Programmer's Guide

JETI Software Development Kit
jeti_core.dll

Version 4.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

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_core.dll*.
For description of the other DLL's please refer to the corresponding documents.

2 Introduction

The jeti_core 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.
JETI Technische Instrumente GmbH offers two versions of the DLL. The first version is for 32bit Windows operating systems (Windows 7/8/8.1/10).
The second version is for real 64bit programs under the 64bit versions of Windows 7/8/8.1/10.
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 file 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_GetNumDevices*. This call will return the number of target devices. This number is then used as a range when calling *JETI_GetSerialDevice* to build a list of device serial numbers.

To access a device, it must first be opened by a call to *JETI_OpenDevice* using an index determined from the call to *JETI_GetNumDevices*. The *JETI_OpenDevice* 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_CloseDevice*.

In case of a fatal communication error (error code 0xFF) JETI_HardReset could be used to reset the device and resume the communication. For more information see the function description of JETI_HardReset and the Appendix A.

3 Function Reference

Convention for calling : __stdcall

Type	Size in Bit	Minimum	Maximum
DWORD (unsigned long integer)	32	0	2^{32}-1
DWORD_PTR (unsigned long integer) (unsigned long long)	 32 (32bit DLLs) 64 (64bit DLLs)	 0 0	 2^{32}-1 2^{64}-1
WORD (unsigned short integer)	16	0	65535
FLOAT (IEEE standard)	32	-3.40282E+38	3.40282E+38
BOOL (long integer)	32	-2^{21}	2^{31}-1
BYTE (unsigned char)	8	0	255

3.1 Device Handling

3.1.1 JETI_GetNumDevices

This function searchs automatically all JETI devices with FTDI USB-to-Serial converter and all other JETI devices connected to COM ports (RS232, VCP (virtual com port), Bluetooth, RealPort (network com port)) or to the network and returns the number of devices connected to the PC.
To open a specific COM port use *JETI_OpenCOMDevice* instead.

NOTE: Do not mix this function with calls to JETI_OpenCOMDevice, JETI_OpenTCPDevice, JETI_OpenFTDIDevice and JETI_OpenBTDevice!

Prototype

DWORD JETI_GetNumDevices (DWORD *dwNumDevices)

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

3.1.2 JETI_GetSerialDevice

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_GetNumDevices* – 1. Please note that the arrays for the serial numbers must provide space for 16 characters.

Prototype

DWORD JETI_GetSerialDevice (DWORD dwDeviceNum, char *cBoardSerialNr, char *cSpecSerialNr, char *cDeviceSerialNr)

Parameters

Input

Name	Type	Description	Call
dwDeviceNum	DWORD	index of the device for which the serial numbers are desired	By value
cBoardSerialNr	char *	address of a string that will contain the electronics serial number	By reference
cSpecSerialNr	char *	address of a string that will contain the spectrometer serial number	By reference
cDeviceSerialNr	char *	address of a string that will contain the device serial number	By reference

Return Value

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

3.1.3 JETI_OpenDevice

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

NOTE: Do not mix this function with calls to JETI_OpenCOMDevice, JETI_OpenTCPDevice, JETI_OpenFTDIDevice and JETI_OpenBTDevice!

Prototype

DWORD JETI_OpenDevice (DWORD dwDeviceNum, DWORD_PTR *dwDevice)

Parameters

Input

Name	Type	Description	Call
dwDeviceNum	DWORD	Device index. 0 for first device, 1 for second, etc.	By value
dwDevice	DWORD_PTR *	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

3.1.4 JETI_OpenCOMDevice

Opens a device (using a specific COM port number and baudrate) and returns a handle which will be used for subsequent accesses.
For searching automatically all JETI devices use *JETI_GetNumDevices* and *JETI_OpenDevice* instead.

NOTE: All JETI devices with FTDI USB-to-Serial converter normally will communicate directly with the FTDI driver instead of using the VCP (virtual com port). If the COM port number of such a device is set with these function the VCP driver will be used. To have the full stability advantage of the FTDI driver don't use this function.

NOTE: Do not mix this function with calls to JETI_GetNumDevices, JETI_OpenTCPDevice, JETI_OpenFTDIDevice, JETI_OpenBTDevice and JETI_OpenDevice!

Prototype

DWORD JETI_OpenCOMDevice (DWORD dwComPort, DWORD dwBaudrate, DWORD_PTR *dwDevice)

Parameters

Input

Name	Type	Description	Call
dwComPort	DWORD	COM port number from 1 to 255	By value
dwBaudrate	DWORD	Baudrate of the COM port (38400, 115200, 921600)	By value
dwDevice	DWORD_PTR *	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

3.1.5 JETI_OpenTCPDevice

Opens a JETI network device (using a specific IP address) and returns a handle which will be used for subsequent accesses.

NOTE: Do not mix this function with calls to JETI_GetNumDevices, JETI_OpenCOMDevice, JETI_OpenFTDIDevice, JETI_OpenBTDevice and JETI_OpenDevice!

Prototype

DWORD JETI_OpenTCPDevice (char * cIPAddr, DWORD_PTR *dwDevice)

Parameters

Input

Name	Type	Description	Call
cIPAddr	char *	A string with the IP address of the JETI device	By value
dwDevice	DWORD_PTR *	Pointer to a variable where the handle to the device will be stored	By reference

Return Value

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

3.1.6 JETI_OpenFTDIDevice

Opens a JETI USB device (using a specific USB serial number) and returns a handle which will be used for subsequent accesses.

NOTE: Do not mix this function with calls to JETI_GetNumDevices, JETI_OpenCOMDevice, JETI_OpenTCPDevice, JETI_OpenBTDevice and JETI_OpenDevice!

Prototype

DWORD JETI_OpenFTDIDevice (char * cUSBSerial, DWORD_PTR *dwDevice)

Parameters

Input

Name	Type	Description	Call
cUSBSerial	char *	A string with the USB serial number of the JETI device	By value
dwDevice	DWORD_PTR *	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

3.1.7 JETI_OpenBTDevice

Opens a JETI bluetooth device (using a specific bluetooth address) and returns a handle which will be used for subsequent accesses.

NOTE: Do not mix this function with calls to JETI_GetNumDevices, JETI_OpenCOMDevice, JETI_OpenFTDIDevice, JETI_OpenTCPDevice and JETI_OpenDevice!

Prototype

DWORD JETI_OpenBTDevice (unsigned long long btAddress, DWORD_PTR *dwDevice)

Parameters

Input

Name	Type	Description	Call
btAddress	unsigned long long	A bluetooth address of the JETI device	By value
dwDevice	DWORD_PTR *	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

3.1.8 JETI_CloseDevice

Closes an open device using the handle provided by *JETI_OpenDevice, JETI_OpenCOMDevice, JETI_OpenTCPDevice, JETI_OpenFTDIDevice* and *JETI_OpenBTDevice*.

Prototype

DWORD JETI_CloseDevice (DWORD_PTR dwDevice)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device to close as returned by *JETI_OpenDevice, JETI_OpenCOMDevice, JETI_OpenTCPDevice, JETI_OpenFTDIDevice* and *JETI_OpenBTDevice*	By value

Return Value

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

3.1.9 JETI_Reset

This function performs a software reset of the device firmware.

Prototype

DWORD JETI_Reset (DWORD_PTR dwDevice)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value

Return Value

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

3.1.10 JETI_HardReset

This function performs a hardware reset of the device. The effect of this function is the same as disconnecting then reconnecting the device from USB.
If the device is connected via bluetooth or RS232 then only the handle to the COM port will be closed and reopened.

Prototype

DWORD JETI_Reset (DWORD_PTR dwDevice)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value

Return Value

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

3.1.11 JETI_GetComPortHandle

Returns the handle of the COM port which can be used in subsequent calls to Windows functions like WriteFile() or ReadFile() to send special firmware commands directly to the device.

NOTE: This function will return an error if the device was opened using FTDI driver or direct LAN access (see JETI_OpenCOMDevice and JETI_OpenTCPDevice for detailed information).

Prototype

DWORD JETI_GetComPortHandle (DWORD_PTR dwDevice, HANDLE *hComPortHandle)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device to close as returned by *JETI_OpenDevice*	By value
hComPortHandle	HANDLE *	Pointer to a variable where the handle to the COM port will be stored	By reference

Return Value

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

3.2 Device Parameter

3.2.1 JETI_GetCoreDLLVersion

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

Prototype

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

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

3.2.2 JETI_GetFirmwareVersion

This function returns the current firmware version string of the JETI device.

Prototype

DWORD JETI_GetFirmwareVersion (DWORD_PTR dwDevice, char *cVersionString)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device to close as returned by *JETI_OpenDevice*	By value
cVersionString	char *	address of a char array of 256 characters to store the firmware version string	By reference

Return Value

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

3.2.3 JETI_GetDeviceType

This function returns the type of the currently connected JETI device.

NOTE: The returned device type number matches with the following devices:
0 – generic JETI device
1 – specbos device (xx01)
2 – specbos 1211
3 – spectraval 1501/1511

Prototype

DWORD JETI_GetDeviceType (DWORD_PTR dwDevice, BYTE *bDeviceType)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device to close as returned by *JETI_OpenDevice*	By value
bDeviceType	BYTE *	Pointer to a variable where the device type will be stored	By reference

Return Value

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

3.2.4 JETI_GetDeviceInfo

This function returns some information on how a JETI device is connected, using device number returned by *JETI_GetNumDevices*.

NOTE: The returned connection type number matches with the following:
0 – device not connected
1 – connected via serial COM port (either real or virtual)
2 – connected via USB (FTDI)
3 – connected via LAN
4 – connected via bluetooth

The returned device type number matches with the following devices:
0 – generic JETI device
1 – specbos device (xx01)
2 – specbos 1211
3 – spectraval 1501/1511

Prototype

```
DWORD JETI_GetDeviceInfo (DWORD dwDeviceNum, BYTE *bConnType, BYTE *bDeviceType,
                          char *cDeviceSerial, WORD *wComPortNr, DWORD *dwBaudrate,
                          char *cIPAddress, char *cUSBSerial, unsigned long long *btAddress)
```

Parameters

Input

Name	Type	Description	Call
dwDeviceNum	DWORD	Device index. 0 for first device, 1 for second, etc.	By value
bConnType	BYTE *	Pointer to a variable where the connection type will be stored	By reference
bDeviceType	BYTE *	Pointer to a variable where the device type will be stored	By reference
cDeviceSerial	char *	Array of 16 characters for the device serial number	By reference
wComPortNr	WORD *	Pointer to a variable where the serial COM port number will be stored if connection type is 1 or 2	By reference
dwBaudrate	DWORD *	Pointer to a variable where the baudrate will be stored if connection type is 1 or 2	By reference
cIPAddress	char *	Array of 16 characters for IP address if connection type is 3	By reference

Name	Type	Description	Call
cUSBSerial	char *	Array of 16 characters for the USB serial number if connection type is 2	By reference
btAddress	unsigned long long *	Pointer to a variable where the bluetooth address will be stored if connection type is 4	By reference

Return Value

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

3.2.5 JETI_GetPixel

Returns the pixel quantity of the used photodiode array.

Prototype

DWORD JETI_GetPixel (DWORD_PTR dwDevice, DWORD *dwPixel)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwPixel	DWORD *	Pointer to a variable where the pixel quantity value will be stored	By reference

Return Value

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

3.2.6 JETI_GetFit

Returns the wavelength fit parameters for the device. These values can be used to calculate the pixel-wavelength-correlation according to the following formula where p is the pixel number for which the wavelength is calculated:

$$\lambda(p) = fit[0] + fit[1] \times p + fit[2] \times p^2 + fit[3] \times p^3 + fit[4] \times p^4$$

Prototype

DWORD JETI_GetFit (DWORD_PTR dwDevice, FLOAT *fFit)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fFit	FLOAT *	Pointer to an array of up to 5 values where the fit parameters will be stored	By reference

Return Value

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

3.2.7 JETI_GetSDelay

Returns the scan delay (time difference between initiating a measurement and its real start) in [ms].

Prototype

DWORD JETI_GetSDelay (DWORD_PTR dwDevice, DWORD *dwSDelay)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwSDelay	DWORD *	Pointer to a variable where the scan delay will be stored	By reference

Return Value

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

3.2.8 JETI_SetSDelay

This function sets the scan delay (time difference between initiating a measurement and its real start) in [ms].

Prototype

DWORD JETI_SetSDelay (DWORD_PTR dwDevice, DWORD dwSDelay)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwSDelay	DWORD	the scan delay in [ms]	By value

Return Value

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

3.2.9 JETI_GetTint

Returns the default integration time of the device in ms.

Prototype

DWORD JETI_GetTint (DWORD_PTR dwDevice, float *fTint)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fTint	float *	Pointer to a variable where the integration time will be stored	By reference

Return Value

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

3.2.10 JETI_GetADCRes

Returns the digital resolution of the ADC (analog-digital-converter) in bit.

Prototype

DWORD JETI_GetADCRes (DWORD_PTR dwDevice, BYTE *bADCRes)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bADCRes	BYTE *	Pointer to a variable where the ADC-resolution will be stored	By reference

Return Value

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

3.2.11 JETI_GetBorder

Returns the low and high border used for the adaption of integration time. The borders are percent of full-scale.

Prototype

DWORD JETI_GetBorder (DWORD_PTR dwDevice, BYTE *bBorderMin, BYTE *bBorderMax)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bBorderMin	BYTE *	Pointer to a variable where the lower border will be stored	By reference
bBorderMax	BYTE *	Pointer to a variable where the upper border will be stored	By reference

Return Value

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

3.2.12 JETI_GetDistance

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

Prototype

DWORD JETI_GetDistance (DWORD_PTR dwDevice, DWORD *dwDistance)

Parameters

Input

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

Return Value

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

3.2.13 JETI_SetDistance

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

Prototype

DWORD JETI_SetDistance (DWORD_PTR dwDevice, DWORD dwDistance)

Parameters

Input

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

Return Value

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

3.2.14 JETI_GetOptTrigg

(only specbos 1211)

Returns the availability of an optical trigger on the device. If an optical trigger is available JETI_GetFlickerFreq can be used to determine the flicker frequency of pulsed light sources / pulsed monitor back-lights.
0 – no optical trigger
1 – optical trigger available

Prototype

DWORD JETI_GetOptTrigg (DWORD_PTR dwDevice, BOOL *boOptTrigg)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boOptTrigg	BOOL *	Pointer to a variable where the optical trigger availability will be stored	By reference

Return Value

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

3.2.15 JETI_SetLaserIntensity

This function sets the intensity and modulation of the internal laser spot of specbos 1201/1211 and spectraval 11501/511.

Prototype

DWORD JETI_SetLaserIntensity (DWORD_PTR dwDevice, DWORD dwIntensity, DWORD dwModulation)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwIntensity	DWORD	the intensity value in [‰]	By value
dwModulation	DWORD	modulation mode: 0 – laser off 1 – PWM 7Hz 2 – PWM 28 Hz 3 – PWM 255 Hz **(ignored if device is spectraval)**	By value

Return Value

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

3.2.16 JETI_SetTrigger

This function sets the trigger mode.
It is possible to trigger a measurement externally. Please read the firmware documentation for further details.

Prototype

DWORD JETI_SetTrigger (DWORD_PTR dwDevice, DWORD dwTriggerMode)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwTriggerMode	DWORD	trigger mode: 0 – trigger disabled 1 – enabled 2 – enquiry mode	By value

Return Value

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

3.2.17 JETI_GetFlickerFreq

(only specbos 1211 and spectraval 1501/1511)

This function can determine a frequency in Hz of pulsed light sources / pulsed monitor back-lights.

Prototype

DWORD JETI_GetFlickerFreq (DWORD_PTR dwDevice, float *fFlickerFreq, DWORD *dwWarning)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fFlickerFreq	float *	pointer to a variable where the flicker frequency in [Hz] will be stored	By reference
dwWarning	DWORD *	pointer to a variable where a warning code will be stored: 0 – no warning 11 – no modulation 12 – fuzzy modulation	By reference

Return Value

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

3.2.18 JETI_GetBatteryStat

(only specbos 1211 and spectraval 1501/1511)

This function returns information about the battery capacity of a battery driven device.

Prototype

DWORD JETI_GetBatteryCapacity (DWORD_PTR dwDevice, float *fBattVolt, BYTE *bBattPercent, BYTE *bIsBattLoading)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fBattVolt	float *	pointer to a variable where the battery voltage in [V] will be stored	By reference
bBattPercent	BYTE *	pointer to a variable where the battery charge in [%] will be stored	By reference
bIsBattLoading	BYTE *	pointer to a vairable where the battery charging flag will be stored	By reference

Return Value

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

3.3 Peripheral Control

3.3.1 JETI_GetLaserStat

This function gets the status of the internal pilot laser of a specbos device:
- 0: laser is off
- 1: laser is on

Prototype

DWORD JETI_GetLaserStat (DWORD_PTR dwDevice, BOOL *boLaserStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boLaserStat	BOOL *	pointer to a variable where the pilot laser status will be stored TRUE (1) – laser is on FALSE (0) – laser is off	By reference

Return Value

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

3.3.2 JETI_SetLaserStat

This function switches on and off the internal pilot laser of a specbos device.

Prototype

DWORD JETI_SetLaserStat (DWORD_PTR dwDevice, BOOL boLaserStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boLaserStat	BOOL	the status to set TRUE (1) – laser on FALSE (0) – laser off	By value

Return Value

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

3.3.3 JETI_GetShutterStat

This function gets the status of the internal shutter and lamp respectively:
- 1: shutter is open *(specbos 1201/1211, spectraval 1501/1511)*
  lamp is on *(specbos 2001/2101/4001)*
- 0: shutter is closed *(specbos 1201/1211, spectraval 1501/1511)*
  lamp is off *(specbos 2001/2101/4001)*

Prototype

DWORD JETI_GetShutterStat (DWORD_PTR dwDevice, BOOL *boShutterStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boShutterStat	BOOL *	pointer to a variable where the shutter/lamp status will be stored TRUE (1) – shutter is open / lamp is on FALSE (0) – shutter is closed / lamp is off	By reference

Return Value

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

3.3.4 JETI_SetShutterStat

This function switch the internal lamp on/off and open and close the internal shutter respectively:
- 1: shutter is open / lamp is on
- 0: shutter is closed / lamp is off

Prototype

DWORD JETI_SetShutterStat (DWORD_PTR dwDevice, BOOL *boLaserStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boShutterStat	BOOL	the status to set TRUE (1) – shutter is open / lamp is on FALSE (0) – shutter is closed / lamp is off	By value

Return Value

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

3.3.5 JETI_GetCalibRange

Returns the calibrated wavelength range of the currently used calibration file.

Prototype

DWORD JETI_GetCalibRange (DWORD_PTR dwDevice, DWORD *dwBegin, DWORD *dwEnd, DWORD *dwStep)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBegin	DWORD *	Pointer to a variable where the start wavelength value in nm will be stored	By reference
dwEnd	DWORD *	Pointer to a variable where the end wavelength value in nm will be stored	By reference
dwStep	DWORD *	Pointer to a variable where the wavelength step value in nm will be stored	By reference

Return Value

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

3.3.6 JETI_SetCalib

Set the calibration file number bCalibNr (1-based index) to use for radiometric measurements.
If bCalibNr is set to zero (standard) the calibration file will be determined automatically in accordance with the attached measuring head (see JETI_GetMeasHead for further informations).

Prototype

DWORD JETI_SetCalib (DWORD_PTR dwDevice, BYTE bCalibNr)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bCalibNr	BYTE	The calibration file number to use for radiometric measurements (0 for automatic)	By value

Return Value

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

3.3.7 JETI_GetCalib

Returns the calibration file number bCalibNr (1-based index) which will be used for radiometric measurements.
If bCalibNr returns zero (standard) the calibration file will be determined automatically in accordance with the attached measuring head (see JETI_GetMeasHead for further informations).

Prototype

DWORD JETI_GetCalib (DWORD_PTR dwDevice, BYTE *bCalibNr)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bCalibNr	BYTE *	Pointer to a variable where the calibration file number will be stored	By reference

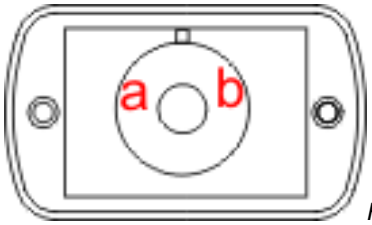
Return Value

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

3.3.8 JETI_GetMeasHead

(only specbos 1201/1211)

This function gets the measuring head configuration (signal of hall sensors). The following settings are possible:



Front view of SCB 1201/1211 with positions of Hall sensors.

sensor signal 'ab'	*BYTE value*	*calibration file number*
00	0	1
01	1	2
10	2	3
11	3	4

NOTE: The measuring head configuration can be used to automatically select an internal calibration file for radiometric measurements. If JETI_SetCalib is set to zero (default) the corresponding calibration file will be used,
e.g. if bMeasHead returns 0 calibration file 1 will be used,
if bMeasHead returns 1 calibration file 2 will be used, etc.

Prototype

DWORD JETI_GetMeasHead (DWORD_PTR dwDevice, BYTE *bMeasHead)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bMeasHead	BYTE *	pointer to a variable where the measuring head status will be stored	By reference

Return Value

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

3.3.9 JETI_ReadUserData64

Reads user data starting from block 'dwStart' to block 'dwEnd' from connected device. Up to 64KByte (depends on the devices capabilities) can be stored.

Prototype

DWORD JETI_ReadUserData64 (DWORD_PTR dwDevice, BYTE *bData, DWORD dwStart, DWORD dwEnd)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenDevice*	By value
bData	BYTE*	Buffer of up to 65536 Byte (64K) of user-defined data	By reference
dwStart	DWORD	zero-based index of 1K starting block (0..63)	By value
dwEnd	DWORD	zero-based index of 1K end block (0..63)	By value

Return Value

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

3.3.10 JETI_WriteUserData64

Writes user data block number 'dwBlock' to connected device. Up to 64KByte (depends on the devices capabilities) can be stored.

Prototype

DWORD JETI_WriteUserData64 (DWORD_PTR dwDevice, BYTE *bData, DWORD dwBlock)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD	Handle to a device as returned by *JETI_OpenDevice*	By value
bData	BYTE*	Buffer of 1024 Byte (1K) of user-defined data	By reference
dwBlock	DWORD	zero-based index of 1K starting block (0..63)	By value

Return Value

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

3.3.11 JETI_MeasureADC1

(only for VersaPIC electronics)

Returns the ADC count value of the 10 bit analogue input ADC1 of VersaPIC S255 Add-On-Connector.

Prototype

DWORD JETI_MeasureADC1 (DWORD_PTR dwDevice, WORD *wADC1)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
wADC1	WORD*	Pointer to a variable for the ADC value	By reference

Return Value

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

3.3.12 JETI_MeasureADC2

(only for VersaPIC electronics)

Returns the ADC count value of the 10 bit analogue input ADC2 of VersaPIC S255 Add-On-Connector.

Prototype

DWORD JETI_MeasureADC2 (DWORD_PTR dwDevice, WORD *wADC2)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
wADC2	WORD*	Pointer to a variable for the ADC value	By reference

Return Value

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

3.3.13 JETI_GetAux1Stat

This function gets the status of the auxiliary 1:
- 1: aux1 is on
- 0: aux1 is off

Prototype

DWORD JETI_GetAux1Stat (DWORD_PTR dwDevice, BOOL *boAuxStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAuxStat	BOOL *	pointer to a variable where the aux1 status will be stored TRUE (1) – aux1 is on FALSE (0) – aux1 is off	By reference

Return Value

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

3.3.14 JETI_SetAux1Stat

This function switch the auxiliary 1 on and off:
- 1: aux1 is on
- 0: aux1 is off

Prototype

DWORD JETI_SetAux1Stat (DWORD_PTR dwDevice, BOOL *boAuxStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAuxStat	BOOL	the status to set TRUE (1) – aux1 is on FALSE (0) – aux1 is off	By value

Return Value

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

3.3.15 JETI_GetAux2Stat

This function gets the status of the auxiliary 2:
- 1: aux2 is on
- 0: aux2 is off

Prototype

DWORD JETI_GetAux2Stat (DWORD_PTR dwDevice, BOOL *boAuxStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAuxStat	BOOL *	pointer to a variable where the aux2 status will be stored TRUE (1) – aux2 is on FALSE (0) – aux2 is off	By reference

Return Value

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

3.3.16 JETI_SetAux2Stat

This function switch the auxiliary 2 on and off:
- 1: aux2 is on
- 0: aux2 is off

Prototype

DWORD JETI_SetAux2Stat (DWORD_PTR dwDevice, BOOL *boAuxStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAuxStat	BOOL	the status to set TRUE (1) – aux2 is on FALSE (0) – aux2 is off	By value

Return Value

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

3.3.17 JETI_AuxOut1

This function switch the TTL output auxout1 of VersaPIC S255 Add-On-Connector on and off:
- 1: auxout1 is on
- 0: auxout1 is off

Prototype

DWORD JETI_AuxOut1 (DWORD_PTR dwDevice, BOOL boAux1)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux1	BOOL	the status to set TRUE (1) – auxou1 is on FALSE (0) – auxout1 is off	By value

Return Value

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

3.3.18 JETI_AuxOut1Stat

This function returns the status of the TTL output auxout1 of VersaPIC S255 Add-On-Connector:
- 1: auxout1 is on
- 0: auxout1 is off

Prototype

DWORD JETI_AuxOut1Stat (DWORD_PTR dwDevice, BOOL *boAux1Stat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux1Stat	BOOL *	pointer to a variable where the auxout1 status will be stored TRUE (1) – auxout1 is on FALSE (0) – auxout1 is off	By reference

Return Value

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

3.3.19 JETI_AuxOut2

This function switch the LV TTL output auxout2 of VersaPIC S255 Add-On-Connector on and off:
- 1: auxout2 is on
- 0: auxout2 is off

Prototype

DWORD JETI_AuxOut2 (DWORD_PTR dwDevice, BOOL boAux2)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux2	BOOL	the status to set TRUE (1) – auxout2 is on FALSE (0) – auxout2 is off	By value

Return Value

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

3.3.20 JETI_AuxOut2Stat

This function returns the status of the LV TTL output auxout2 of VersaPIC S255 Add-On-Connector:
- 1: auxout2 is on
- 0: auxout2 is off

Prototype

DWORD JETI_AuxOut2Stat (DWORD_PTR dwDevice, BOOL *boAux2Stat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux2Stat	BOOL *	pointer to a variable where the auxout2 status will be stored TRUE (1) – auxout2 is on FALSE (0) – auxout2 is off	By reference

Return Value

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

3.3.21 JETI_AuxOut3

This function switch the LV TTL output auxout3 of VersaPIC S255 Add-On-Connector on and off:
- 1: auxout3 is on
- 0: auxout3 is off

Prototype

DWORD JETI_AuxOut3 (DWORD_PTR dwDevice, BOOL boAux3)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux3	BOOL	the status to set TRUE (1) – auxout3 is on FALSE (0) – auxout3 is off	By value

Return Value

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

3.3.22 JETI_AuxOut3Stat

This function returns the status of the LV TTL output auxout3 of VersaPIC S255 Add-On-Connector:
- 1: auxout3 is on
- 0: auxout3 is off

Prototype

DWORD JETI_AuxOut3Stat (DWORD_PTR dwDevice, BOOL *boAux3Stat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux3Stat	BOOL *	pointer to a variable where the auxout3 status will be stored TRUE (1) – auxout3 is on FALSE (0) – auxout3 is off	By reference

Return Value

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

3.3.23 JETI_AuxOut4

This function switch the LV TTL output auxout4 of VersaPIC S255 Add-On-Connector on and off:
- 1: auxout4 is on
- 0: auxout4 is off

Prototype

DWORD JETI_AuxOut4 (DWORD_PTR dwDevice, BOOL boAux4)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux4	BOOL	the status to set TRUE (1) – auxout4 is on FALSE (0) – auxout4 is off	By value

Return Value

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

3.3.24 JETI_AuxOut4Stat

This function returns the status of the LV TTL output auxout4 of VersaPIC S255 Add-On-Connector:
- 1: auxout4 is on
- 0: auxout4 is off

Prototype

DWORD JETI_AuxOut4Stat (DWORD_PTR dwDevice, BOOL *boAux4Stat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux4Stat	BOOL *	pointer to a variable where the auxout4 status will be stored TRUE (1) – auxout4 is on FALSE (0) – auxout4 is off	By reference

Return Value

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

3.3.25 JETI_AuxOut5

This function switch the LV TTL output auxout5 of VersaPIC S255 Add-On-Connector on and off:
- 1: auxout5 is on
- 0: auxout5 is off

Prototype

DWORD JETI_AuxOut5 (DWORD_PTR dwDevice, BOOL boAux5)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux5	BOOL	the status to set TRUE (1) – auxout5 is on FALSE (0) – auxout5 is off	By value

Return Value

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

3.3.26 JETI_AuxOut5Stat

This function returns the status of the LV TTL output auxout5 of VersaPIC S255 Add-On-Connector:
- 1: auxout5 is on
- 0: auxout5 is off

Prototype

DWORD JETI_AuxOut5Stat (DWORD_PTR dwDevice, BOOL *boAux5Stat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAux5Stat	BOOL *	pointer to a variable where the auxout5 status will be stored TRUE (1) – auxout5 is on FALSE (0) – auxout5 is off	By reference

Return Value

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

3.3.27 JETI_AuxIn1Stat

This function returns the status of the TTL input auxin1 of VersaPIC S255 Add-On-Connector:
- 1: auxin1 is on
- 0: auxin1 is off

Prototype

DWORD JETI_AuxIn1Stat (DWORD_PTR dwDevice, BOOL *boAuxIn1Stat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAuxIn1Stat	BOOL *	pointer to a variable where the auxin1 status will be stored TRUE (1) – auxin1 is on FALSE (0) – auxin1 is off	By reference

Return Value

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

3.3.28 JETI_AuxIn2Stat

This function returns the status of the TTL input auxin2 of VersaPIC S255 Add-On-Connector:
- 1: auxin2 is on
- 0: auxin2 is off

Prototype

DWORD JETI_AuxIn2Stat (DWORD_PTR dwDevice, BOOL *boAuxIn2Stat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boAuxIn2Stat	BOOL *	pointer to a variable where the auxin2 status will be stored TRUE (1) – auxin2 is on FALSE (0) – auxin2 is off	By reference

Return Value

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

3.3.29 JETI_GetDIOIn

(only for RU60 electronics)

This function returns the status of the digital input pins of a RU60 electronics as a byte value.
The bits 0...2 represents the pins DI0...DI2.

Prototype

DWORD JETI_GetDIOIn (DWORD_PTR dwDevice, BYTE *bDIOIn)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bDIOIn	BYTE *	pointer to a variable for the digital input status	By reference

Return Value

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

3.3.30 JETI_GetDIOOut

(only for RU60 electronics)

This function returns the status of the digital output pins of a RU60 electronics as a byte value.
The bits 0...6 represents the pins DO0...DO6.

Prototype

DWORD JETI_GetDIOOut (DWORD_PTR dwDevice, BYTE *bDIOOut)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bDIOOut	BYTE *	pointer to a variable for the digital output status	By reference

Return Value

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

3.3.31 JETI_SetDIOOut

(only for RU60 electronics)

This function sets the status of the digital output pins of a RU60 electronics.
The bits 0...6 of the byte value given represents the pins DO0...DO6.

Prototype

DWORD JETI_SetDIOOut (DWORD_PTR dwDevice, BYTE bDIOOut)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bDIOOut	BYTE	the byte value to set the digital output status	By value

Return Value

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

3.3.32 JETI_SetDIOOutPin

(only for RU60 electronics)

This function sets the status of a single digital output pin of a RU60 electronics.
To set a single output status set bPinNr to the corresponding digital output (0 – DO0, 1 – DO1,...) and set boDIOOut to 0 or 1.

Prototype

DWORD JETI_SetDIOOutPin (DWORD_PTR dwDevice, BYTE bPinNr, BOOL boDIOOut)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bPinNr	BYTE	the pin number of the digital output to set	By value
boDIOOut	BOOL	the status to set (0 / 1)	By value

Return Value

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

3.3.33 JETI_GetTemperature

(only for RU60 electronics)

This function gets the temperature of a connected sensor of a RU60 electronics.

Prototype

DWORD JETI_GetTemperature (DWORD_PTR dwDevice, FLOAT *fTemperature)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fTemperature	FLOAT *	pointer to a variable for the temperature value	By reference

Return Value

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

3.4 Measurement

3.4.1 JETI_InitMeasure

Starts a pre configured measurement.

NOTE: The function will return ***immediately.*** Before any other DLL-function call the function *JETI_MeasureStatusCore* 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.

Prototype

DWORD JETI_InitMeasure (DWORD_PTR dwDevice)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value

Return Value

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

3.4.2 JETI_MeasureStatusCore

Returns the status of a measurement started with *JETI_InitMeasure*. 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_MeasureStatusCore* reports that the measurement has finished.

Prototype

DWORD JETI_MeasureStatusCore (DWORD_PTR dwDevice, BOOL *boStatus)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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

3.4.3 JETI_Break

This function cancels an initiated measurement.

Prototype

DWORD JETI_Break (DWORD_PTR dwDevice)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value

Return Value

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

3.4.4 JETI_StartAdaption

Starts an adaption of integration time.

NOTE: The function will return ***immediately.*** Before any other DLL-function call the function *JETI_CheckAdaptionStat* 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.

Prototype

DWORD JETI_StartAdaption (DWORD_PTR dwDevice, BOOL boReference)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boReference	BOOL	0 – only light measurement 1 – additionally reference measurement	By value

Return Value

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

3.4.5 JETI_CheckAdaptionStat

Returns the status of the automatic adaption of integration time started with *JETI_StartAdaption*. 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 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_CheckAdaptionStat* reports that the measurement has finished.

Prototype

DWORD JETI_CheckAdaptionStat (DWORD_PTR dwDevice, FLOAT *fTint, WORD *wAverage, BOOL *boStatus)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fTint	FLOAT *	Pointer to a variable where the actual integration time in [ms] will be stored	By reference
wAverage	WORD *	Pointer to a variable where the actual averages will be stored	By reference
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	JETI_SUCCESS
	0x…	see Appendix A for error codes

3.4.6 JETI_GetLevel

Returns the exposure level of a previously performed radiometric or reference measurement.

Prototype

DWORD JETI_GetLevel (DWORD_PTR dwDevice, DWORD *dwLevelCounts, DWORD *dwLevelPercent)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwLevelCounts	DWORD *	Pointer to a variable where the exposure level (ADC-counts) will be stored	By reference
dwLevelPercent	DWORD *	Pointer to a variable where the exposure level (Percent) will be stored	By reference

Return Value

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

3.4.7 JETI_GetDarkmodeConf

(only specbos 1201/1211)

This function gets the dark measurement mode of radiometric and reference measurement:
- 1: perform a dark scan after each measurement
- 0: no dark scan after each measurement, use "dark values" for the dark current compensation

To use the darkmode 0 a dark current measurement must be performed by JETI_MeasCompDark.

Prototype

DWORD JETI_GetDarkmodeConf (DWORD_PTR dwDevice, BYTE *bDarkmode)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bDarkmode	BYTE *	pointer to a variable where the dark mode will be stored	By reference

Return Value

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

3.4.8 JETI_SetDarkmodeConf

(only specbos 1201/1211)

This function sets the dark measurement mode of radiometric and reference measurement:
- 1: perform a dark scan after each measurement
- 0: no dark scan after each measurement, use "dark values" for the dark current compensation

To use the darkmode 0 a dark current measurement must be performed by JETI_MeasCompDark.

Prototype

DWORD JETI_SetDarkmodeConf (DWORD_PTR dwDevice, BYTE bDarkmode)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bDarkmode	BYTE	the dark measurement mode to set	By value

Return Value

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

3.4.9 JETI_MeasCompDark

(only specbos 1201/1211)

Initiates a dark measurement for further dark compensation.

NOTE: The function will return ***immediately.*** Before any other DLL-function call the function *JETI_MeasureStatusCore* must be used to check if the measurement has finished. **Please note that this measurement could take up to 5 seconds.**

Prototype

DWORD JETI_MeasDarkComp (DWORD_PTR dwDevice)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value

Return Value

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

3.4.10 JETI_GetCutoffStat

This function gets the cutoff mode of radiometric and reference measurement:
- 1: negative spectral values will be truncated by a special compensation method
- 0: no negative values will be truncated

Prototype

DWORD JETI_GetCutoffStat (DWORD_PTR dwDevice, BOOL *boCutoffStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boCutoffStat	BOOL *	pointer to a variable where the cutoff status will be stored	By reference

Return Value

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

3.4.11 JETI_SetCutoffStat

This function sets the cutoff mode of radiometric and reference measurement:
- 1: negative spectral values will be truncated by a special compensation method
- 0: no negative values will be truncated

Prototype

DWORD JETI_GetCutoffStat (DWORD_PTR dwDevice, BOOL *boCutoffStat)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
boCutoffStat	BOOL	the cutoff status to set	By reference

Return Value

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

3.4.12 JETI_GetExposureConf

(not for spectraval 1501/1511)

This function gets the handling of the integration time:
- 0: use previous integration time
- 1: always adapt integration time
- 2: use configured integration time

Prototype

DWORD JETI_GetExposureConf (DWORD_PTR dwDevice, BYTE *bExpmode)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bExpmode	BYTE *	pointer to a variable where the exposure mode will be stored	By reference

Return Value

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

3.4.13 JETI_SetExposureConf

(not for spectraval 1501/1511)

This function sets the handling of the integration time
- 0: use previous integration time
- 1: always adapt integration time
- 2: use configured integration time

Prototype

DWORD JETI_SetExposureConf (DWORD_PTR dwDevice, BYTE bExpmode)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bExpmode	BYTE	the exposure mode to set	By value

Return Value

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

3.4.14 JETI_GetFunctionConf

This function gets the measurement function:
- 1: exposure spectrum
- 2: dark spectrum
- 3: reference spectrum
- 4: transmission spectrum (not for spectraval 1501/1511)
- 6: radiometric spectrum (not for spectraval 1501/1511)

Prototype

DWORD JETI_GetFunctionConf (DWORD_PTR dwDevice, BYTE *bPrevFunc, BYTE *bConfFunc)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bPrevFunc	BYTE *	pointer to a variable where the last used measurement function will be stored	By reference
bConfFunc	BYTE *	pointer to a variable where the configured function for next measurement will be stored	By refernce

Return Value

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

3.4.15 JETI_SetFunctionConf

This function sets measurement function:

- 1: exposure spectrum
- 2: dark spectrum
- 3: reference spectrum
- 4: transmission spectrum (not for spectraval 1501/1511)
- 6: radiometric spectrum (not for spectraval 1501/1511)

Prototype

DWORD JETI_SetFunctionConf (DWORD_PTR dwDevice, BYTE bFunction)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bDarkmode	BYTE	the measurement function to set	By value

Return Value

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

3.4.16 JETI_GetTintConf

This function gets the integration time configuration.

Prototype

DWORD JETI_GetTintConf (DWORD_PTR dwDevice, float *fPrevTint, float *fConfTint)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fPrevTint	float *	pointer to a variable where the last used integration time will be stored	By reference
fConfTint	float *	pointer to a variable where the configured integration time for the next measurement will be stored	By reference

Return Value

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

3.4.17 JETI_SetTintConf

This function sets the integration time for the next measurement.
The maximum value for integration time is 60000.0 ms.
The minimum value can be obtained by the function *JETI_GetMinTintConf*.

NOTE: Digits after the decimal point will be ignored on specbos devices.

Prototype

DWORD JETI_SetTintConf (DWORD_PTR dwDevice, float fTint)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fTint	float	the integration time to set (minimum integration time… 60000.0 ms)	By value

Return Value

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

3.4.18 JETI_GetMinTintConf

This function gets the minimum integration time which can be used with the currently connected instrument.

Prototype

DWORD JETI_GetMinTintConf (DWORD_PTR dwDevice, float *fMinTint)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fMinTint	float *	pointer to a variable where the minimum integration time will be stored	By reference

Return Value

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

3.4.19 JETI_GetMaxTintConf

This function gets the maximum integration time which will be used for adaption.

Prototype

DWORD JETI_GetMaxTintConf (DWORD_PTR dwDevice, float *fMaxTint)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fMaxTint	float *	pointer to a variable where the maximum integration time will be stored	By reference

Return Value

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

3.4.20 JETI_SetMaxTintConf

This function sets the maximum integration time which will be used for the next auto-adapted measurement. Allowed values are between 1000 and 60000 ms for specbos devices and between 400 and 6000 ms for spectraval devices.

Prototype

DWORD JETI_SetMaxTintConf (DWORD_PTR dwDevice, float fMaxTint)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fMaxTint	float	the maximum integration time to set (400…60000 ms)	By value

Return Value

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

3.4.21 JETI_GetMaxAverConf

(only spectraval 1501/1511)

This function gets the maximum averages which will be used for adaption.

Prototype

DWORD JETI_GetMaxTintConf (DWORD_PTR dwDevice, WORD *wMaxAver)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
wMaxAver	WORD *	pointer to a variable where the maximum averages will be stored	By reference

Return Value

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

3.4.22 JETI_GetAverConf

This function gets the count of measurement scans for average calculation.

Prototype

DWORD JETI_GetAverConf (DWORD_PTR dwDevice, WORD *wPrevAver, WORD *wConfAver)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
wPrevAver	DWORD *	pointer to a variable where the last used average value will be stored	By reference
wConfAver	DWORD *	pointer to a variable where the configured average value for the next measurement will be stored	By reference

Return Value

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

3.4.23 JETI_SetAverConf

This function sets the count of measurement scans for average calculation.

Prototype

DWORD JETI_SetAverConf (DWORD_PTR dwDevice, WORD wAver)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
wAver	WORD	the count of measurement scans to set	By value

Return Value

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

3.4.24 JETI_GetAdaptConf

(only specbos devices)

This function gets the adaptation mode:
- 0: no adaptation if under or over exposure
- 1: new adaptation only if over exposure
- 2: new adaptation if under or over exposure

Prototype

DWORD JETI_GetAdaptConf (DWORD_PTR dwDevice, BYTE *bAdaptmode)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bAdaptmode	BYTE *	pointer to a variable where the adaptation mode setting will be stored	By reference

Return Value

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

3.4.25 JETI_SetAdaptConf

(only specbos devices)

This function sets the adaptation mode.
- 0: no adaptation if under or over exposure
- 1: new adaptation only if over exposure
- 2: new adaptation if under or over exposure

Prototype

DWORD JETI_SetAdaptConf (DWORD_PTR dwDevice, BYTE bAdaptmode)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bAdaptmode	BYTE	the adaptation mode to set	By value

Return Value

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

3.4.26 JETI_GetWranConf

This function gets the wavelength range.

Prototype

DWORD JETI_GetWranConf (DWORD_PTR dwDevice, DWORD *dwBeg, DWORD *dwEnd, DWORD *dwStep)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD *	pointer to a variable where the start wavelength will be stored in [nm]	By reference
dwEnd	DWORD *	pointer to a variable where the end wavelength will be stored in [nm]	By reference
dwStep	DWORD *	pointer to a variable where the step-width will be stored in [nm]	By reference

Return Value

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

3.4.27 JETI_SetWranConf

This function sets the wavelength range.

Prototype

DWORD JETI_SetWranConf (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, DWORD dwStep)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	the start wavelength to set in [nm]	By value
dwEnd	DWORD	the end wavelength to set in [nm]	By value
dwStep	DWORD	the step-width to set in [nm]	By value

Return Value

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

3.4.28 JETI_GetPDARowConf

This function gets the PDA (photo-diode array) row setting.
On some PDAs (e.g. Hamamatsu S9840, S7030, S10420) it is possible to read out single rows of the detector. The possible settings depends on the detector. If dwPDARow returns 0 (zero) the whole detector array is used.

Prototype

DWORD JETI_GetPDARowConf (DWORD_PTR dwDevice, DWORD *dwPDARow, DWORD *dwRowNumber)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwPDARow	DWORD *	pointer to a variable where the starting PDA row will be stored	By reference
dwRowNumber	DWORD *	pointer to a variable where the number of PDA rows will be stored	By reference

Return Value

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

3.4.29 JETI_SetPDARowConf

This function sets the PDA (photo-diode array) row setting.
On some PDAs (e.g. Hamamatsu S9840, S7030, S10420) it is possible to read out single rows of the detector. The possible settings depends on the detector. If dwPDARow is set to 0 (zero) the whole detector array is used.

Prototype

DWORD JETI_SetPDARowConf (DWORD_PTR dwDevice, DWORD dwPDARow, DWORD dwRowNumber)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwPDARow	DWORD	the starting PDA row to set	By value
dwRowNumber	DWORD	the number of PDA rows to set	By value

Return Value

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

3.4.30 JETI_GetSyncMode

(only for specbos 1211 and spectraval 1501/1511)

This function gets the sync mode.
The function JETI_SetSyncFreq must be used to set the sync frequency.

SPECBOS 1211
If sync mode is set to 1 the firmware will use number of cycles instead of milliseconds for the integration time.
If sync mode is set to 0 the integration time will be in milliseconds.

SPECTRAVAL 1501/1511
If sync mode is set to 1 the firmware will sync the adapted integration time according to the sync frequency.
If sync mode is set to 0 the adapted integration time is independent from the sync frequency.
Fixed integration times will always be in milliseconds.

Prototype

DWORD JETI_GetSyncMode (DWORD_PTR dwDevice, BYTE *bSyncMode)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bSyncMode	BYTE *	pointer to a variable where the sync mode setting will be stored	By reference

Return Value

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

3.4.31 JETI_SetSyncMode

(only for specbos 1211 and spectraval 1501/1511)

This function sets the sync mode.
The function JETI_SetSyncFreq must be used to set the sync frequency.

SPECBOS 1211
If sync mode is set to 1 the firmware will use number of cycles instead of milliseconds for the integration time.
If sync mode is set to 0 the integration time will be in milliseconds.

SPECTRAVAL 1501/1511
If sync mode is set to 1 the firmware will sync the adapted integration time according to the sync frequency.
If sync mode is set to 0 the adapted integration time is independent from the sync frequency.
Fixed integration times will always be in milliseconds.

Prototype

DWORD JETI_SetCycModeConf (DWORD_PTR dwDevice, BYTE bSyncMode)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
bSyncMode	BYTE	the sync mode to set	By value

Return Value

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

3.4.32 JETI_GetSyncFreq

(only for specbos 1211 and spectraval 1501/1511)

This function gets the sync frequency in [Hz].

Prototype

DWORD JETI_GetSyncFreq (DWORD_PTR dwDevice, float *fSyncFreq)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fSyncFreq	float *	pointer to a variable where the sync frequency will be stored	By reference

Return Value

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

3.4.33 JETI_SetSyncFreq

(only for specbos 1211 and spectraval 1501/1511)

This function sets the sync frequency in [Hz].

Prototype

DWORD JETI_SetSyncFreq (DWORD_PTR dwDevice, float fSyncFreq)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fSyncFreq	float	the sync frequency to set	By value

Return Value

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

3.4.34 JETI_SetDefault

This function sets all measurement parameters to default values.

Prototype

DWORD JETI_SetDefault (DWORD_PTR dwDevice)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value

Return Value

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

3.5 Fetch Results

3.5.1 JETI_FetchDark

This function returns the previously measured dark spectrum in counts per pixel.

NOTE: The array dwDark must provide space for at least as many values as the count of pixel of the photodiode-array. See function *JETI_GetPixel* for further information

Prototype

DWORD JETI_FetchDark (DWORD_PTR dwDevice, INT32 *iDark)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
iDark	INT32 *	pointer to an array where the dark spectrum will be stored	By reference

Return Value

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

3.5.2 JETI_FetchLight

This function returns the previously measured light spectrum in counts per pixel.

NOTE: The array dwLight must provide space for at least as many values as the count of pixel of the photodiode-array. See function *JETI_GetPixel* for further information

Prototype

DWORD JETI_FetchLight (DWORD_PTR dwDevice, INT32 *iLight)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
iLight	INT32 *	pointer to an array where the light spectrum will be stored	By reference

Return Value

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

3.5.3 JETI_FetchRefer

This function returns the previously measured reference spectrum in counts per pixel.

NOTE: The array dwRefer must provide space for at least as many values as the count of pixel of the photodiode-array. See function *JETI_GetPixel* for further information

Prototype

DWORD JETI_FetchRefer (DWORD_PTR dwDevice, INT32 *iRefer)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
iRefer	INT32 *	pointer to an array where the reference spectrum will be stored	By reference

Return Value

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

3.5.4 JETI_FetchTransRefl

This function returns the previously measured transmission / reflection spectrum in counts per pixel.

NOTE: The array dwTransRefl must provide space for at least as many values as the count of pixel of the photodiode-array. See function *JETI_GetPixel* for further information

Prototype

DWORD JETI_FetchTransRefl (DWORD_PTR dwDevice, INT32 *iTransRefl)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
iTransRefl	INT32 *	pointer to an array where the transmission / reflection spectrum will be stored	By reference

Return Value

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

3.5.5 JETI_FetchSprad

This function returns the previously measured radiometric spectrum. The unit of the spectrum depends on the measuring head and the corresponding calibration file (see JETI_GetMeasHead for further informations).

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	$W / sr \times nm$

NOTE: The array fSprad must provide space for the values accordingly to the wavelength range setting. For example if the wavelength range is set to 380…780 nm in 5 nm steps 81 values will be received.

$$\frac{endwavelength - startwavelength}{wavesteps} + 1$$

Prototype

DWORD JETI_FetchSprad (DWORD_PTR dwDevice, FLOAT *fSprad)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fSprad	FLOAT *	pointer to an array where the radiometric spectrum will be stored	By reference

Return Value

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

3.5.6 JETI_FetchRadio

This function returns the previously measured radiometric value. The unit of the value depends on the measuring head and the corresponding calibration file (see JETI_GetMeasHead for further informations).

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

Prototype

DWORD JETI_FetchRadio (DWORD_PTR dwDevice, FLOAT *fRadio)

Parameters

Input

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

Return Value

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

3.5.7 JETI_FetchPhoto

This function returns the previously measured photometric value. The unit of the value depends on the measuring head and the corresponding calibration file (see JETI_GetMeasHead for further informations).

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

Prototype

DWORD JETI_FetchPhoto (DWORD_PTR dwDevice, FLOAT *fPhoto)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fPhoto	FLOAT *	pointer to a value where the photometric value will be stored	By reference

Return Value

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

3.5.8 JETI_FetchChromxy

This function returns the previously measured CIE-1931 chromaticity coordinates x and y. The calculation is based on a 2° observer, and the wavelength range is 380…780 nm.

Prototype

DWORD JETI_FetchChromxy (DWORD_PTR dwDevice, FLOAT *fChromx, FLOAT *fChromy)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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

3.5.9 JETI_FetchChromuv

This function returns the previously measured CIE-1976 chromaticity coordinates u' and v'. The calculation is based on a 2° observer, and the wavelength range is 380…780 nm.

Prototype

DWORD JETI_FetchChromxy (DWORD_PTR dwDevice, FLOAT *fChromx, FLOAT *fChromy)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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

3.5.10 JETI_FetchDWLPE

This function returns the previously measured dominant wavelength (DWL) and color purity (PE). The calculation is based on a 2° observer, and the wavelength range is 380…780 nm.
The unit for the dominant wavelength is [nm].
The unit for the color purity is [%] (percent).

Prototype

DWORD JETI_FetchDWLPE (DWORD_PTR dwDevice, FLOAT *fDWL, FLOAT *fPE)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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

3.5.11 JETI_FetchCCT

This function returns the previously measured correlated color temperature.

Prototype

DWORD JETI_FetchCCT (DWORD_PTR dwDevice, float *fCCT)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fCCT	float *	pointer to a variable where the correlated color temperature will be stored	By reference

Return Value

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

3.5.12 JETI_FetchDuv

This function returns the previously measured Δuv of the measured CCT.

Prototype

DWORD JETI_FetchDuv (DWORD_PTR dwDevice, FLOAT *fDuv)

Parameters

Input

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

Return Value

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

3.5.13 JETI_FetchXYZ

This function returns the previously measured tristimulus XYZ.

Prototype

DWORD JETI_FetchXYZ (DWORD_PTR dwDevice, FLOAT *fX, FLOAT *fY, FLOAT *fZ)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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

3.5.14 JETI_FetchCRI

This function returns the previously calculated color rendering indices according to CIE 13.3-1995 publication and the color rendering index of the JIS color sample.

The function returns an array of 17 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 No. 1…8.
Value number three (index 2) to value number 17 (index 16) contains the special color rendering indices.

Prototype

DWORD JETI_FetchCRI (DWORD_PTR dwDevice, float *fCRI)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fCRI	float *	pointer to an array where the CRI-values will be stored **(the array must contain space for at least 17 values)**	By reference

Return Value

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

3.6 Calculations

3.6.1 JETI_CalcLintDark

This function calculates the linear interpolated ADC-counts per wavelength of a previously performed dark measurement.

NOTE: The array fDark must provide space for the values accordingly to the wavelength range setting. For example if the wavelength range is set to 380…780 nm in 5 nm steps 81 values will be received.

$$\frac{dwEnd - dwBeg}{fStep} + 1$$

Prototype

DWORD JETI_CalcLintDark (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT fStep, FLOAT *fDark)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	the start wavelength for calculation in [nm]	By value
dwEnd	DWORD	the end wavelength for calculation in [nm]	By value
fStep	FLOAT	the step-width for calculation in [nm]	By value
fDark	FLOAT *	pointer to an array where the linear interpolated dark values will be stored	By reference

Return Value

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

3.6.2 JETI_CalcSplinDark

This function calculates the cubic spline interpolated ADC-counts per wavelength of a previously performed dark measurement.

NOTE: The array fDark must provide space for the values accordingly to the wavelength range setting. For example if the wavelength range is set to 380…780 nm in 5 nm steps 81 values will be received.

$$\frac{dwEnd - dwBeg}{fStep} + 1$$

Prototype

DWORD JETI_CalcSplinDark (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT fStep, FLOAT *fDark)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	the start wavelength for calculation in [nm]	By value
dwEnd	DWORD	the end wavelength for calculation in [nm]	By value
fStep	FLOAT	the step-width for calculation in [nm]	By value
fDark	FLOAT *	pointer to an array where the cubic spline interpolated dark values will be stored	By reference

Return Value

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

3.6.3 JETI_CalcLintLight

This function calculates the linear interpolated ADC-counts per wavelength of a previously performed light measurement.

NOTE: The array fLight must provide space for the values accordingly to the wavelength range setting. For example if the wavelength range is set to 380…780 nm in 5 nm steps 81 values will be received.

$$\frac{dwEnd - dwBeg}{fStep} + 1$$

Prototype

DWORD JETI_CalcLintLight (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT fStep, FLOAT *fLight)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	the start wavelength for calculation in [nm]	By value
dwEnd	DWORD	the end wavelength for calculation in [nm]	By value
fStep	FLOAT	the step-width for calculation in [nm]	By value
fLight	FLOAT *	pointer to an array where the linear interpolated light values will be stored	By reference

Return Value

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

3.6.4 JETI_CalcSplinLight

This function calculates the cubic spline interpolated ADC-counts per wavelength of a previously performed light measurement.

NOTE: The array fLight must provide space for the values accordingly to the wavelength range setting. For example if the wavelength range is set to 380…780 nm in 5 nm steps 81 values will be received.

$$\frac{dwEnd - dwBeg}{fStep} + 1$$

Prototype

DWORD JETI_CalcSplinLight (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT fStep, FLOAT *fLight)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	the start wavelength for calculation in [nm]	By value
dwEnd	DWORD	the end wavelength for calculation in [nm]	By value
fStep	FLOAT	the step-width for calculation in [nm]	By value
fLight	FLOAT *	pointer to an array where the cubic spline interpolated light values will be stored	By reference

Return Value

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

3.6.5 JETI_CalcLintRefer

This function calculates the linear interpolated ADC-counts per wavelength of a previously performed reference measurement.

NOTE: The array fRefer must provide space for the values accordingly to the wavelength range setting. For example if the wavelength range is set to 380…780 nm in 5 nm steps 81 values will be received.

$$\frac{dwEnd - dwBeg}{fStep} + 1$$

Prototype

DWORD JETI_CalcLintRefer (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT fStep, FLOAT *fRefer)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	the start wavelength for calculation in [nm]	By value
dwEnd	DWORD	the end wavelength for calculation in [nm]	By value
fStep	FLOAT	the step-width for calculation in [nm]	By value
fRefer	FLOAT *	pointer to an array where the linear interpolated reference values will be stored	By reference

Return Value

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

3.6.6 JETI_CalcSplinRefer

This function calculates the cubic spline interpolated ADC-counts per wavelength of a previously performed reference measurement.

NOTE: The array fRefer must provide space for the values accordingly to the wavelength range setting. For example if the wavelength range is set to 380…780 nm in 5 nm steps 81 values will be received.

$$\frac{dwEnd - dwBeg}{fStep} + 1$$

Prototype

DWORD JETI_CalcSplinRefer (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT fStep, FLOAT *fRefer)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	the start wavelength for calculation in [nm]	By value
dwEnd	DWORD	the end wavelength for calculation in [nm]	By value
fStep	FLOAT	the step-width for calculation in [nm]	By value
fRefer	FLOAT *	pointer to an array where the cubic spline interpolated reference values will be stored	By reference

Return Value

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

3.6.7 JETI_CalcLintTransRefl

This function calculates the linear interpolated transmission / reflection values per wavelength of a previously performed transmission / reflection measurement.

NOTE: The array fTransRefl must provide space for the values accordingly to the wavelength range setting. For example if the wavelength range is set to 380…780 nm in 5 nm steps 81 values will be received.

$$\frac{dwEnd - dwBeg}{fStep} + 1$$

Prototype

DWORD JETI_CalcLintTransRefl (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT fStep,
FLOAT *fTransRefl)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	the start wavelength for calculation in [nm]	By value
dwEnd	DWORD	the end wavelength for calculation in [nm]	By value
fStep	FLOAT	the step-width for calculation in [nm]	By value
fTransRefl	FLOAT *	pointer to an array where the linear interpolated transmission / reflection values will be stored	By reference

Return Value

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

3.6.8 JETI_CalcSplinTransRefl

This function calculates the cubic spline interpolated transmission / reflection values per wavelength of a previously performed transmission / reflection measurement.

NOTE: The array fDark must provide space for the values accordingly to the wavelength range setting. For example if the wavelength range is set to 380…780 nm in 5 nm steps 81 values will be received.

$$\frac{dwEnd - dwBeg}{fStep} + 1$$

Prototype

DWORD JETI_CalcSplinTransRefl (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT fStep,
FLOAT *fTransRefl)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	the start wavelength for calculation in [nm]	By value
dwEnd	DWORD	the end wavelength for calculation in [nm]	By value
fStep	FLOAT	the step-width for calculation in [nm]	By value
fTransRefl	FLOAT *	pointer to an array where the cubic spline interpolated transmission / reflection values will be stored	By reference

Return Value

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

3.6.9 JETI_CalcRadio

This function returns the calculated radiometric value. The unit of the value depends on the measuring head and the corresponding calibration file (see JETI_GetMeasHead for further informations).

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

Prototype

DWORD JETI_CalcRadio (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT *fRadio)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
dwBeg	DWORD	start wavelength for calculation	By value
dwEnd	DWORD	end wavelength for caluclation	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

3.6.10 JETI_CalcPhoto

This function returns the calculated photometric value. The unit of the value depends on the measuring head and the corresponding calibration file (see JETI_GetMeasHead for further informations).

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

Prototype

DWORD JETI_CalcPhoto (DWORD_PTR dwDevice, FLOAT *fPhoto)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fPhoto	FLOAT *	pointer to a value where the photometric value will be stored	By reference

Return Value

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

3.6.11 JETI_CalcChromxy

This function returns the calculated CIE-1931 chromaticity coordinates x and y. The calculation is based on a **2° observer**, and the wavelength range is 380…780 nm.

Prototype

DWORD JETI_CalcChromxy (DWORD_PTR dwDevice, FLOAT *fChromx, FLOAT *fChromy)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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

3.6.12 JETI_CalcChromxy10

This function returns the calculated CIE-1931 chromaticity coordinates x and y. The calculation is based on a **10° observer**, and the wavelength range is 380…780 nm.

Prototype

DWORD JETI_CalcChromxy10 (DWORD_PTR dwDevice, FLOAT *fChromx10, FLOAT *fChromy10)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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	JETI_SUCCESS
	0x…	see Appendix A for error codes

3.6.13 JETI_CalcChromuv

This function returns the calculated CIE-1976 chromaticity coordinates u' and v'. The calculation is based on a 2° observer, and the wavelength range is 380…780 nm.

Prototype

DWORD JETI_CalcChromuv (DWORD_PTR dwDevice, FLOAT *fChromu, FLOAT *fChromv)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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

3.6.14 JETI_CalcDWLPE

This function returns the calculated dominant wavelength (DWL) and color purity (PE). The calculation is based on a 2° observer, and the wavelength range is 380…780 nm.
The unit for the dominant wavelength is [nm].
The unit for the color purity is [%] (percent).

Prototype

DWORD JETI_CalcDWLPE (DWORD_PTR dwDevice, FLOAT *fDWL, FLOAT *fPE)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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

3.6.15 JETI_CalcCCT

This function returns the calculated correlated color temperature.

Prototype

DWORD JETI_CalcCCT (DWORD_PTR dwDevice, float *fCCT)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fCCT	float *	pointer to a variable where the correlated color temperature will be stored	By reference

Return Value

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

3.6.16 JETI_CalcDuv

This function returns the calculated Δuv for the correlated color temperature.

Prototype

DWORD JETI_CalcDuv (DWORD_PTR dwDevice, FLOAT *fDuv)

Parameters

Input

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

Return Value

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

3.6.17 JETI_CalcXYZ

This function returns the calculated tristimulus XYZ.

Prototype

DWORD JETI_CalcXYZ (DWORD_PTR dwDevice, FLOAT *fX, FLOAT *fY, FLOAT *fZ)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	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

3.6.18 JETI_CalcCRI

This function returns the calculated color rendering indices according to CIE 13.3-1995 publication and the color rendering index of the JIS color sample.
The color temperature of the reference source is specified by fCCT. To use the same CCT as calculated, set fCCT to zero (0),
The function returns an array of 17 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 No. 1…8.
Value number three (index 2) to value number 17 (index 16) contains the special color rendering indices.

Prototype

DWORD JETI_CalcCRI (DWORD_PTR dwDevice, float fCCT, float *fCRI)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fCCT	float	color temperature of reference light source	By value
fCRI	float *	pointer to an array where the CRI-values will be stored **(the array must contain space for at least 17 values)**	By reference

Return Value

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

3.6.19 JETI_CalcAllValue

This function calculates all radiometric, photometric and colorimetric values.

Prototype

DWORD JETI_CalcAllValue (DWORD_PTR dwDevice, DWORD dwBeg, DWORD dwEnd, FLOAT *fRadio, FLOAT *fPhoto, FLOAT *fChromx, FLOAT *fChromy, FLOAT *fChromu, FLOAT *fChromv, FLOAT *fDWL, FLOAT *fPE)

Parameters

Input

Name	Type	Description	Call
dwDevice	DWORD_PTR	Handle to a device as returned by *JETI_OpenDevice*	By value
fRadio	FLOAT *	pointer to a variable where the radiometric value will be stored	By reference
fPhoto	FLOAT *	pointer to a value where the photometric value will be stored	By reference
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
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
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 0x…	JETI_SUCCESS see Appendix A for error codes

4 Examples

To help starting development the SDK includes several examples for different programming languages.

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_OVEREXPOSURE	measurement failed due to overexposure
0x22	JETI_MEASURE_FAIL	measurement failed due to other reasons
0x23	JETI_ADAPTION_FAIL	adaption failed
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 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_GetNumDevices and JETI_OpenDevice) instead of using the VCP (virtual com port) driver (JETI_OpenCOMDevice).
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_CloseDevice 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_GetNumDevices and JETI_OpenDevice.

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_CloseDevice 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) 2016 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.

September 23, 2016