Programmer's Guide

JETI Software Development Kit
jeti_radio.dll

Version 3.x

JETI Technische Instrumente GmbH
Tatzendpromenade 2
D-07745 Jena
Tel.: +49-3641-225680
Fax: +49-3641-225681
e-mail: sales@jeti.com
internet: www.jeti.com

Table of contents

1 JETI SDK Overview 3

2 Introduction 4

2.1 How to communicate 4

3 Function Reference 5

3.1 JETI_GetRadioDLLVersion 6
3.2 JETI_GetNumRadio 7
3.3 JETI_GetSerialRadio 8
3.4 JETI_OpenRadio 9
3.5 JETI_CloseRadio 10
3.6 JETI_Measure 11
3.7 JETI_PrepareMeasure 12
3.8 JETI_MeasureStatus 13
3.9 JETI_MeasureBreak 14
3.10 JETI_SpecRad 15
3.11 JETI_Radio 16
3.12 JETI_Photo 17
3.13 JETI_Chromxy 18
3.14 JETI_Chromxy10 19
3.15 JETI_Chromuv 20
3.16 JETI_ChromXYZ 21
3.17 JETI_DWLPE 22
3.18 JETI_CCT 23
3.19 JETI_Duv 24
3.20 JETI_CRI 25
3.21 JETI_SetMeasDist 26
3.22 JETI_GetMeasDist 27

4 Examples 28

4.1 C Examples 28
4.2 LabVIEW Examples 28
4.3 VisualBasic / VBA Examples 28

Appendix A 29

5 Service 30

1 JETI SDK Overview

The JETI Software Development Kit provides a complete software solution for interfacing spectrometric and radiometric devices from JETI Technische Instrumente GmbH. No firmware command expertise is required. Instead, a simple, high-level Application Program Interface (API) is used to provide complete connectivity. The API is provided in the form of several Windows Dynamic Link Libraries (DLL). The libraries can be used by any programming language that can handle DLL's such C/C++, VisualBasic, or LabVIEW.
To get access to the functions the needed DLL files have to be copied to the Windows System Folder or to the working directory of the calling application.

The following DLLs are available:

- jeti_spectro.dll / jeti_spectro64.dll
  - provides a set of functions for simple spectrometric measurement

- jeti_spectro_ex.dll / jeti_spectro_ex64.dll
  - a set of functions like jeti_spectro.dll, but with more options to control the measurement

- jeti_radio.dll / jeti_radio64.dll
  - provides a set of functions for simple radiometric measurement, including calculation of colorimetric values (e.g. xy- and u'v'-values, CCT, CRI,…)

- jeti_radio_ex.dll / jeti_radio_ex64.dll
  - a set of functions like jeti_radio.dll, but with more options to control the measurement and calculations

- jeti_core.dll / jeti_core64.dll
  - a set of functions to fully control the device and perform custom measurement sequences

Please note that this documentation describes only the functions provided by the* jeti_radio.dll*.
For description of the other DLL's please refer to the corresponding documents.

2 Introduction

The jeti_radio API is provided in the form of a Windows Dynamic Link Library (DLL). The interface DLL communicates with the device via the provided device driver and the basic driver DLL jeti_core.dll.
JETI Technische Instrumente GmbH offers two versions of the DLL. The first version is for 32bit Windows operating systems (WinXP / Windows Vista / Windows 7).
The second version is for real 64 bit programs under the 64 bit versions of Windows Vista and Windows 7.
There are no differences in the functionality between the two versions.

2.1 How to communicate

To get access to the functions you have to copy the files jeti_radio.dll and jeti_core.dll to the working directory of your application, or to the windows\system32 directory.
In general, the user initiates communication with the target device(s) by making a call to *JETI_GetNumRadio*. This call will return the number of target devices. This number is then used as a range when calling *JETI_GetSerialRadio* to build a list of device serial numbers.

To access a device, it must first be opened by a call to *JETI_OpenRadio* using an index determined from the call to *JETI_GetNumRadio*. The *JETI_OpenRadio* function will return a handle to the device that is used in all subsequent accesses. When I/O operations are complete, the device is closed by a call to *JETI_CloseRadio*.

In case of a fatal communication error (error code 0xFF) JETI_HardReset (from jeti_core.dll) could be used to reset the device and resume the communication. For more information see the function description of JETI_HardReset in 'JETI SDK Programmer's Guide jeti_core.dll' and the Appendix A.

3 Function Reference

Convention for calling : __stdcall

Type	Size in Bit	Minimum	Maximum
DWORD (unsigned long)	32	0	2^{32}-1
FLOAT (IEEE standard)	32	-3.40282E+38	3.40282E+38
BOOL (integer)	32	-2^{21}	2^{31}-1

3.1 JETI_GetRadioDLLVersion

This function returns the current version number of the jeti_radio DLL.

3.1.1 Prototype

DWORD JETI_GetRadioDLLVersion (WORD *wMajorVersion, WORD *wMinorVersion, WORD *wBuildNumber)

3.1.2 Parameters

Input

Name	Type	Description	Call
wMajorVersion	WORD *	address of a WORD variable that will contain the major version	By reference
wMinorVersion	WORD *	address of a WORD variable that will contain the minor version	By reference
wBuildNumber	WORD *	address of a WORD variable that will contain the build number	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.2 JETI_GetNumRadio

This function returns the number of JETI devices connected to the PC and to the LAN.

3.2.1 Prototype

DWORD JETI_GetNumRadio (DWORD *dwNumDevices)

3.2.2 Parameters

Input

Name	Type	Description	Call
dwNumDevices	DWORD *	address of a DWORD variable that will contain the number of devices connected	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.3 JETI_GetSerialRadio

This function returns the serial numbers for the device specified by an index passed in dwDeviceNum. The index for the first device is 0 and the last device is the value returned by *JETI_GetNumRadio* – 1.

3.3.1 Prototype

DWORD JETI_GetSerialRadio (DWORD dwDeviceNum, DWORD *dwSerial1, DWORD *dwSerial2)

3.3.2 Parameters

Input

Name	Type	Description	Call
dwDeviceNum	DWORD	index of the device for which the serial numbers are desired	By value
dwSerial1	DWORD *	address of a DWORD variable that will contain the first serial number	By reference
dwSerial2	DWORD *	address of a DWORD variable that will contain the second serial number	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.4 JETI_OpenRadio

Opens a device (using device number returned by *JETI_GetNumRadio*) and returns a handle which will be used for subsequent accesses.

3.4.1 Prototype

DWORD JETI_OpenRadio (DWORD dwDeviceNum, DWORD *dwDevice)

3.4.2 Parameters

Input

Name	Type	Description	Call
dwDeviceNum	DWORD	Device index. 0 for first device, 1 for second, etc.	By value
dwDevice	DWORD *	Pointer to a variable where the handle to the device will be stored	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.5 JETI_CloseRadio

Closes an open device using the handle provided by *JETI_OpenRadio*.

3.5.1 Prototype

DWORD JETI_CloseRadio (DWORD dwDevice)

3.5.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device to close as returned by *JETI_OpenRadio*	By value

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.6 JETI_Measure

Starts a radiometric measurement in the range of 380 to 780 nm with a step-width of 5 nm. The integration time will be determined automatically.

NOTE: The function will return ***immediately.*** Before any other DLL-function call the function *JETI_MeasureStatus* must be used to check if the measurement has finished.
Please note that a measurement could take several seconds up to 2 minutes, depending on the intensity of the light source to measure.

3.6.1 Prototype

DWORD JETI_Measure (DWORD dwDevice)

3.6.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.7 JETI_PrepareMeasure

Prepares a radiometric measurement in the range of 380 to 780 nm with a step-width of 5 nm. The integration time will be determined automatically.
It will not start the measurement.

NOTE: The function will return ***immediately.*** Before any other DLL-function call the function *JETI_MeasureStatus* must be used to check if the measurement has started by an external trigger.
Please note that a measurement could take several seconds up to 2 minutes, depending on the intensity of the light source to measure.

3.7.1 Prototype

DWORD JETI_PrepareMeasure (DWORD dwDevice)

3.7.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.8 JETI_MeasureStatus

Returns the status of a measurement started with *JETI_Measure*. A measurement has finished if the boStatus variable is FALSE (0). If the measurement is already in progress the variable boStatus returns TRUE (1).
If a measurement was initiated with automatic adaption of integration time and the measurement could not be performed because of overexposure boStatus will be switched to FALSE (0) and the function will return an error code 0x20.

NOTE: A function to get a measuring result should not be called until the *JETI_MeasureStatus* returns that the measurement has finished.

3.8.1 Prototype

DWORD JETI_MeasureStatus (DWORD dwDevice, BOOL *boStatus)

3.8.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
boStatus	BOOL *	Pointer to a variable where the status will be stored TRUE (1) – in progress FALSE (0) – ready	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.9 JETI_MeasureBreak

This function cancels an initiated measurement.

3.9.1 Prototype

DWORD JETI_MeasureBreak (DWORD dwDevice)

3.9.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.10 JETI_SpecRad

Returns the spectral radiance per wavelength in the range of 380 to 780 nm with a step-width of 5 nm. Please note that the array for the values must provide space for at least 81 values!

The unit of the values depends on the used measuring head. By default it's as follows:

Measuring Head	***Description***	***Unit***
none	spectral radiance	$W / sr \times m^2 \times nm$
cosine corrector head-piece	spectral irradiance	$W / m^2 \times nm$
integrating sphere	spectral radiant flux	W / nm
tube	spectral radiant intensity	$W / sr \times nm$

3.10.1 Prototype

DWORD JETI_SpecRad (DWORD dwDevice, FLOAT *fSprad)

3.10.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fSprad	FLOAT *	Pointer to an array where the spectral radiance values will be stored **(the array must provide space for at least 81 values)**	By reference

Return Value

Type	Description	
DWORD	0x00 0x..	JETI_SUCCESS see Appendix A for error codes

3.11 JETI_Radio

Returns the radiometric value determined by the last measure.

The unit of the value depends on the used measuring head. By default it's as follows:

Measuring Head	***Description***	***Unit***
none	radiance	$W/sr \times m^2$
cosine corrector head-piece	irradiance	W/m^2
integrating sphere	radiant flux	W
tube	radiant intensity	W/sr

3.11.1 Prototype

DWORD JETI_Radio (DWORD dwDevice, FLOAT *fRadio)

3.11.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fRadio	FLOAT *	Pointer to a variable where the radiometric value will be stored	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.12 JETI_Photo

Returns the photometric value determined by the last measure.

The unit of the value depends on the used measuring head. By default it's as follows:

Measuring Head	***Description***	***Unit***
none	luminance	cd/m^2
cosine corrector head-piece	illuminance	lx
integrating sphere	luminous flux	lm
tube	luminous intensity	cd

3.12.1 Prototype

DWORD JETI_Photo (DWORD dwDevice, FLOAT *fPhoto)

3.12.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fPhoto	FLOAT *	Pointer to a variable where the photometric value will be stored	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.13 JETI_Chromxy

Returns the CIE-1931 chromaticity coordinates x and y determined by the last measure. The calculation is based on a 2° observer, and the wavelength range for calculation is 380…780 nm.

3.13.1 Prototype

DWORD JETI_Chromxy (DWORD dwDevice, FLOAT *fChromx, FLOAT *fChromy)

3.13.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fChromx	FLOAT *	Pointer to a variable where the x-value will be stored	By reference
fChromy	FLOAT *	Pointer to a variable where the y-value will be stored	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.14 JETI_Chromxy10

Returns the CIE-1931 chromaticity coordinates x and y determined by the last measure. The calculation is based on a 10° observer, and the wavelength range for calculation is 380…780 nm.

3.14.1 Prototype

DWORD JETI_Chromxy10 (DWORD dwDevice, FLOAT *fChromx10, FLOAT *fChromy10)

3.14.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fChromx10	FLOAT *	Pointer to a variable where the x-value will be stored	By reference
fChromy10	FLOAT *	Pointer to a variable where the y-value will be stored	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.15 JETI_Chromuv

Returns the CIE-1976 chromaticity coordinates u' and v' determined by the last measure. The calculation is based on a 2° observer, and the wavelength range for calculation is 380…780 nm.

3.15.1 Prototype

DWORD JETI_Chromuv (DWORD dwDevice, FLOAT *fChromu, FLOAT *fChromv)

3.15.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fChromu	FLOAT *	Pointer to a variable where the u'-value will be stored	By reference
fChromv	FLOAT *	Pointer to a variable where the v'-value will be stored	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.16 JETI_ChromXYZ

Returns the tristimulus XYZ determined by the last measure. The calculation is based on a 2° observer, and the wavelength range for calculation is 380…780 nm.

3.16.1 Prototype

DWORD JETI_ChromXYZ (DWORD dwDevice, FLOAT *fX, FLOAT *fY, FLOAT *fZ)

3.16.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fX	FLOAT *	pointer to a variable where the tristimulus X will be stored	By reference
fY	FLOAT *	pointer to a variable where the tristimulus Y will be stored	By reference
fZ	FLOAT *	pointer to a variable where the tristimulus Z will be stored	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.17 JETI_DWLPE

Returns the dominant wavelength (DWL) and color purity (PE) determined by the last measure. The calculation is based on a 2° observer, and the wavelength range for calculation is 380…780 nm.
The unit for the dominant wavelength is [nm].
The unit for the color purity is [%] (percent).

3.17.1 Prototype

DWORD JETI_DWLPE (DWORD dwDevice, FLOAT *fDWL, FLOAT *fPE)

3.17.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fDWL	FLOAT *	Pointer to a variable where the dominant wavelength will be stored	By reference
fPE	FLOAT *	Pointer to a variable where the color purity will be stored	By reference

Return Value

Type	Description	
DWORD	0x00	JETI_SUCCESS
	0x…	see Appendix A for error codes

3.18 JETI_CCT

Returns the correlated color temperature (CCT) determined by the last measure.

3.18.1 Prototype

DWORD JETI_CCT (DWORD dwDevice, DWORD *dwCCT)

3.18.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
dwCCT	DWORD *	Pointer to a variable where the correlated color temperature value will be stored	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.19 JETI_Duv

Returns the Δuv for the correlated color temperature determined by the last measure.

3.19.1 Prototype

DWORD JETI_Duv (DWORD dwDevice, FLOAT *fDuv)

3.19.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fDuv	FLOAT *	pointer to a variable where the Δuv value will be stored	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.20 JETI_CRI

Returns the color rendering indices according to CIE 13.3-1995 publication determined by the last measure. The color temperature of the reference source is the same as the calculated CCT.
The function returns an array of 16 values containing the different CRI-values.
The first value (index 0) contains the chromaticity difference (DC) between the lamp to be tested and the reference illuminant. If DC is greater than 0.0054 the resulting color rendering indices may become less accurate.
The second value (index 1) contains the general color rendering index which is the arithmetical mean of eight special color rendering indices for the CIE-1974 test-color samples Nos. 1…8.
Value number three (index 2) to value number 16 (index 15) contains the special color rendering indices.

3.20.1 Prototype

DWORD JETI_CRI (DWORD dwDevice, FLOAT *fCRI)

3.20.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
fCRI	FLOAT *	Pointer to an array where the CRI-values will be stored **(the array must provide space for at least 16 values)**	By reference

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.21 JETI_SetMeasDist

This function sets the measuring distance which is used to calculate the values in intensity measuring mode.

3.21.1 Prototype

DWORD JETI_SetMeasDist (DWORD dwDevice, DWORD dwDistance)

3.21.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
dwDistance	DWORD	the measuring distance in [mm]	By value

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

3.22 JETI_GetMeasDist

This function gets the measuring distance which is used to calculate the values in intensity measuring mode.

3.22.1 Prototype

DWORD JETI_GetMeasDist (DWORD dwDevice, DWORD *dwDistance)

3.22.2 Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenRadio*	By value
dwDistance	DWORD *	pointer to a variable where the measuring distance in [mm] will be stored	By value

Return Value

Type	Description	
DWORD	0x00 0x…	JETI_SUCCESS see Appendix A for error codes

4 Examples

To help starting development the SDK includes several examples.

4.1 C Examples

4.1.1 RadioSample

This sample demonstrates the basic usage of the jeti_radio DLL.

4.1.2 SyncSample

The SyncSample demonstrates the use of special functions to synchronize the measurements integration time with the frequency of pulsed light sources and pulsed monitor back-lights.

4.1.3 TriggerSample

This sample demonstrates the handle of measurements initiated by an external trigger event.

4.2 LabVIEW Examples

These samples demonstrate the basic usage of the DLLs within a LabVIEW program.

4.3 VisualBasic / VBA Examples

These sample demonstrate the usage of the jeti_radio DLL within a VBA macro inside an excel spreadsheet.

Appendix A

Error codes and their description:

Error code	*#define*	*Description*
0x00	JETI_SUCCESS	no error occured
0x02	JETI_ERROR_OPEN_PORT	could not open COM-port
0x03	JETI_ERROR_PORT_SETTING	could not set COM-port settings
0x04	JETI_ERROR_BUFFER_SIZE	could not set buffer size of COM-port
0x05	JETI_ERROR_PURGE	could not purge buffers of COM-port
0x06	JETI_ERROR_TIMEOUT_SETTING	could not set COM-port timeout
0x07	JETI_ERROR_SEND	could not send to device
0x08	JETI_TIMEOUT	communication timeout error
0x0A	JETI_ERROR_RECEIVE	could not receive from device
0x0B	JETI_ERROR_NAK	command not supported or invalid argument
0x0C	JETI_ERROR_CONVERT	could not convert received data
0x0D	JETI_ERROR_PARAMETER	invalid argument
0x0E	JETI_BUSY	device busy
0x11	JETI_CHECKSUM_ERROR	invalid checksum of received data
0x12	JETI_INVALID_STEPWIDTH	invalid step width
0x13	JETI_INVALID_NUMBER	invalid device number
0x14	JETI_NOT_CONNECTED	device not connected
0x15	JETI_INVALID_HANDLE	invalid device handle
0x16	JETI_INVALID_CALIB	invalid calibration file number
0x17	JETI_CALIB_NOT_READ	calibration data not read
0x20	JETI_MEASURE_FAIL	measurement failed (overexposure)
0x80	JETI_DLL_ERROR	internal DLL error
0xFF	JETI_FATAL_ERROR	fatal communication error
0x2710...0x2AFC		Windows sockets error codes

If a fatal communication error occurs (error code 0xFF) there are several ways to solve the problem.

1) Call JETI_HardReset (from jeti_core) to perform a device hardware reset. The effect of this function is the same as disconnecting then reconnecting the device from USB. This will work only if the device uses an FTDI USB-to-serial converter and was opened with direct access to the FTDI driver (opened with JETI_GetNumRadio and JETI_OpenRadio) instead of using the VCP (virtual com port) driver (JETI_OpenCOMDevice and/or JETI_SetComSearch).
Please note that all custom settings (e.g. integration time, function,...) will be set to default values and have to be repeated.

2) Closing the device with JETI_CloseRadio will also perform a hardware reset if a fatal communication error occurred on a device with FTDI USB-to-Serial converter. After closing the device it should be possible to reopen the device with JETI_GetNumRadio and JETI_OpenRadio.

3) If a JETI device with FTDI USB-to-Serial converter was opened using VCP driver (e.g. by using JETI_OpenCOMDevice) or by using other connections (like RS232, bluetooth,...) a fatal communication error can only be resolved by closing the device with JETI_CloseRadio and manually reset the device.

5 Service

In case of any questions or technical problems please contact:

JETI Technische Instrumente GmbH
Tatzendpromenade 2
D-07745 Jena
Tel. +49 3641 225 680
Fax +49 3641 225 681
e-mail: sales@jeti.com
Internet: www.jeti.com

Copyright (c) 2012 JETI Technische Instrumente GmbH. All rights reserved.

Software and operating instruction are delivered with respect to the License agreement and can be used only in accordance with this License agreement.

The hard and software as well as the operating instruction are subject to change without notice. JETI Technische Instrumente GmbH assumes no liability or responsibility for inaccuracies and errors in the operating instruction.

It is not allowed to copy this documentation or parts of it without previous written permission by JETI Technische Instrumente GmbH.

June 13 , 2012