allows one easily to extend or modify your application on a abstract and logical level.

The described pros to simplify your T&M task is also a reason why reuse of already written software can be realised. Due to the concept of instrument family drivers, software written for research and development reasons on a high end instrument like the Vector Signal Generator R&S SMU200A can easily be reused for production lines which are equipped with e.g. the mid-range Vector Signal Generator R&S SMBV100A. The instrument driver abstracts the instrument, so this software can often be reused without any changes. Of cause the functionality needs to be supported on both instruments.

VISA and physical interfaces

## 2.3 VISA and physical interfaces

All R&S instrument drivers use VISA to avoid an individual driver for every different device interface. The VISA I/O software is generally a application programming interface (API) providing an input and output low level API to communicate with measurement instruments. This standard was defined by the VXIplug&play (VXIPnP) Systems Alliance, with National Instruments and others, being represented in the consortium.

These I/O functions are independent of the used device interface.

An initialization string (so called VISA resource locater) in the relevant application program defines which device interface (GPIB, RS-232, LAN, USB) to use. The subsequent program routine is completely independent of the used interface.

All Rohde & Schwarz instrument drivers requires at least National Instruments VISA (NI-VISA) revision 3.0 or higher. For Linux it is strongly recommended to use the latest version of NI-VISA. Also the usage of Agilent I/O Library Suite<sup>4</sup> is possible.

### 2.4 What are attribute based instrument drivers?

The concept of attribute based drivers is that nearly every SCPI command is accessible via an attribute. Its modularity and the possibility to access almost every instrument property via attributes enables a very sophisticated way of T&M application programming.

Attribute based drivers can easily be indentified: All attribute based instrument drivers has an additional *Driver Attribute Help* file (*rsXXX\_attr.chm*) within the drivers installation directory.

For using attributes to write more flexible application, please see chapter 4.5 and chapter 4.6 for further explanation.

It is important to realize that not every R&S instrument drivers is an attribute based driver.

<sup>&</sup>lt;sup>4</sup> For further information please visit http://www.agilent.com

What is the RsSpecAn instrument driver?

## 2.5 What is the RsSpecAn instrument driver?

The new generation of R&S attribute based spectrum analyzer instrument drivers (RsSpecAn) gives great flexibility and ease of use by combining low-level access to nearly every SCPI command via attribute and well balanced high level functions and VIs, combining several attributes. For those who don't want to learn new attribute concept, complete functionality of instrument is covered by high level commands. In other words, you may use attributes, but you don't have to. Of course it is possible to combine both low level attributes and high level functions together exactly according to needs of every specific application.

Comparing to older family instrument drivers (so called legacy drivers), the source code of both LabWindows/CVI and LabVIEW RsSpecAn instrument drivers are divided into modules/folders according to personalities. This gives opportunity to easily get rid of non-used functions and VIs.

# 2.6 Are R&S instrument drivers compatible with the VXIPnP driver standard?

Yes. For further information please have a look into the *readme.txt* in side your instrument driver's installation directory.

## 2.7 Where to find the VXIPnP installation directory?

This path is defined via an environment variable on your operating system, which can be read out in the environment variable "*VXIPNPPATH*".

For Windows based operating systems, this variable can easily read out via Windows command prompt using the command "*set*".

Note: With NI-VISA Version 4.2 the default installation path changed from C:\VXIPNP to C:\Program Files\IVI Foundation\VISA.

## 2.8 How to install R&S instrument drivers on Windows?

For installation hints please use the driver specific "How to use...", which can be downloaded at the instrument's driver download page:

http://www2.rohde-schwarz.com/en/service\_and\_support/Downloads/Drivers

How to install R&S instrument drivers on Windows?

Further specific steps and hints for installing instrument driver are available on the driver download site available at the product internet site.

Getting started

## 3 Best practice for software development

## 3.1 Getting started

To get started please download one of the provided application examples from the driver download site

http://www2.rohde-schwarz.com/en/service and support/Downloads/Drivers/

to get an idea of how a remote control application can look like.

It is recommended to use the provided instrument driver compressed HTML help files (chm files) for getting started.

## 3.2 How to start to develop my own application?

To getting started it is applicable to use the *Instruments Driver Help* (rsXXX[\_vxi].chm) file inside your instruments driver's installation directory in combination with our instruments operating manual remote control command chapter.

A good approach is to use the provided application examples as skeleton for new applications.

The easiest way to find the proper driver function call is to find the SCPI command (in the long version) inside the instrument operating manual and look for the command inside the *Instrument Driver Help* file using the index or search functionality.

For developing more complex applications it is practical to identify the manual operation steps and utilize the "Instrument Drivers Tree Structure" to find your driver function calls for the manual configuration steps in this logical structure.

Which information can be found inside the *Instrument Driver Help*, also *Attribute Driver Help* (*rsXXX[\_vxi].chm*)):

- Mapping of SCPI command and driver calls (and vice versa) via the search and index functionality
- Grouped driver functions to easily identify all functions for a certain 3GPP standard (Instrument Driver Tree Structure)
- Possibility to include instrument help to search for instrument and driver functionality at one time (Please note: this is not in available in every driver)
- For attribute based drivers (with rsXXX\_attr.chm): Mapping of attributes and driver calls (see figure below)

What is the purpose of the "Driver Attribute Help"?



Figure 1: Mapping of attributes and driver calls within the Driver Attribute Help.

For more information about the "Driver Attribute Help" please see chapter 3.4.

Note: This feature is not available in all R&S instrument drivers.

## 3.3 What is the purpose of the "Driver Attribute Help"?

For sophisticated programming the "Driver Attribute Help" as a chapter of the driver help file can help to identify instruments attributes. These low level configuration can be used to set or get instrument properties, which are normally encapsulated in high level driver functions. For some reasons this attributes need to be set or read out independently via get or set attribute functions.

How to use attributes is described in chapter 4.5 and chapter 4.6.

# 3.4 How to combine instrument manuals and the driver's compressed help files?

There is the possibility to include the "*Instrument Operating Manual*" inside the "*Instrument Driver Help*".

Please follow the instructions in the last chapter of your "*Instrument Driver Help*" file. See for instance the RsSpecAn help in the figure below:



Figure 2: Where to download and include the Instruments Operating Manual.

Depending on your Windows operating system, it can be necessary to open the instruments operating manual once, to mark the file as trustworthy for the operating system.

Note: This feature is not available in all R&S instrument drivers.

## 3.5 How to integrate R&S instrument drivers into Microsoft Visual Studio 200x

The following examples are referring to the RsSpecAn instrument driver. The described procedures are all adaptable to other R&S instrument drivers, only the naming of the files can be different. The naming convention is *rsXXX*, where *XXX* is the abbreviation of the instrument's (-family) driver.

#### 3.5.1 C# and instrument drivers: How to get started?

A wrapper is necessary to enable a direct access to the driver dynamic linked library (*dll*). This wrapper is automatically installed in the ~*VXIPnP\WinNT\include* directory, where *VXIPNPPATH* is your VXIPnP environment variable pointing to your VXIPnP installation directory.

#### How to create a new T&M application project in Microsoft Visual Studio 200x:

Create a new project, e.g. File -> New -> Project.



Figure 3: Create a new project in Visual Studio

Select your programming language and your template type, e.g. *Visual C# Windows Application*, or *Console Application*.

New Project			? ×
Project types: Visual C# Windows B: Smart Device - Database	Templates: Visual Studio installed templa Windows Application	tes @ Class Library @ Web Control Library	
Starter Kits Other Languages Other Project Types	Console Application C Empty Project My Templates	感 Windows Service 妃 Crystal Reports Application	
	Search Online Templates		

Figure 4: Select your desired template

To access the instrument driver, it is necessary to include the provided VXIPnP wrapper for *C*#, e.g. *Project->Add Existing Item...* ~*VXIPnP\WinNT\include\rsspecan.cs* 

Also the drivers namespace has to be included to the current source code file for example via the directive "using InstrumentDrivers". The result is shown in the picture below:

👯 WindowsApplication1 - Microsoft Visual Studio	
File Edit View Refactor Project Build Debug ClearCase Data Tools Window C	Community Help
🔚 🕶 🎫 🕶 🛃 🎒   🐰 ங 隆 🖉 🖃 🕫 🖛 Ray CPU	J 🔹 🖄 🗒
圓 % 🍆 🗤   華 華  🗏 😫   🗆 🗩 🗣 🖉 🛤 🧶 🐘 👷	
Form1.cs* Program.cs Form1.cs [Design]* Start Page	Solution Explorer 🗸 🕂 🗙
g 😤 WindowsApplication1.Form1	
4 using System.Data;	Solution 'WindowsApplicat
5 using System.Drawing;	+ Properties
6 using System.Text;	🖅 🔤 References
7 using System.Windows.Forms;	🗄 🗉 Form1.cs
8 - using InstrumentDrivers;	Program.cs
9 10 meneration Minders Ann Minetian 1	🔛 🕋 rsspecan.cs
10 mamespace windowsApplication1	

Figure 5: Include the RsSpecAn driver to your current project.

Now the development environment is configured in a proper way. For getting started your application development, please have a look at the provided examples on your instrument's driver download site:

http://www2.rohde-schwarz.com/en/service\_and\_support/Downloads/Drivers/.

#### 3.5.2 VB.NET and instrument drivers: How to get started?

A wrapper is necessary to enable a direct access to the driver dynamic linked library (*dll*). This wrapper is automatically installed in the ~*VXIPnP\WinNT\include* directory, where *VXIPNPPATH* is your VXIPnP environment variable pointing to your VXIPnP installation directory.

How to create a new T&M application project in Microsoft Visual Studio 200x:



Create a new project, e.g. File -> New -> Project

Figure 6: Create a new project in Visual Studio

Select your programming language and your template type, e.g. *Visual C# Windows Application*, or *Console Application*.



Figure 7: Select your desired template

Also the drivers namespace has to be included to the current source code file via the directive "*Imports WindowsApplication1.rsspecan*" (where *rsspecan* can be seen as example driver). The result is shown in the picture below:



Figure 8: Include the RsSpecAn driver to your current project.

Now the development environment is configured in a proper way. For getting started your application development, please have a look at the provided examples on your instrument's driver download site:

http://www2.rohde-schwarz.com/en/service and support/Downloads/Drivers/.

#### 3.5.3 C++ and instrument drivers: General hints

#### Problem: The header file rsXXX.h was not found while compiling

Solution: Please set your VXIPnP include directory in your Visual Studio Project properties (see picture below). Your include directory is the *\$VXIPNPPATH\$\WinNT\include*, where *VXIPNPPATH* is your VXIPnP environment variable pointing to your VXIPnP installation directory.

Configuration: Active(Debug)	Platform: Active(Win32)	Configuration Manager
Common Properties	Additional Include Directories	C:\VXIPNP\WinNT\include
Configuration Properties	Resolve #using References	
General	Debug Information Format	Program Database for Edit & Continue (/ZI)
Debugging	Suppress Startup Banner	Yes (/nologo)
⊡ · C/C++	Warning Level	Level 3 (/₩3)
General	Detect 64-bit Portability Issues	No
Optimization	Treat Warnings As Errors	No
Preprocessor	Use UNICODE Response Files	Yes
Advanced     Advanced     Command Line     Advanced     Command Line     Another the second sec	Additional Include Directories Specifies one or more directories to add t (/I[path])	o the include path; use semi-colon delimited list if more than one.
	L	OK Abbrechen Übernehmen

Figure 9: C++ Compiler settings for instrument driver header files.

#### Problem: Linker error while building your software project.

Solution: Please specify your VXIPnP libraries in your Visual Studio Project properties (see picture below). Your specific library *rsXXX.lib* can be found in *\$VXIPNPPATH\$\WinNT\ibrary\msc*, where *VXIPNPPATH* is your VXIPnP environment variable pointing to your VXIPnP installation directory.

Configuration: Active(Debug)	Platform: Active(Win32)	•	Configuration Manager
- Configuration Properties	Additional Dependencies	rsetl.lib visa32.lib	
General	Ignore All Default Libraries	No	
Debugging	Ignore Specific Library		
⊡. C/C++	Module Definition File		
General	Add Module to Assembly		
Optimization	Embed Managed Resource File		
Preprocessor	Force Symbol References		
- Code Generation	Delay Loaded DUIs		
Language	Assembly Link Resource		
Precompiled Headers			
Output Files			
Browse Information			
Advanced			
Command Line			
General			
Input Marcife at Cile			
Debugging			
Optimization			
Embedded IDI			
Advanced	Additional Dependencies		
	Specifies additional items to add to the link li	ine (ex: kernel32.lib); configuratio	n specific.
		ОК	Abbrechen Übernehmen

Figure 10: C++ Linker settings for instrument driver libraries.

## Problem: The library *rsXXX.lib* was not found while building your software project.

Solution: Please set your VXIPnP library path in your Visual Studio Project properties (see picture below). Your library directory is the *\$VXIPNPPATH\$\WinNT\library\msc*, where *VXIPNPPATH* is your VXIPnP environment variable pointing to your VXIPnP installation directory.

- Configuration Properties - General - Debugging	Output File Show Progress	\$(OutDir)\\$(ProjectName).exe
- General	Show Progress	
- Debuaaina		Not Set
	Version	
⊡ · C/C++	Enable Incremental Linking	Yes (/INCREMENTAL)
General	Suppress Startup Banner	Yes (/NOLOGO)
Optimization	Ignore Import Library	No
Preprocessor	Register Output	No
Code Generation	Per-user Redirection	No
Language	Additional Library Directories	c:\VXIPNP\WinNT\lib\msc
Precompiled Headers	Link Library Dependencies	Yes
Provide Tefermation	Use Library Dependency Inputs	No
- Browse Information	Use UNICODE Response Files	Yes
Command Line	Additional Library Directories Specifies one or more additional paths to delimited list if more than one. (/LIBPA	search for libraries; configuration specific; use semi-colon TH:[dir])

Figure 11: C++ Linker settings for instrument driver libraries.

## Problem: The Instrument was not found and VISA resource string in NI-Spy has only the size of one character.

Solution: Correct your character settings in your Visual Studio Project properties (see picture below).

Configuration: Active(Debug)		Platform: Active(Win32)				onfiguratio	on Manager
	E	General					
Configuration Properties		Output Directory	\$(Soluti	ionDir)\$(Config	uration	Name)	
General		Intermediate Directory	\$(Confi	gurationName)			
···· Debugging		Extensions to Delete on Clean	*.obj;*.i		*.tmp;*.	.rsp;*.pgc	;*.pgd;*.meta;s
±. C/C++		Build Log File	\$(IntDir)	\BuildLog.htm			
i Linker		Inherited Project Property Sheets					
⊕ Manifest Tool		Enable Managed Incremental Build	Yes				
. Resources	E	Project Defaults					
		Configuration Type	Applicatio	on (.exe)			
E Browse Information		Use of MFC	Use MFC	in a Shared DLL			
Ele Custom Build Step		Use of ATL	Not Using	g ATL			
		Character Set	Use Multi	i-Byte Character S	iet		-
		Common Language Runtime support	Not Set				
		Whole Program Optimization	Use Unico	ode Character Set			
			Use Multi-	-Byte Character S	et		
	C	haracter Set					
	Т	ells the compiler to use the specified chara	acter set; aids	s in localization issu	ues.		
				ОК	Abbr	echen	Übernehmen

Figure 12: Correct character settings for Visual Studio 2009

### 3.5.4 C++ and SCPI commands: How to get started?

This chapter shall help developers to set up their development environment to easily getting started without using instrument drivers.

After creating a new project, it is important to enable your development environment's linker to access the VISA library. The figures below are showing a possible configuration. Note that this depends on your host PC VXIPnP installation.

nfiguration: Active(Debug)	Platform: Active(Win32)	Configuration Manage
Common Properties Configuration Properties General Debugging C/C++ Linker General Display Command Line Display Generator Display Genera	Output File Show Progress Version Enable Incremental Linking Suppress Startup Banner Ignore Import Library Register Output Additional Library Directories Link Library Dependencies Use Library Dependencies Use UNICODE Response Files	\$(OutDir)\\$(ProjectName).exe Not Set Yes (/INCREMENTAL) Yes (/NOLOGO) No No C:\YXIPNP\WinNT\lib\msc Yes No Yes
	Output File Override the default output file name.	(/OUT:[file])

Figure 13: Setting up the "Additional Library Directory"

onfiguration: Active(Debug)	Platform: Active(Win32)		•	Configuration Manager.
🕂 Common Properties 📃	Additional Dependencies	visa32.lib		
Configuration Properties	Ignore All Default Libraries	No		
General	Ignore Specific Library			
Debugging	Module Definition File			
⊖ C/C++	Add Module to Assembly			
General	Embed Managed Resource File			
Optimization	Force Symbol References			
Preprocessor	Delay Loaded DLLs			
Code Generation	Assembly Link Resource			
Output Files     Browse Information     Advanced     Command Line     General     Input     Manifest File     Debugging     System     Optimization     Endeddet TD	Additional Dependencies			
Advanced V	Specifies additional items to add to the l	ink line (ex: kernel32.lib); (	configuration	specific.

Figure 14: Setting up linker parameter "Additional Dependencies"

## 4 General programming hints

The following is referring to the RsSpecAn instrument driver. The described procedures are all adaptable to other R&S instrument drivers, only the naming of the files can be different. The naming convention is *rsXXX*, where *XXX* is the abbreviation of the instrument's (-family) driver.

## 4.1 How to disable option checking?

In the LabWindows/CVI instrument driver the command is called:

In the LabVIEW instrument driver the command is called:

**e.g.** rsXXX Option Checking.vi

Where XXX is instrument driver specific. Please note that this function is not available in all instrument drivers.

Note: Inside the folder "*Instrument Driver Tree Structure->Utilities*" of the instrument driver's *chm* help file you can check, whether the command is available.

## 4.2 How to disable error/status checking?

In the LabWindows/CVI instrument driver the command is called:

**e.g.** rsXXX \_setCheckStatus (ViSession instrumentHandle, ViBoolean state);

In the LabVIEW instrument driver the command is called:

**e.g.** rsXXX Instrument Status Checking.vi

Where XXX is instrument driver specific. Please note that this function is not available in all instrument drivers.

Note: Inside the folder "*Instrument Driver Tree Structure->Utilities*" of the instrument driver's *chm* help file you can check, whether the command is available.

### 4.3 How to disable range checking?

While initializing a driver session, the range checking functionality can be disabled using an option string. Further details can be found inside the driver instrument *chm* help file.

Instead of using the <code>rsxxx\_Init (...)</code> function to initialize a driver session a second function is available. In the LabWindows/CVI instrument driver the function is called:

e.g. rsXXX\_InitWithOptions (ViRsrc resourceName, ViBoolean idQuery, ViBoolean resetDevice, ViString optionString, ViSession\* instrumentHandle);

Where XXX is instrument driver specific.

In the LabVIEW instrument driver the express VI *rsspecan\_core\_attribute\_express Source.vi* can be used to set the attribute *RS\_ATTR\_RANGE\_CHECK*. How to modify attributes in LabVIEW is described in chapter 4.5.

Note: Inside the folder "*Instrument Driver Tree Structure->Utilities*" of the instrument driver's *chm* help file you can check, whether the command is available. This function is not available in all instrument drivers.

### 4.4 How to speed up the driver execution?

In normal operation mode a status check is preformed after every command. It is strongly recommend to utilize this functionality to identify errors fast and reliable.

For improving execution time in many drivers it is possible to disable error status checking. To do so, please disable error/status checking, see chapter 4.2 for details.

### 4.5 How to use attributes in LabVIEW?

The following example is referring to the RsSpecAn instrument driver. The described procedures are all adaptable to other attribute based R&S instrument drivers, only the naming of the files can be different. The naming convention is *rsXXX*, where *XXX* is the abbreviation of the instrument's (-family) driver.

This chapter shows how to use attributes in the RsSpecAn driver. This feature is useful for sophisticated instrument control.

Please also use the *Driver Attribute Help* chm file. This help file is accessible via the *Instruments Driver Help* (rsXXX[\_vxi].chm).

#### Example: How to set the *Frequency Start* value by attributes

To select an attribute use the provided Express VI, included in instrument driver package. This instrument driver Express VI can be found in:

User Libraries => Express User Libraries => rsspecan



Figure 15: Palette Menu of RsSpecAn driver

Drag and drop the *rsspecan Attribute* in your block diagram. After placing the express VI the front panel of the Express VI will be opened (see Figure below).

Basic Operation		Attribute Help
<ul> <li>Markers</li> <li>System</li> <li>Reference Oscillator</li> <li>Trigger</li> <li>External Mixer</li> <li>Amplitude</li> <li>Limit Lines</li> <li>Display Control</li> <li>Hard Copy</li> <li>File</li> <li>Service</li> <li>Measurement</li> <li>Receiver</li> <li>B10 - External Generator Control</li> <li>B27 - FM Demodulation Analog</li> <li>B71 - Jogital Baseband Input</li> <li>K5 - GSM/EDGE</li> <li>K7 - FM Demodulator</li> </ul>		This class contains the attribute that configure option FSP-B6 - TV Trigger. (1) Trigger RF Power Level: This attribute sets the level of the RF power trigger source. (2) TV and RFP Trigger Source: This attribute selects the trigger source for the start of a sweep. (3) TV Trigger Free Run Mode: This attribute activates or deactivates the free run trigger mode. (4) TV Trigger Vertical Signal: This attribute activates triggering at the vertical sync signal. (5) TV Trigger Line System: This attribute defines the line system in use (525 or 625 lines) with active TV trigger. (6) TV Trigger Horizontal Sync Signal:
Attribute Operation Repeated Capabil	itie	5

Figure 16: Front panel of rsspecan\_core\_attribute\_express Source.vi

Now you can select the desired attribute from tree structured list. Each attribute can has a different access type (*read only, write only or read/write*). Proper *read/write* operation can be selected by "*Attribute Operation*" in the bottom left corner. Repeated capabilities are set by using a string control at the bottom of the panel called "*Repeated Capabilities*". The attribute value is entered using standard LabVIEW controls for each data type (numeric, boolean, string or ring).

For example use in Basic Operation "Set Frequency Start" value:

Basic Operation	Attribute Help
Amplitude Units	
Attenuation	Specifies the left edge of the frequency domain
Attenuation Auto	RSSPECAN ATTR ERECUENCY STOP Stop
Frequency Start	attribute to define the frequency domain. If the
Frequency Stop	RSSPECAN_ATTR_FREQUENCY_START attribute
Frequency Offset	Value is equal to the
Center Frequency	value then the spectrum analyzers horizontal
Center Frequency Step	attributes are in time-domain.
Center Frequency Step Auto	
Center Frequency Coupling	Available Repeated Capabilities:
Center Frequency Coupling Factor	Wildow. Wild, Will, Will2
Frequency Span	Remote-control command(s):
Full Frequency Span	[SENSe<1 2>]:FREQuency:STARt
Frequency Mode	
Input Impedance	
Number of Sweeps	
Reference Level	
Reference Level Offset	
Resolution Bandwidth	
Attribute Operation Repeated Capabi	lities Value
Write 💌	() 1E+9

Figure 17: Configure "Set Frequency Start" example

- 1. Select the desired attribute. To find the designated attribute easily the *rsspecan.chm* help file can be used to look for it. Be aware of the fact that the tree structures in the help file and the Express VI are the same
- 2. Leave "Repeated Capabilities" empty
- 3. Type the desired frequency value in the text box "Value"
- 4. After clicking the "*OK*" button, a new instance attributed based VI is generated. The name of the VI and its predefined values and also the help will be adapted.

The result is shown in following figure.

How to use attributes in LabVIEW?

	Context Help 🛛							
	Set Frequency Start ()							
	Input Double							
	Start () instrument handle Attribute ID							
Set Frequency Start () Instrument handle Attribute ID Repeated Capabil error in (no error) Instrument handle error out	This Express VI is configured as follows: Attribute ID: RSSPECAN_ATTR_FREQUENCY_START Repeated Capabilities: Attribute Value: IE+9 Specifies the left edge of the frequency domain in Hertz. This is used in conjunction with the RSSPECAN_ATTR_FREQUENCY_STOP Stop attribute to define the frequency domain. If the RSSPECAN_ATTR_FREQUENCY_START attribute value is equal to the RSSPECAN_BATTR_FREQUENCY_START attribute value is equal to the RSSPECAN_BATTR_FREQUENCY_START attribute value is equal to the REMOVE-control command(s): [SENSec1]=3;FREQuency:START							

Figure 18: Instanced attribute based VI "Set Frequency Start"

### 4.5.1 Repeated capabilities

Many measurement instruments have capabilities which are duplicated. For example, an oscilloscope might have several channels with identical functionality or a power supply may have several outputs. For this reason repeated capabilities are introduced.

Repeated capability instances are specified by a string parameter for each property. It also can be specified by a function which selects the active instance.

To define the usage of particular capability fill the text box "*Repeated Capability*" in the front panel of the Express VI with the proper value. To use more than one repeated capability in one attribute separate them with a comma.

See a marker example using repeated capabilities pictured in the following figure:

#### How to use attributes in LabWindows/CVI?

Basic Operation     Markers     Marker Amplitude     Marker State     Marker Position     Assign Marker to Trace     All Markers Off     Marker Local Oscillator Suppression     Signal Track State     Signal Track Bandwidth     Signal Track Bandwidth	Attribute Help This attribute switches on or off the currently selected marker in the selected measurement window. If no indication is made, marker 1 is selected automatically. If marker 2, 3 or 4 is selected automatically. If marker 2, 3 or 4 selected and used as a delta marker, it is switched to marker mode. Available Repeated Capabilities: Windows Windows Windows				
Marker Amplitude Marker State Marker State Assign Marker to Trace All Markers Off Marker Local Oscillator Suppression Signal Track State Signal Track State	This attribute switches on or off the currently selected marker in the selected measurement window. If no indication is made, marker 1 is selected automatically. If marker 2, 3 or 4 is selected and used as a delta marker, it is switched to marker mode. Available Repeated Capabilities: Windows Wind Mitol. Windows				
Marker Position Assign Marker to Trace All Markers Off Markers Local Oscillator Suppression Signal Track State Signal Track Bandwidth Signal Track Trackersched	selected automatically. If marker 2, 3 or 4 is selected and used as a delta marker, it is switched to marker mode. Available Repeated Capabilities:				
Marker Local Oscillator Suppression     Signal Track State     Signal Track Bandwidth     Signal Track Threshold	Available Repeated Capabilities:				
Signal Track Threshold	Marker: M1,M2,M3,M4				
Signal Track Trace	Remote-control command(s): CALCulate<1 2>:MARKer<1 to 4>[:STATe]				
Marker Probability Marker Receiver Frequency Coupling					
Marker Subscan Coupling					
Marker Search Configuration     Marker Functions     Delta Marker					

Figure 19: Example for "Repeated Capabilities"

The attribute and its features can be changed any time later by double clicking the VI, which recalls the shown Express VIs front panel.

### 4.6 How to use attributes in LabWindows/CVI?

The following example is referring to the RsSpecAn instrument driver. The described procedures are all adaptable to other attribute based R&S instrument drivers, only the naming of the files can be different. The naming convention is *rsXXX*, where *XXX* is the abbreviation of the instrument's (-family) driver.

The following chapter shows how to use attributes in the RsSpecAn driver. This is only necessary in rare cases, for instance if the driver is not supporting the functionality via high level API.

Please use the *Driver Attribute Help* chm file. This help file is accessible via the *Instruments Driver Help* (rsXXX[\_vxi].chm).

**Note:** The *rsspecan.sub* file has to be added to the project. This is mandatory for browsing attributes.

Example: How to set the *Frequency Start* value by attributes

To select an attribute use the provided function panels, included in instrument driver package. This instrument driver function panels can be found in:

Instruments=>R&S SpecAn=>Configuration=>Set/Get/Check Attribute/Repeated Capabilities



Figure 20: Function panel for RsSpecAn driver attributes

To open the function panel double click on the desired function. In the figure below the function panel of "*rsspecan\_SetAttributeViReal64*" is pictured. By clicking on text box "*'Attribute ID*" the "*Select Attribute Constant*" window will be opened.

Now it is possible to select the desired attribute from tree structured list. Each attribute can have different access type (*read only, write only or read/write*). Proper *read/write* operation can be selected by the proper function panel, either *Set Attribute*, *Get Attribute*, or *Check Attribute*.

Repeated capabilities are set by using identifier or identifier names within the function panel called *rsspecan\_GetAttributeRepeatedCapabilityId[s|Names]*.

For example use in Basic Operation "Set Frequency Start" value:

#### How to use attributes in LabWindows/CVI?



Figure 21: How to select a attribute in the "SetAttributeViReal64" function panel

- 1. Please select the desired attribute. To find the designated attribute easily the *rsspecan.chm* help file can be used to look for it. Be aware of the fact that the tree structures in the help file and the Express VI are the same
- 2. Leave "Repeated Capabilities" string empty
- 3. Type the desired frequency value in the text box "Attribute Value", e.g. "1e+9"
- 4. Enter a valid instrument handle in the text box "Instrument Handle"
- 5. It is strongly recommended to fetch the return value of your function, if you type the name of a declared integer variable in the text box "*Status*"
- 6. To include the function into your source code, you can copy the created function string out of the window on the button of the function panel and insert it into the source code.

The result is shown in following figure.

```
rsspecan_SetAttributeViReal64 (instrSession, "", RSSPECAN_ATTR_FREQUENCY_START, 1e+9);
```

Figure 22: Build a function string on the button of the "rsspecan\_SetAttributeViReal64" function panel

The attribute and its features can be changed any time later by right clicking the function and recall the function panel. See picture below:

3 4	rsspecan_Set	Ati	tributeViReal64	(instrSession,	۰.,	RSSPECAN_	_ATTR_	FREQUENCY	_START,	1e+
		58	Recall Function Panel	Ctrl+P						
		۲	Browse Identifier	Ctrl+F1						
	<	શ્ચ	Find <u>U</u> I Object	Ctrl+Shift+U						
		3	Open <u>Q</u> uoted Text	Ctrl+U						
		<b>*</b> > <sup>-</sup>	Colto Definition	Chilat						

Figure 23: Recalling the function panel to modify the attributes

#### 4.6.1 Repeated capabilities

Many instruments have capabilities which are duplicated. For example, an oscilloscope might have several channels with identical functionality or a power supply may have several outputs. For this reason repeated capabilities are introduced.

Repeated capability instances are specified by a string parameter for each property.

To define the usage of particular capability fill the text box "*Repeated Capability Id*" in the front panel with the proper value. To use more than one repeated capability in one attribute separate them with a comma.

See a marker example using repeated capabilities pictured in the following figure:

🖃 💊 Untitled	Untitled1.c* SetAttributeViReal64 GetAttributeRepeatedCapabilityIds GetAttributeRepeaCapabilityIdNames	×
E C Source Files		
-s.c rsidr_core.c		
.c rsspecan.c		
rsspecan attributes.c		
<u> </u>	Attribute ID Repeated Capability Id Buffer Size	
Diselau Centrel		
Transducer	Marker State M1, M2 256	
Hardsonu		
Calibration		
Liser Port	Repeated Capability Id Name(s)	
Set/Get/Check Attribute	(articharCtwing[NUM_OF_PED_CAD]	
- Set Attribute	any violation ing (kon_or_kon_on_	
rsspecan SetAttributeViInt32		
- 💯 rsspecan SetAttributeViReal64		
- 🖼 rsspecan_SetAttributeViString		
- 📴 rsspecan_SetAttributeViBoolean		
🖼 rsspecan_SetAttributeViSession		
	Stature	
	Instrument Handle	
Repeated Capability	instrSession status =	
-55 rsspecan_GetAttributeRepeatedCapabilityIds	11011000101	
rsspecan_GetAttributeRepeatedCapabilityIdNames		_
Measurement	status = rsspecan_GetAttributeRepeatedCapabilityIdNames (instrSession, Decempon and a state of the state of t	1
Receiver	256 & WUCHARSTING NUM OF REP CAPI)	륑
<u>e i i i i i i i i i i i i i i i i i i i</u>		5

Figure 24: Example for "Repeated Capabilities" in LabWindows/CVI

## 4.7 How to use attributes in Agilent VEE?

Due to the fact that Agilent VEE<sup>5</sup> does not parse the *sub*-file corresponding to the VXIPnP standard, there is only the possibility to look for the corresponding attributes<sup>6</sup> in the *rsspecan\_attr.chm* file and use the *Rohde&Schwarz Spectrum Analyzer-> Configuration -> Get/Set/CheckAttribute* function set.

<sup>&</sup>lt;sup>5</sup> Tested with Agilent VEE Pro 9.0, see http://www.home.agilent.com/agilent/product.jspx?nid=-34095.806312.00&cc=US&lc=eng

<sup>&</sup>lt;sup>6</sup> The following is referring to the RsSpecAn instrument driver. The described procedures are all adaptable to other attribute based R&S instrument drivers, only the naming of the files can be different. The naming convention is *rsXXX*, where *XXX* is the abbreviation of the instrument's (-family) driver.

How to run a scan measurement with the R&S®ESL EMI Test Receiver and the RsSpecAn instrument driver?

## 5 RsSpecAn specific hints

## 5.1 How to run a scan measurement with the R&S<sup>®</sup>ESL EMI Test Receiver and the RsSpecAn instrument driver?

Please find necessary commands the *rsspecan\_vxi.chm* help file. Crucial for a continuous reading of the scan trace is following function:

For LabVIEW applications the command is called

rsspecan Configure Receiver Trace Feed Control.vi

For further information please have a look at the scan measurement example application on the instrument's driver download site.

## 6 Linux instrument drivers

Due to the fact that the Linux as desktop operating system is becoming more and more popular Rohde & Schwarz supports instrument drivers for Linux distributions.

The aim of the this chapters is to give a brief overview of how to get started with R&S instruments drivers on a Linux operating system.

## 6.1 Supported distributions

Rohde & Schwarz instrument drivers are build up on the VISA standard and thereafter they are compatible to the ANSI-C99 standard<sup>7</sup>. National Instruments VISA installation package, which is crucial for using Rohde & Schwarz instrument drivers, is currently supporting<sup>8</sup> following distributions:

- Mandriva Linux 2008 and Mandriva Linux 2009
- openSUSE 10.3 and openSUSE 11.0

Rohde & Schwarz supports these distributions as well.

## 6.2 How to view "Compressed HTLM Help Files" (chm) in Linux

On Linux operating systems many viewers for Compressed HTML Help files available, for instance:

#### **KDE desktop environment:**

In the KDE environment the *kchmviewer*<sup>9</sup> is available.

<sup>&</sup>lt;sup>7</sup> For futher information please visit http://www.open-

std.org/JTC1/SC22/WG14/www/docs/n1256.pdf <sup>8</sup> For latest updates please visit http://www.ni.com/visa/

<sup>&</sup>lt;sup>9</sup> Available under http://www.kchmviewer.net/

### 6.3 How to install R&S instruments driver?

#### Getting started with LabVIEW 7 or higher:

Please be aware of the fact that for various instruments *dll*-based instrument driver are available (a), otherwise native LabVIEW driver are provided (b).

- a. For *dll*-based VXIPnP instrument driver please download the *rsXXX-linuxxxx.tar.gz* and the *rsXXX-linux-lvx-xxx.tar.gz* from the instrument's download site. Extract the *tar* files and follow the instructions of the *README* file.
- b. For native LabVIEW drivers it is only necessary to download the *rsXXX-linux-lvx-xxx.tar.gz* file. Thereafter extract the *tar* file and follow the instructions of the *README* file.

#### Getting started with C/C++:

Visit your instrument's driver site to download the *rsXXX-linux-xxx.tar.gz* (Linux Instrument Driver) *tar* file. Thereafter extract the *tar* file and follow the instructions of the *README* file.

#### How to install R&S instruments driver?

## 7 Resources

- Rohde & Schwarz Driver download site <a href="http://www2.rohde-schwarz.com/en/service\_and\_support/Downloads/Drivers">http://www2.rohde-schwarz.com/en/service\_and\_support/Downloads/Drivers</a>
- National Instruments VISA: <u>http://www.ni.com/visa/</u>
- National Instruments LabVIEW: <u>http://www.ni.com/labview/</u>
- National Instruments LabWindows/CVI: <u>http://www.ni.com/lwcvi/</u>
- Agilent VEE: www.agilent.com/find/vee
- IVI Foundation: <u>http://www.ivifoundation.org/</u>

Accessed: 05/27/2009

#### About Rohde & Schwarz

Rohde & Schwarz is an independent group of companies specializing in electronics. It is a leading supplier of solutions in the fields of test and measurement, broadcasting, radio monitoring and radiolocation, as well as secure communications. Established 75 years ago, Rohde & Schwarz has a global presence and a dedicated service network in over 70 countries. Company headquarters are in Munich, Germany.

#### **Regional contact**

Europe, Africa, Middle East +49 1805 12 42 42\* or +49 89 4129 137 74 customersupport@rohde-schwarz.com

North America 1-888-TEST-RSA (1-888-837-8772) customer.support@rsa.rohde-schwarz.com

Latin America +1-410-910-7988 customersupport.la@rohde-schwarz.com

Asia/Pacific +65 65 13 04 88 customersupport.asia@rohde-schwarz.com

Certified Quality System ISO 9001 DQS REG. NO 1954 QM

Certified Environmental System ISO 14001 DQS REG. NO 1954 UM

This document and the supplied programs may only be used subject to the conditions of use set forth in the download area of the Rohde & Schwarz website.

Rohde & Schwarz GmbH & Co. KG Mühldorfstraße 15 | D - 81671 München Phone + 49 89 4129 - 0 | Fax + 49 89 4129 – 13777

www.rohde-schwarz.com