How to create a Hardcopy using R&S®Instrument Drivers

Application Note

Products:

- R&S®FMU
- R&S®FSG
- R&S®FSL
- R&S®ESL
- R&S®FSP
- R&S®FSQ
- R&S®FSU
- R&S®FSV
- R&S®FSUP
- R&S®ETL
- R&S®ZVL
- R&S®ZVA
- R&S®ZVB
- R&S®FSH
- R&S®CMU200
- R&S®CMU300
- R&S®CBT

This application note explains how to create screenshots of Rohde & Schwarz instruments using instrument drivers. The presented examples are written in C (LabWindows/CVI), C# programming language and LabVIEW

Jiri Kominek, Juergen Straub

12.2009-1MA160_1e

Application Note

ROHDE&SCHWARZ

Table of Contents

1 Preface

Most Rohde & Schwarz measurement instruments offers the possibility to directly print out hardcopies to a attached printer or to a file on the instruments hard disk.

This application note explains how to create screenshots of Rohde & Schwarz instruments using instrument drivers. The presented examples are written in C and C# programming language and G (LabVIEW). Development environment LabWindows/CVI, as well as Microsoft Visual Studio and LabVIEW 8.0 were used.

Rohde & Schwarz supports its customer with freely available instrument drivers to remote control Rohde & Schwarz measurement equipment. This enables the development of advanced test and measurement (T&M) application in a professional and time-saving manner.

The Rohde & Schwarz instrument driver[1] updates are scheduled to be released after the corresponding instrument firmware. A driver version is always tight to a specific firmware release, so it is possible to ensure maximum compatibility to the instrument firmware remote control function set.

[1] As referring Rohde & Schwarz instrument drivers, VXIplug&play drivers are targeted. This includes the LabVIEW, LabWindows/CVI and VXIplug&play instrument drivers available on the Rohde & Schwarz driver download site at http://www2.rohde-schwarz.com/en/service_and_support/Downloads/Drivers/.

2 Instrument drivers

The remote control function set of the Rohde & Schwarz measurement instruments includes a function subset to initiate a copy of the current screen of the measurement instrument via remote control connection. Furthermore a subset for transferring files from the instrument to the host PC is available allowing the transfer of screenshot images to the host PC.

This application note covers the following instruments:

- R&S®FMU
- R&S®FSG
- R&S®FSH4/8 (planned support end of 2009)
- R&S®FSL
- R&S®ESL
- R&S®FSP
- R&S®FSQ
- R&S®FSU
- R&S®FSV
- R&S®FSUP
- R&S®ETL
- R&S®FSH3/6/18
- R&S®ZVL
- R&S®ZVA
- R&S®ZVB
- R&S®CMU200
- R&S®CMU300
- R&S®CBT

The application note 1MA153: Development Hints and Best Practices for Using Instrument Drivers also covers instrument drivers. Its intention is to give extended information about instrument driver provided by Rohde & Schwarz.

2.1 How to select the instrument driver?

Nearly all high level programming languages are supported by Rohde & Schwarz via different instruments drivers. How to select the proper driver for your specific environment is described in this chapter.

The following table gives an overview of available and supported driver technologies at the Rohde & Schwarz website[2].

[2] Rohde & Schwarz supports the IVI-COM driver technology which is not listed here.

Development environment \ Driver technology		LabVIEW	LabWindows/CVI	VXIplug&play
Windows	LabVIEW	X		
	LabWindows/CVI		X	
	C#			X
	VB.Net			X
	C/C++			X
	VEE			X
	etc.			X
Linux	LabVIEW	X		
	C/C++			X
	etc.			X

Table 1: Overview of development environments and corresponding instrument drivers

3 Creating a hardcopy via remote control

This chapter shows how to create a hardcopy using the instrument (-family) specific driver. This can be done in two steps: First initiate a hardcopy of the instruments screen and save this as a file on the instruments hard disk and then transfer the created file to the attached PC.

The this chapter is categorized by the instrument driver's name. In the driver specific chapters programming examples and instrument specific features are described.

Furthermore two applications to create screen shots are mentioned for the R&S CMU200 as well as for the R&S FSH3/6/18 instrument.

General hints

- When using the instrument drivers, it is crucial to set a valid hardcopy file path and file name.
- The LabVIEW 7.1, LabWindows/CVI and Microsoft Visual Studio example projects are available at the instrument specific driver download site[3].
- To create a hardcopy by remote control, the instrument driver session must first be opened successfully and then the hardcopy device can be defined. Often at this stage the different file formats, page orientation, clipboard of the instruments and a connected printer can be set. In case of using a file as storage destination, a destination path and a destination file name has to be defined. Once these settings are complete, a copy of the screen can be initiated. The file created by this step can be downloaded via the remote control connection[4].

3.1 Source code availability

The complete source code of the following examples is available on the instrument specific driver download site. This site can be accessed via following link:

http://www2.rohde-schwarz.com/en/service_and_support/Downloads/Drivers/

[3] Fast access to the instrument drivers provides the driver download overview available on: http://www2.rohde-schwarz.com/en/service_and_support/Downloads/Drivers/

[4] The function calls for downloading files from the instrument to the PC can be found in the chapter "Instrument Driver Tree Structure->Utility Functions->Instrument I/O" within the provided *chm* help file, which is shipped with the instrument driver.

3.2 Common source code snippets used in the LabWindows/CVI code

This source code is already included in the provided example project.

CHECKERR(fcal)

Note that *CHECKERR(fcal)* is a simple macro which is defined as follows:

```
(1)   /* Useful macros */
(2)   #define CHECKERR(fCal) \
(3)   if (status = checkError((fCal)), status < VI_SUCCESS) \
(4)         return status; else
```

checkError(fCal)

checkError(fCal) is a simple error checking function as defined below:

If an error occurs the expression in line (8) will be true. Line (28) distinguishes between an instrument error or an error which recognized by the driver. The occurred error will be evaluated and show in a pop-up window afterwards.

```
(1)   ViStatus checkError (ViStatus status)
(2)   {
(3)   ViChar  error_message [STRING_LENGTH];
(4)   ViChar  error_buffer  [STRING_LENGTH*2];
(5)   ViChar* p2buf;
(6)   ViInt32 error;
(7)   /* Any error occurred? */
(8)   if (status < VI_SUCCESS)
(9)   {
(10)        /* Converts a status code returned by an instrument
(11)        driver function into a user-readable string */
(12)        rsXXX_error_message (instrSession, status,
(13)              error_message);
(14)
(15)        p2buf = error_buffer + sprintf (error_buffer,
(16)              "Primary Error: 0x%08X, %s\n", status,
(17)              error_message);
(18)
(19)        /* This function reads an error code and a message
(20)         From the instrument's error queue */
(21)
(22)        rsXXX_error_query (instrSession, &error,
(23)              error_message);
(24)
(25)        /* Enable the cursor */
(26)        SetWaitCursor (0);

(27)        /*Insturment error occurred*/
(28)        if (error != VI_SUCCESS)
```

```
(29)        {
(30)              sprintf (p2buf, "Secondary Error: 0x%08X,
(31)                    %s\n", error, error_message);
(32)        }
(33)        MessagePopup ("Error", error_buffer);
(34)
(35)        /* close session */
(36)        cleanup(VI_TRUE);

(37)        /* enable buttons for configuration */
(38)        EnableButtons(VI_TRUE);
(39)  }
(40)  return status;
(41)  }
```

readData(void)

The function *readData(void)* only transfers a file from the R&S instrument to the attached PC.

In the source code below line (13) specifies which file will be loaded from the instrument and stored on the host PC. In line (25) this file on the instrument will be deleted.

```
(1)   int readData (void)
(2)   {
(3)   ViStatus temp_status  = VI_SUCCESS;
(4)   ViStatus status       = VI_SUCCESS;
(5)   ViInt32  timeout      = 15000; /* default value */
(6)   ViReal64 data [2][ARRAY_SIZE];
(7)
(8)   /* Is save file specified? */
(9)   GetCtrlVal (panelHandle, PANEL_FILEPATH, filePath);
(10)  if (filePath[0] == NULL | !FileExists(filePath,0))
(11)  {
(12)        /* open or create new file or exit if canceled */
(13)        if (FileSelectPopup ("", "*.wmf", "*.wmf", "Name of
(14)                    File to Save", VAL_OK_BUTTON, 0, 1, 1,
(15)                    1, filePath) == 0)
(16)              return -1;
(17)        /* write file path to front panel text box */
(18)        SetCtrlVal (panelHandle, PANEL_FILEPATH, filePath);
(19)  }
(20)  /* copy file from instrument to controller */
(21)  CHECKERR (rsXXX_ReadToFileFromInstrument (instrSession,
(22)        tempFilePath, filePath));
(23)
(24)  /* delete temporary file */
(25)  CHECKERR (rsXXX_FileManagerOperations (instrSession,
(26)        RSSPECAN_VAL_FILE_DELETE, tempFilePath, ""));
(27)
```

```
(28)  return 0;
(29)  }
```

3.3 Rsspecan driver

The Rohde & Schwarz Spectrum Analyzer (*rsspecan*) Driver supports the following listed instruments:

Currently the R&S®FSU, FMU, FSG, FSH4/8 (end of 2009), FSL, FSP, FSQ, FSV and FSUP are supported.

LabWindows/CVI example
The following LabWindows/CVI code listing shows an example configuration to create a screenshot of the instrument display. This code covers all instruments supported by the *rsspecan* LabWindows/CVI instrument driver:

In this source code snippet it is important to pay attention to line (14) and (17). Due to a different file system in different instrument families a corresponding filename for the hardcopy image must be set. After the hardcopy is triggered in line (24) the generated image will be uploaded to the host PC. This is done in line (29) and described in chapter 3.2.

```
(1)   /* initialize driver */
(2)   CHECKERR (rsspecan_init (instr_desc, id, reset,
(3)      &instrSession));
(4)
(5)   /* configure hardcopy options */
(6)   CHECKERR (rsspecan_ConfigureHardcopyDevice (instrSession,
(7)      1,
(8)      RSSPECAN_VAL_HCOPY_DEST_WMF,
(9)      RSSPECAN_VAL_HCOPY_DEVICE_ORIENT_LAND));
(10)
(11)  /* select temporary file location */
(12)  if (instrType) /
(13)        /* R&S FSP, R&S FSQ family */
(14)        tempFilePath = "D:\\user\\output.wmf";
(15)  else
(16)        /* R&S FSL, R&S FSV family */
(17)        tempFilePath = "C:\\R_S\\instr\\user\\output.wmf";
(18)
(19)  /* configure hardcopy save file on the instruments hdd*/
(20)  CHECKERR (rsspecan_HardcopySetFileName (instrSession,
(21)     tempFilePath));
(22)
(23)  /* initiate a hardcopy on device "file" */
(24)  CHECKERR (rsspecan_HardcopyPrint (instrSession,
```

```
(25)     1,
(26)     RSSPECAN_VAL_HCOP_TRACE));
(27)
(28)  /* read data and save to file */
(29)  readData();
(30)
(31)  /* close session */
(32)  cleanup(VI_FALSE);
```

The function *cleanup(deviceClear)* simply destructs the driver session, see line (16). It also has capabilities to clear the remote control bus in case of pending operations. In that case line (9) clears the instruments input and output buffers.

```
(1)   void cleanup (ViBoolean deviceClear)
(2)   {
(3)   /* Device Clear is a low-level command and should be used
(4)    in case the instrument is waiting for Operation Complete
(5)    to cancel the wait. It is useful for instance in case of
(6)    incorrect external trigger when the instrument does not
(7)    respond to any other command because of waiting for
(8)    trigger.*/
(9)   if (deviceClear)
(10)  {
(11)        viClear (instrSession);
(12)        viPrintf (instrSession, "*CLS\n");
(13)  }
(14)
(15)  /* close instrument session */
(16)  rsspecan_close (instrSession);
(17)  instrSession = VI_NULL;
(18)
(19)  /* enable GUI */
(20)  EnableButtons (1);
(21)  /* enable mouse events*/
(22)  SetWaitCursor (0);
(23)
(24)  }
```

Microsoft C# example

The following C# code listing shows an example configuration to create a screenshot of the instrument display. This code covers all instruments supported by the *rsspecan* VXIplug&play instrument driver:

As stated before also in this source code snippet it is important to pay attention to line (21) and (25). Due to a different file system in different instrument families a corresponding filename for the hardcopy image must be set. After the hardcopy was triggered in line (31) the generated image will be uploaded to host PC. Also the file on the instrument will be deleted afterwards. This is done in line (44) to (52).

```
(1)   /* initialize driver */
```

```
(2)   m_instrument = new rsspecan(ResourceDescriptor.Text,
(3)         IDQuery.Checked, ResetDevice.Checked);
(4)   if (ResetDevice.Checked)
(5)         m_instrument.reset();
(6)
(7)   /* switch device display on/off */
(8)   m_instrument.ConfigureDisplayUpdate(
(9)         DisplayEnable.Checked);

(10)  /* configure hardcopy options */
(11)  m_instrument.ConfigureHardcopyDevice(
(12)        1,
(13)        rsspecanConstants.HcopyDestWmf,
(14)        rsspecanConstants.HcopyDeviceOrientLand,);
(15)
(16)  /* select temporary file location, depending to the
(17)  connected instrument type which is selected on the user
(18)  interface */
(19)
(20)  if (InstrType.SelectedIndex == 1)
(21)        /* R&S FSP, R&S FSQ family */
(22)        tempFilePath = "D:\\user\\output.wmf";
(23)  else
(24)        /* R&S FSV, R&S FSL family */
(25)        tempFilePath = "C:\\R_S\\instr\\user\\output.wmf";
(26)
(27)  /* configure hardcopy save file */
(28)  m_instrument.HardcopySetFileName(tempFilePath);
(29)
(30)  /* save screen image to file */
(31)  m_instrument.HardcopyPrint(
(32)        1,
(33)        rsspecanConstants.HcopAll);
(34)
(35)  if (File.Exists(textBox1.Text) == false)
(36)  {
(37)  if (saveFileDialog1.ShowDialog() == DialogResult.OK)
(38)        textBox1.Text = saveFileDialog1.FileName;
(39)  else
(40)        return;
(41)  }
(42)
(43)  /* copy file from instrument to controller */
(44)  m_instrument.ReadToFileFromInstrument(tempFilePath,
(45)        textBox1.Text);
(46)
(47)  /* delete temporary file */
(48)  m_instrument.FileManagerOperations(
(49)        rsspecanConstants.FileDelete,
(50)        tempFilePath,
```

```
(51)  "");
(52)
(53)  /* close session */
(54)  m_instrument.Dispose();
(55)  m_instruemnt = null;
```

LabVIEW example

The crucial part LabVIEW hardcopy example is shown in the picture below. As stated before also in this source code snippet it is important to pay attention to Figure 1. Due to a different file system in different instrument families a corresponding filename for the hardcopy image must be set.

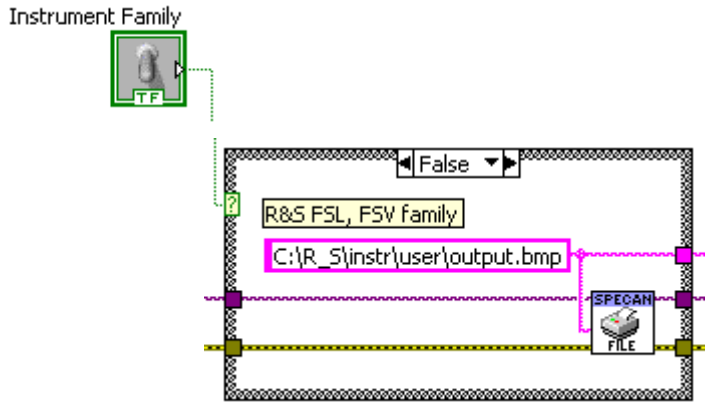

Figure 1: LabVIEW VI responsible for setting the filename of the hardcopy image

After the hardcopy was triggered in Figure 2 the generated image will be uploaded to host PC.

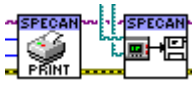

Figure 2: LabVIEW Vis responsible for triggering the hardcopy and uploading files to the host PC

Also the file on the instrument will be deleted afterwards. This is shown in Figure 3.

Figure 3: LabVIEW VI File Manager Operations

3.4 Rsetl driver

The *rsetl* driver also provides a function set for creating hardcopies of the instrument's display.

LabWindows/CVI example

The following LabWindows/CVI code listing shows an example configuration to create a screenshot of the instrument display. This code covers all instruments supported by the *rsetl* LabWindows/CVI instrument driver:

In this source code snippet it is important to pay attention to line (16). The filename of the hardcopy image must be set. After the hardcopy is triggered in line (23) the generated image will be uploaded to the host PC. This is done in line (28) and described in chapter 3.2.

```
(1)   /* initialize driver */
(2)   CHECKERR (rsetl_init (instr_desc, id, reset,
(3)         &instrSession));
(4)
(5)   /* switch device display on/off as set on the GUI */
(6)   CHECKERR (rsetl_ConfigureSANDisplayUpdate (instrSession,
(7)         display));
(8)
(9)   /* configure hardcopy options */
(10)  CHECKERR (rsetl_ConfigureSANHardcopyDevice (instrSession,
(11)     1,
(12)     RSETL_VAL_HCOPY_DEST_WMF,
(13)     RSETL_VAL_HCOPY_DEVICE_ORIENT_LAND));
(14)
(15)  /* select temporary file location */
(16)  tempFilePath = "C:\\R_S\\Instr\\user\\output.wmf";
(17)
(18)  /* configure hardcopy save file */
(19)  CHECKERR (rsetl_SANHardcopySetFileName (instrSession,
(20)        tempFilePath));
(21)
(22)  /* save screen image to file */
(23)  CHECKERR (rsetl_SANHardcopyPrint (instrSession,
(24)        1,
(25)        RSETL_VAL_HCOP_ALL));
(26)
(27)  /* read data and save to file */
(28)  readData();
(29)
(30)  /* close session */
(31)  rsetl_close(instrSession);
```

Microsoft C# example

The following C# code listing shows an example configuration to create a screenshot of the instrument display. This code covers all instruments supported by the *rsetl* VXIplug&play instrument driver:

As stated before also in this source code snippet it is important to pay attention to line (16). The filename of the hardcopy image must be set. After the hardcopy is triggered in line (22) the generated image will be uploaded to the host PC. Also the file on the instrument will be deleted afterwards. This is done in line (35) to (42).

```
(1)   /* initialize driver */
(2)   m_instrument = new rsetl(ResourceDescriptor.Text,
(3)         IDQuery.Checked, ResetDevice.Checked);
(4)
(5)   /* switch device display on/off as set on the GUI*/
(6)   m_instrument.ConfigureSANDisplayUpdate(
(7)         DisplayEnable.Checked);
(8)
(9)   /* configure hardcopy options */
(10)  m_instrument.ConfigureSANHardcopyDevice(
(11)        1,
(12)        rsetlConstants.HcopyDestWmf,
(13)        rsetlConstants.HcopyDeviceOrientLand);
(14)
(15)  /* select temporary file location */
(16)  tempFilePath = "C:\\R_S\\Instr\\useroutput.wmf";
(17)
(18)  /* configure hardcopy save file */
(19)  m_instrument.SANHardcopySetFileName(tempFilePath);
(20)
(21)  /* save screen image to file */
(22)  m_instrument.SANHardcopyPrint(
(23)        1,
(24)        rsetlConstants.HcopAll);
(25)
(26)  if (File.Exists(textBox1.Text) == false)
(27)  {
(28)        if (saveFileDialog1.ShowDialog() == DialogResult.OK)
(29)              textBox1.Text = saveFileDialog1.FileName;
(30)        else
(31)              return;
(32)  }
(33)
(34)  /* copy file from instrument to controller */
(35)  m_instrument.ReadToFileFromInstrument(tempFilePath,
(36)        textBox1.Text);
(37)
(38)  /* delete temporary file */
(39)  m_instrument.FileManagerOperations(
(40)        rsetlConstants.FileDelete,
(41)        tempFilePath,
(42)        "");
(43)
(44)  /* close session */
(45)  m_instrument.Dispose();
```

```
(46)  m_instruemnt = null;
```

LabVIEW example
The crucial part LabVIEW hardcopy example is shown in the picture below. As stated before also in this source code snippet it is important to pay attention to Figure 4. The filename of the hardcopy image must be set.

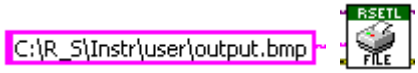

Figure 4: LabVIEW VI responsible for setting the filename of the hardcopy image

After the hardcopy was triggered in Figure 5 the generated image will be uploaded to host PC.

Figure 5: LabVIEW Vis responsible for triggering the hardcopy and uploading files to the host PC

Also the file on the instrument will be deleted afterwards. This is shown in Figure 6.

Figure 6: LabVIEW VI File Manager Operations

3.5 Rszvl driver

The approach to create a screen copy with the *rszvl* instrument driver is explained in this chapter.

LabWindows/CVI example
The following LabWindows/CVI code listing shows an example configuration to create a screenshot of the instrument display. This code covers all instruments supported by the *rszvl* LabWindows/CVI instrument driver:

In this source code snippet it is important to pay attention to line (24). The filename of the hardcopy image must be set. After the hardcopy is triggered in line (31) the generated image will be uploaded to the host PC. This is done in line (36) and described in chapter 3.2.

```
(1)   /* initialize driver */
(2)   CHECKERR (rszvl_init (instr_desc, id, reset,
(3)         &instrSession));
(4)
(5)   /* switch device display on/off */
(6)   CHECKERR (rszvl_ConfigureSANDisplayUpdate (instrSession,
```

```
(7)      display));
(8)
(9)   /* Set instrument to single sweep mode - continuous mode
(10)  is done programmatically because this is the only way how
(11)  to synchronize measurement using *OPC */
(12)
(13)  CHECKERR (rszvl_ConfigureSANAcquisition (instrSession,
(14)        VI_FALSE,
(15)        1)); /* number of sweeps */
(16)
(17)  /* configure hardcopy options */
(18)  CHECKERR (rszvl_ConfigureSANHardcopyDevice (instrSession,
(19)        1,
(20)        RSZVL_VAL_HCOPY_DEST_WMF,
(21)        RSZVL_VAL_HCOPY_DEVICE_ORIENT_LAND));
(22)
(23)  /* select temporary file location */
(24)  tempFilePath = "C:\\R_S\\Instr\\user\\output.wmf";
(25)
(26)  /* configure hardcopy save file */
(27)  CHECKERR (rszvl_SANHardcopySetFileName (instrSession,
(28)        tempFilePath));
(29)
(30)  /* save screen image to file */
(31)  CHECKERR (rszvl_SANHardcopyPrint (instrSession,
(32)        1,
(33)        RSZVL_VAL_HCOP_ALL));
(34)
(35)  /* read data and save to file */
(36)  readData();
(37)
(38)  /* close session */
(39)  cleanup(VI_FALSE);
```

Microsoft C# example

The following C# code listing shows an example configuration to create a screenshot of the instrument display. This code covers all instruments supported by the *rszvl* VXIplug&play instrument driver:

As stated before also in this source code snippet it is important to pay attention to line (16). The filename of the hardcopy image must be set. After the hardcopy is triggered in line (22) the generated image will be uploaded to the host PC. Also the file on the instrument will be deleted afterwards. This is done in line (35) to (42).

```
(1)   /* initialize driver */
(2)   m_instrument = new rszvl(ResourceDescriptor.Text,
(3)         IDQuery.Checked,
(4)         ResetDevice.Checked);
(5)
(6)   /* switch device display on/off */
```

```
(7)   m_instrument.ConfigureSANDisplayUpdate(
(8)         DisplayEnable.Checked);
(9)
(10)  /* configure hardcopy options */
(11)  m_instrument.ConfigureSANHardcopyDevice(1,
(12)        rszvlConstants.HcopyDestWmf,
(13)        rszvlConstants.HcopyDeviceOrientLand);
(14)
(15)  /* select temporary file location */
(16)  tempFilePath = "C:\\R_S\\Instr\\user\\output.wmf";
(17)
(18)  /* configure hardcopy save file */
(19)  m_instrument.SANHardcopySetFileName(tempFilePath);
(20)
(21)  /* save screen image to file */
(22)  m_instrument.SANHardcopyPrint(
(23)        1,
(24)        rszvlConstants.HcopAll);
(25)
(26)  if (File.Exists(textBox1.Text) == false)
(27)  {
(28)        if (saveFileDialog1.ShowDialog() == DialogResult.OK)
(29)              textBox1.Text = saveFileDialog1.FileName;
(30)        else
(31)              return;
(32)  }
(33)
(34)  /* copy file from instrument to controler */
(35)  m_instrument.ReadToFileFromInstrument(tempFilePath,
(36)        textBox1.Text);
(37)
(38)  /* delete temporary file */
(39)  m_instrument.FileManagerOperations(
(40)        rszvlConstants.filed,
(41)        tempFilePath,
(42)        "");
(43)
(44)  /* close session */
(45)  m_instrument.Dispose();
(46)  m_instruemnt = null;
```

LabVIEW example

The crucial part LabVIEW hardcopy example is shown in the picture below. As stated before also in this source code snippet it is important to pay attention to Figure 7. The filename of the hardcopy image must be set.

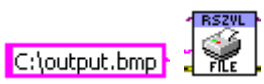

Figure 7: LabVIEW VI responsible for setting the filename of the hardcopy image

After the hardcopy was triggered in Figure 8 the generated image will be uploaded to host PC.

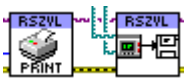

Figure 8: LabVIEW Vis responsible for triggering the hardcopy and uploading files to the host PC

Also the file on the instrument will be deleted afterwards. This is shown in Figure 9.

Figure 9: LabVIEW VI File Manager Operations

3.6 Rszvb driver

Creating a screen copy of the R&S®ZVA and ZVB is supported by the *rszvb* instrument driver. The function set is lightweight and easy to use as shown in the following example.

LabWindows/CVI example
The following LabWindows/CVI code listing shows an example configuration to create a screenshot of the instrument display. This code covers all instruments supported by the *rszvb* LabWindows/CVI instrument driver:

In this source code snippet it is important to pay attention to line (9). The filename of the hardcopy image must be set. After the hardcopy is configured and triggered in line (14). The generated image will be uploaded to the host PC. This is done in line (23) and described in chapter 3.2.

```
(1)   /* initialize driver */
(2)   CHECKERR (rszvb_init (instr_desc, id, reset,
(3)         &instrSession));
(4)
(5)   /* switch device display on/off */
(6)   CHECKERR (rszvb_SetDisplayUpdate(instrSession, display));
(7)
(8)   /* select temporary file location */
(9)   tempFilePath = "C:\\temp\\output.wmf";
(10)
(11)  /* configure hardcopy save file */
(12)  /* configure hardcopy options */
(13)  /* save screen image to file */
(14)  CHECKERR (rszvb_PrinttoFile(instrSession,
(15)        tempFilePath,
```

```
(16)        RSZVB_EMF,
(17)        RSZVB_HCOPY_ALL,
(18)        VI_TRUE,
(19)        VI_TRUE,
(20)        VI_TRUE));
(21)
(22)  /* read data and save to file */
(23)  readData();
(24)
(25)  /* close session */
(26)  rszvb_close(instrSession);
```

Microsoft C# example

The following C# code listing shows an example configuration to create a screenshot of the instrument display. This code covers all instruments supported by the *rszvb* VXIplug&play instrument driver:

As stated before also in this source code snippet it is important to pay attention to line (10). The filename of the hardcopy image must be set. After the hardcopy is configured and triggered in line (15) the generated image will be uploaded to the host PC. Also the file on the instrument will be deleted afterwards. This is done in line (31) to (37).

```
(1)   /* initialize driver */
(2)   m_instrument = new rszvb(ResourceDescriptor.Text,
(3)         IDQuery.Checked, ResetDevice.Checked);
(4)
(5)   /* switch device display on/off */
(6)   m_instrument.SetDisplayUpdate(
(7)         Convert.ToInt32(DisplayEnable.Checked));
(8)
(9)   /* select temporary file location */
(10)  tempFilePath = "C:\\temp\\output.wmf";
(11)
(12)  /* configure hardcopy save file */
(13)  /* configure hardcopy options */
(14)  /* save screen image to file */
(15)  m_instrument.PrinttoFile(tempFilePath,
(16)        rszvbConstants.Emf,
(17)        rszvbConstants.HcopyAll,
(18)        true,
(19)        true,
(20)        true);
(21)
(22)  if (File.Exists(textBox1.Text) == false)
(23)  {
(24)        if (saveFileDialog1.ShowDialog() == DialogResult.OK)
(25)              textBox1.Text = saveFileDialog1.FileName;
(26)        else
(27)              return;
(28)  }
```

```
(29)
(30)  /* copy file from instrument to controller */
(31)  m_instrument.readToFile(tempFilePath, textBox1.Text);
(32)
(33)  /* delete temporary file */
(34)  m_instrument.FileManager(
(35)     rszvbConstants.FileManDelete,
(36)     tempFilePath,
(37)     "");
(38)
(39)  /* close session */
(40)  m_instrument.Dispose();
(41)  m_instruemnt = null;
```

LabVIEW example

The crucial part LabVIEW hardcopy example is shown in the picture below. As stated before also in this source code snippet it is important to pay attention to Figure 10.The filename of the hardcopy image must be set. After the hardcopy was triggered in Figure 5 the generated image will be uploaded to host PC.

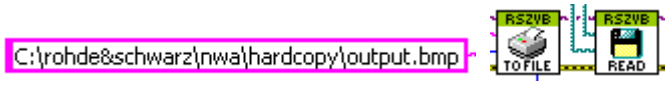

Figure 10: LabVIEW VI responsible for setting the filename and triggering of the hardcopy image

Also the file on the instrument will be deleted afterwards. This is shown in Figure 11.

Figure 11: LabVIEW VI File Manager Operations

3.7 R&S CMU200 hardcopy

The application note 1MA45: CMUCopy - Transferring CMU Hardcopy to PC using the IEEE Bus or the serial interface discusses hardcopies. It describes a program for transferring hardcopies of the screen from a R&S®CMU200/R&S®CMU300 Radiocommunication Tester / R&S®CBT Bluetooth Tester to a PC, connected via the IEEE bus or the serial interface.

In the application note 1MA74: RSCommander - Versatile Software Tool for Rohde & Schwarz Instruments a versatile software tool for the R&S®CMU200/R&S®CMU300 for making hardcopies and uploading data is described.

3.8 R&S FSH3/6/18 hardcopy

For the R&S®FSH3/6/18 the remote control software R&S®FSH View software is capable of creating screen copies. This software is available http://www2.rohde-schwarz.com/file/FSHView%20V13.12.zip.

It is possible with the R&S®FSH4/8 to create screen copies via the instrument driver. Please have a look at chapter 3.3, which describes the process of creating screen copies with the *rsspecan* instrument driver.

3.9 Further instruments

For further examples please contact the Rohde & Schwarz Support Center.

4 Related documents

The application note 1MA153: Development Hints and Best Practices for Using Instrument Drivers also covers instrument drivers. Its intention is to give extended information about instrument driver provided by Rohde & Schwarz.

An introduction to remote control programming of the R&S®CMU200 is given in 1MA157: Using R&S®CMU200 Drivers in Microsoft Visual Studio 2008 with Visual Basic .NET and C#

Creating hardcopies of the R&S®CMU200/CMU300 is described in 1MA45: CMUCopy - Transferring CMU Hardcopy to PC using the IEEE Bus or the serial interface.

In the application note 1MA74: RSCommander - Versatile Software Tool for Rohde & Schwarz Instruments a versatile software tool for a wide range of Rohde & Schwarz spectrum analyzers, signal generators and network analyzers for making hardcopies, reading traces and uploading user correction data for linearization is described.

The application GPIBShot is described in 1MA25: GPIBShot - Taking Screenshots via IEEE Bus. GPIBShot is a program for taking and storing screenshots from the Rohde & Schwarz FSx, ESx, ZVx and SMIQ instrument families.

5 References

Driver download area: http://www2.rohde-schwarz.com/en/service_and_support/Downloads/Drivers/

Application notes download area: http://www2.rohde-schwarz.com/en/service_and_support/Downloads/Application_Notes/

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, radiomonitoring 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

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

www.rohde-schwarz.com