Products: R&S® CRTU-G

Migration to Visual Studio .NET 2005 of proprietary Test Cases

on the GSM Protocol Tester R&S® CRTU-G

Application Note

Due to the fact that Microsoft does not maintain the Visual Studio .NET 2003 anymore Rohde & Schwarz has to use the successor compiler product Visual Studio .NET 2005 for building the test case packages. All CRTU-G customers having proprietary test case packages, developed based on Visual Studio .NET 2003, can migrate them to Visual Studio .NET 2005. This is required when the proprietary test case packages are going to be enhanced on a platform with installed Visual Studio .NET 2005.

ROHDE&SCHWARZ

Subject to change – Gerd Huber 04.2007 – Application Note 1CM63_4E

Contents

1 Overview

Due to the fact that Microsoft does not maintain the Visual Studio .NET 2003 anymore Rohde & Schwarz has to use the successor compiler product Visual Studio .NET 2005 for building the test case packages.

All CRTU-G customers having proprietary test case packages, developed based on Visual Studio .NET 2003, can migrate them to Visual Studio .NET 2005. This is required when the proprietary test case packages are going to be enhanced on a platform with installed Visual Studio .NET 2005.

The old Visual Studio .NET 2003 compiler can still be used for existing software and, until further notice, for future software releases by R&S for the CRTU-G. **However Rohde & Schwarz strongly recommends all CRTU-G customers to migrate to Visual Studio .NET 2005.**

The new Visual Studio .NET 2005 compiler can also be used for already released software and for future software released by R&S for the CRTU-G.

The GSM Protocol Tester R&S® CRTU-G is abbreviated as CRTU-G, Visual Studio .NET 2003/2005 is abbreviated as VS2003/2005 for the remainder of this Application Note.

2 Software Requirements

Compiler	Microsoft Visual Studio .NET 2005
TC Package	Proprietary GSM Test Case Package based on .NET 2003

3 Dev. Environment Migration Overview

In general the migration shall be transparent to the customer. This means in detail:

Both environments, VS2003 and VS2005 can be used. There is no need to upgrade existing VS2003. Both compilers can even be installed in parallel.

All new versions of the CRTU-G operational software as well as test packages will be compatible to the new and to the old environment.

VS2003 and VS2005 solution files (*:SLN) will be provided. Also for future software releases both solution files will be delivered until further notice.

There is one change according the distribution of VS2005:

The new environment will not be delivered anymore together with the CRTU-G . It must be ordered separately (R&S® CRTU-WP05).

All new CRTU-G operational software components will be provided created under VS2005. The following software releases and all successors are not available in a version compiled using the old environment:

- BP 2.00
- ASP 4.00
- EP 2.00

4 Possible Component Configurations

When migrating from VS2003 to VS2005, the following configurations of operational software components are possible:

Index	BP	ASP / KPACK.LIB	ApplComm / Test Package	Result
1	VS03	VS03	VS03	Ok
2	VS03	VS03	**VS05**	Ok
3	VS03	**VS05**	VS03	Ok*
4	VS03	**VS05**	**VS05**	Ok
5	**VS05**	VS03	VS03	Not tested**
6	**VS05**	VS03	**VS05**	Not tested**
7	**VS05**	**VS05**	VS03	Not tested**
8	**VS05**	**VS05**	**VS05**	Not tested**

*See section “Visual Studio 2003 Environment Adaptations”

**due to missing BP. The first BP compiled with VS05 will be BP 2.00. No problems are expected because BP consists only of DLLs and EXE files. There are no libraries to be linked.

5 Visual Studio 2003 Adaptations

In order to ensure that the new CRTU-G operational software components can be used/linked by VS2003 there are a view environment adaptations required. All the adaptations are transparent to the customer and done

automatically when installing a new ASP. The following environment changes are performed:

1. The run-time library path is extended by VC8\:

   Old path: $(VCInstallDir)lib

   New path: $(VCInstallDir)VC8\lib

2. A new directory is created where the new patch is pointing to. The contents of the directory are all the new VS2005 run-time libraries which are required for linking the new CRTU-G operational software.

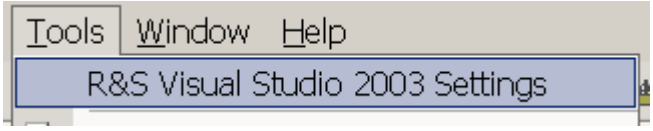
Configuration Tool for ASP 4.10

To be able to compile SW for ASP 4.10 with the Visual Studio 2003 it is required to extend the library path as described in the previous section. On the other hand, to be able to compile SW for former ASPs, the runtime library path must be reset to the original status. Together with ASP 4.10 a tool is installed which switches between these two states. The tool can be started by clicking on this icon:

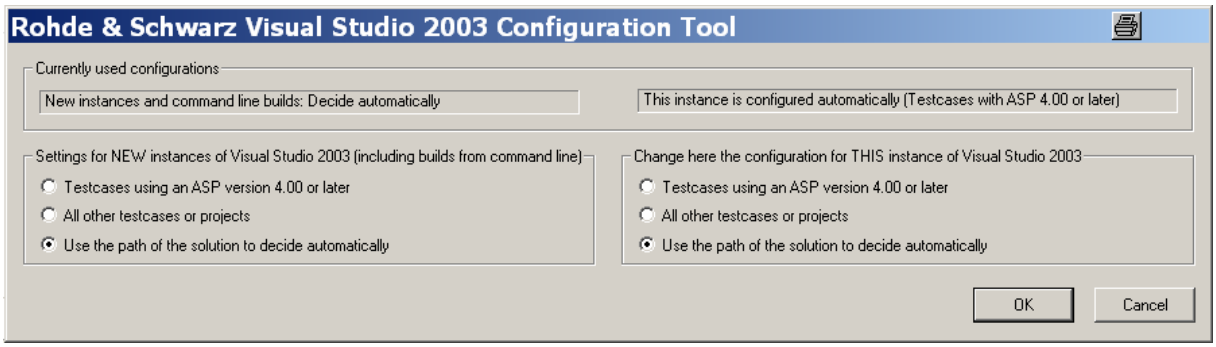
At installation of ASP 4.10 the tool configures an installed Visual Studio 2003 to compile for ASP 4.10.

Configuration Tool for ASP 4.11 and later

Since ASP 4.11 the Visual Studio .NET 2003 is extended by a new configuration tool which performs all required compiler settings automatically. Additionally it is possible to change the configuration manually by selecting the “R&S Visual Studio 2003 Settings” menu item in the “Tools” menu in the Visual Studio .NET 2003 IDE (see below).

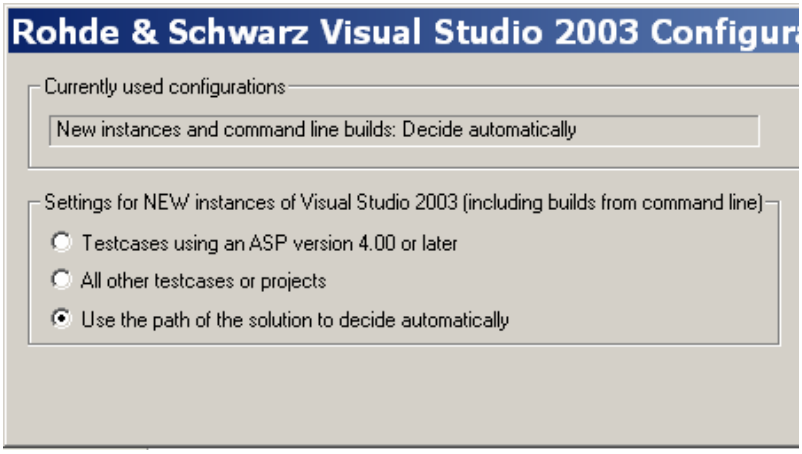


By selecting this item the following window will pop up:



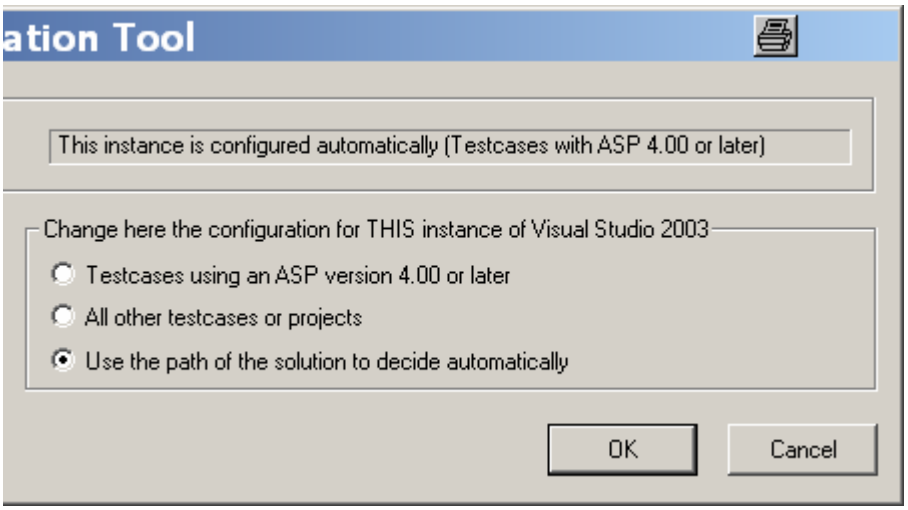
There are two sections. Each section uses a small text window at the upper part to show the currently used configuration and some radio buttons at the lower part to change the configuration.

The section on the left side is used for the default settings for all new instances of Visual Studio .NET 2003. This includes builds from the command line.



If selecting an item, the new value will be used for every instance of Visual Studio .NET 2003 that will start after pressing the OK button of this dialog. All currently running instances will not be affected by these settings.

The section on the right shows the settings for the currently used instance:



If selecting an item, the new value will be used for this instance of Visual Studio .NET 2003 only after pressing the OK button of this dialog. All other instances will not be affected by these settings.

In each section there are three possible values:

- **Testcases using an ASP version 4.00 or later**: this is a special configuration for ASP 4.00 and later (ASP built with Visual Studio .NET 2005). Visual Studio .NET 2003 can't handle such libraries without special preparations. So the installation of such an ASP installs some little helper files and this configuration uses theses files. In case it can't find the files it will display a warning and use the next available configuration
- **All other testcases or projects**: this is required for all pure Visual Studio .Net 2003 projects. That includes all testcases that are using an ASP before version 4.00.
- **Use the path of the solution to decide automatically**: this is the default and will select one of the two available configurations for Visual Studio .NET 2003 any time a solution is loaded. It looks at the path of a testcases solution and in case of it will find a path for an ASP version 4.00 or later it will use the extended configuration settings. In all other cases it will decide to use a normal configuration.

CAUTION: If you don't use the automation (third bullet) you should know what you are doing!

6 Procedure of Migration

This procedure describes how to create a second set of solution and project files, which can be built with Visual Studio .NET 2005.

General

After changing the compiler from VS2003 to VS2005 several problems can appear. Those can be sorted into three groups:

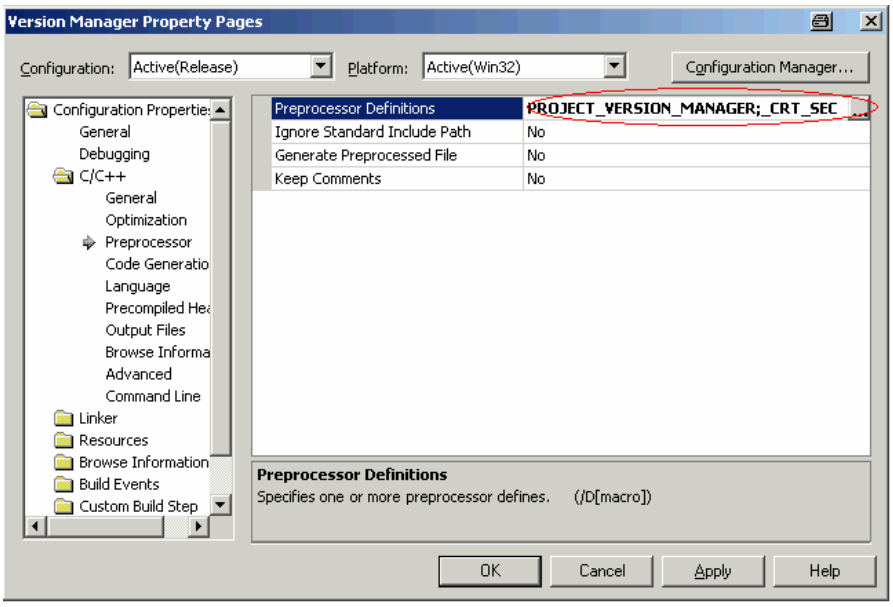
1. Compiler errors (see chapter "*Solutions for fixing frequent Compiler Errors*")
2. Security deprecate warnings

   These are proposals from Microsoft how to replace potentially unsafe ANSII C++ functions with proprietary MS functions. We do not recommend to implement this proposal for two reasons:

   - It would cause enormous migration effort without any performance or functionality improvements but introducing new sources of errors.
   - Non-ANSI compliant functions can be changed by MS any time, so a new migration might be required by another compiler change.
3. Style warnings

   These are caused by a more sensitive compiler and are pointing to wrong programming style, i.e. style which might cause several serious software errors like memory leaks. If solving those warnings the application stability can be improved, but having a perfectly tested working application the risk of any changes shall be well calculated for each case.

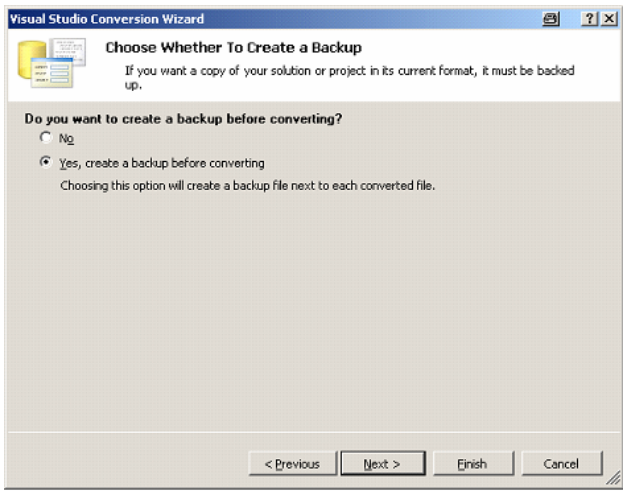
The warnings shall be switched off by the compiler switch *_CRT_SECURE_NO_DEPRACATE*, which shall be included in each project (this can be automated if many projects are affected) as shown in the following screen shot:



Convert Solution and Project Files

- Open the existing solution file (*.SLN) using Visual Studio .NET 2005. example: open CRTUGC31.sln
- The Visual Studio Conversion Wizard opens

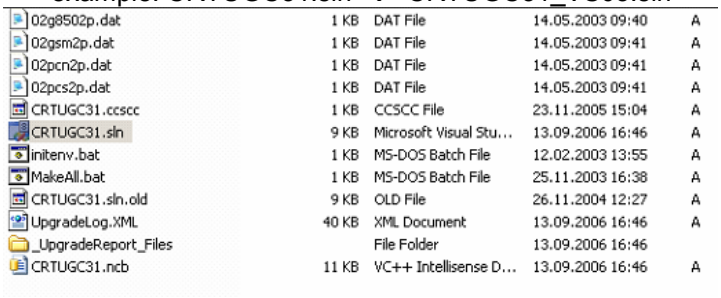
- Press the Next button
- Select 'Yes, create a backup before converting' and press the Next button



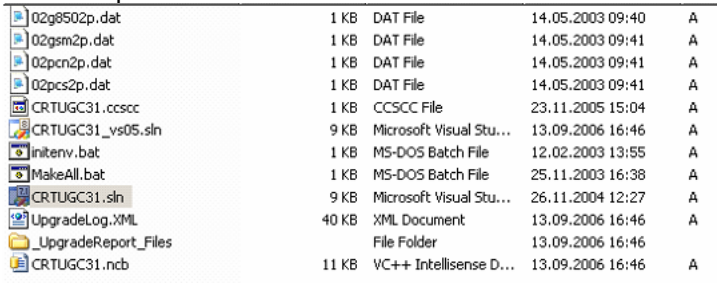
Rename Solution Files

Note: *This procedure is only required when it is intended still using the old *.SLN and *.VCPROJ files (.NET 2003) in parallel to the .NET 2005 files.*

- Add to the solution (*.SLN) file name the suffix '_VS05'
  example: CRTUGC31.sln → CRTUGC31_VS05.sln



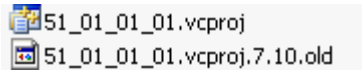
- Remove the extension 'old' from the file *.SLN.OLD
  example: CRTUGC31.sln.old → CRTUGC31.sln

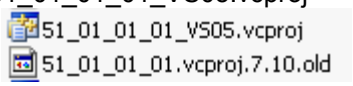


Rename Project Files

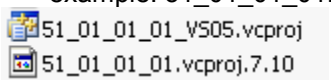
Note: *This procedure is only required when it is intended still using the old *.SLN and *.VCPROJ files (.NET 2003) in parallel to the .NET 2005 files.*

- Add to each project (*.VCPROJ) file name the suffix '_VS05'
  example: 51_01_01_01.vcproj → 51_01_01_01_VS05.vcproj





- Remove the extension 'old' from all files *.VCPROJ.7.10.OLD
  example: 51_01_01_01.vcproj.7.10.old → 51_01_01_01.vcproj.7.10



Note: *The project files of the used ApplComm integrated in this solution also have to be modified in the same way.*

Edit *_VS05.SLN Files

Note: *This procedure is only required when it is intended still using the old *.SLN and *.VCPROJ files (.NET 2003) in parallel to the .NET 2005 files.*

- Open the solution (*.SLN) file with the suffix '_VS05' with an editor (e.g. Notepad)
- Replace the old project references '*.VCPROJ' by '*_VS05.VCPROJ'

example: ‘51_01_01_01.vcproj’ → ‘51_01_01_01**_VS05**.vcproj’

```
# Visual Studio 2005

Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}")= "51_01_01_01",
"..\51\01\51_01_01_01.vcproj","{5420739A-EE2C-421A-9977-8E92A1D0F90C}"

...


# Visual Studio 2005

Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "51_01_01_01",
"..\51\01\51_01_01_01_VS05.vcproj","{5420739A-EE2C-421A-9977-8E92A1D0F90C}"

...
```

Warning LNK 4229 “invalid directive ‘/manifestdependency’ encountered; ignored"

This warning appears when linking a library created with Visual Studio .NET 2005 to an application using Visual Studion .NET 2003 and cannot be switched off.

Avoid ‘C4966 Warnings’

When compiling with Visual Studio .NET 2005 a lot of the following warnings appear:

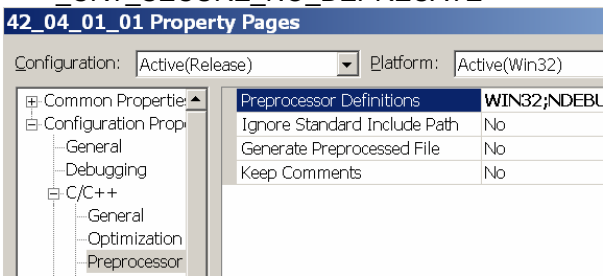
C4996: 'function': was declared deprecated

The code ‘**C4996**’ means, that the compiler encountered a function marked with deprecated. A deprecated function marked may no longer be supported in a future release. Visual Studio .NET 2005 offers a pre-compiler setting to disable this warning. Therefore you have to add the following entry to your preprocessor definitions.

_CRT_SECURE_NO_DEPRECATE

example:

- Open Visual Studio .NET 2005
- Open the solution file of CRTUGC74 (CRTUGC74_VS05.sln)
- Select the project 42_04_01_01 in the Solution Explorer
- Right mouse-click and select “Properties”
- Select ‘Configuration Properties → C/C++ → Preprocessor → Preprocessor Defintions’ and enter ‘_CRT_SECURE_NO_DEPRECATE’



Rework the Code

- Compile the test case package with the Visual Studio .NET 2005 solution file

example: CRTUGC31_VS05.sln

- Correct the errors

Note: *The chapter “Solutions for fixing frequent Compiler Errors”*

proposes solutions for frequent compiler errors.

Check if your modifications are backward compatible

Note: *This procedure is only required when it is intended still using the old *.SLN and *.VCPROJ files (.NET 2003) in parallel to the .NET 2005 files.*

- Compile the test case package using the old solution file (Visual Studio .NET 2003 should open)

Note: *Every time you use another compiler, you have to rebuild the whole solution, because otherwise linking of the libraries will not work.*

Solutions for fixing frequent Compiler Errors

C2668: 'function' : ambiguous call to overloaded function

The specified overloaded function call could not be resolved. You may want to explicitly cast one or more of the actual parameters.

You can also get this error through template use. If, in the same class, you have a regular member function and a templated member function with the same signature, the templated one must come first. This is a limitation of the current implementation of Visual C++.

Example: CRTUGC31 function pow()used:

```
int main()
{
...
NonDrxTimerValue = (tLongword)pow(2,
AppFnCtl.PagingInfo.DRXParameter.NonDrxTimer - 1)// C2668
...
NonDrxTimerValue = (tLongword)pow((long double)2,
         (long double)AppFnCtl.PagingInfo.DRXParameter.NonDrxTimer -
1)// OK
...
}
```

Note 1: `tLongword = unsigned long` → (long double) for best precision

Note 2: You have to cast both parameters to be able to compile the source with Visual Studio .NET 2003 and Visual Studio .NET 2005.

C4430: missing type specifier - int assumed. Note: C++ does not support default-int

This error can be generated as a result of compiler conformance work that was done for Visual C++ 2005: all declarations must now explicitly specify

the type; int is no longer assumed.

example: CRTKLU1 (SimAuto.c)

```
The line

static SimuUsed = LEGACY_MODE;    // Legacy SIM

changed to

static int SimuUsed = LEGACY_MODE;    // Legacy SIM
```

Note: The value is an enum so it should be better to select the enum type for this variable. But it is an unnamed enum so it is changed to int.

C2065: 'identifier' : undeclared identifier

A variable's type must be specified in a declaration before it can be used. The parameters that a function uses must be specified in a declaration, or prototype, before the function can be used. The most common reason for this error is:

Declaring an iterator variable in a **for** loop, and then trying to use that iterator variable outside the scope of the **for** loop.

example: CRTUGC08, Project 26_6_3_7, File 637.c

```
...
for(int i=0; i<8; i++)
{
...
}
...
for(int i=0; i<8; i++)
{
...
}
...
for(i=0; i<8; i++)  <- error C2065
```

Solution:

```
...
int i;
for(i=0; i<8; i++)
{
...
}
```

```
...

for(i=0; i<8; i++)  <- removed the “int” because of optimization

{

...

}

...

for(i=0; i<8; i++)  <- no error now
```

C2050: switch expression not integral

The **switch** expression evaluates to a non integer value. To resolve the error, use only integral values in switch statements.

This is a resulting error from C2065 (undeclared identifier). Because of the use of a variable that is outside its scope the switch can't evaluate the value of the variable.

example: CRTUGC08, Project 26_6_3_7, File 637.c

```
...
for(int i=0; i<8; i++)
{
...
}
...
switch(i)   <- error C2050
```

Solution: Refer to the description for error C2065. If you remove the reason for C2065 then C2050 should also disappear.

C2228: left of '.identifier' must have class/struct/union

The operand to the left of the period (.) is not a class, structure, or union.

This is a resulting error from C2065 (undeclared identifier). Because of the use of a variable that is outside its scope the same variable used as an index value can't be evaluated.

example: CRTUGC08, Project 26_6_3_7, File 637.c

```
...
for(int i=0; i<8; i++)
{
...
}
```

```
...

BcchCarrier[i].SysInfoType6   <- error C2228
```

Solution: Refer to the description for error C2065. If you remove the reason for C2065 then C2050 should also disappear.

Solutions for fixing frequent Run-Time Errors

Ensure that all memory allocated by the ASP is deallocated using the free() function, not delete[]. VS2003 compiler did not distinguish between free() and delete[], but for an ASP compiled with VS2005 (>= ASP 4.00) it is required to follow the rule: for malloc() use free(), for new user delete.

7 Additional Information

Please send any comments or suggestions about this application note to **TM-Applications@rsd.rohde-schwarz.com**.

8 Ordering Information

GSM Protocol Tester

CRTU-G	1140.0009.02

For additional information about GSM protocol testing, see the Protocol Testing Web area on the Rohde & Schwarz GLORIS website https://gloris.rohde-schwarz.com.

ROHDE & SCHWARZ GmbH & Co. KG · Mühldorfstraße 15 · D-81671 München · Postfach 80 14 69 · D-81614 München ·

Tel (089) 4129 -0 · Fax (089) 4129 - 13777 · Internet: http://www.rohde-schwarz.com