Spvis Spectrometer Library Manual

Spvis 光谱仪开发动态库手册

1. DLL Information 动态库信息

DLL Name 动态库名称	Spvis Spectrometer develop dynamic library	Software Version 版本序号	Lc Spvis v2.0 for XS

2. Library Description 动态库说明

Provides an interface from the PC to the dynamic library, which is developed based on the .NET platform. This interface allows the application to configure the spectrometer and receive and send data from the spectrometer.

A dynamic library uses a pair of open and close functions (LC_Init() and LC_Done()) that must be called using the program. As long as the open function is not called, all other functions will return an error code. The open function (LC_Init()) attempts to open a communication interface for all linked devices. The close function (LC_Done()) closes the communication interface and releases all internal data.

The interface between the application program and the dynamic library can be divided into four functional groups:

- **Internal data read functions**

  -These functions are used to read device configuration data from the internal storage of the device.

- **Blocking control functions**

  -These functions control the blocking behavior of sending requests to the device and waiting for a response. They can either block until a reply is received or send a timeout notification before returning control to the application program.

- **Non-blocking data read functions**

  - These functions send requests to the device and then return control to the application program. After receiving a reply from the device or experiencing a timeout, they send a notification to the application program.

- **Data send functions**

  - These functions are used to send device configuration data to the device. After the application program is initialized, it should select the spectrometer to be used.

3. Name-space and Class 名称空间及类库

Library File 库文件名称	LcSpvis_XS.dll	
Framework 开发平台	.NET Framework 4.6.1	
Name-space 名称空间	LcSpvis	-
Class 类库	LC_Func	光谱仪功能开发

4. Activating a USB Device 激活 USB 设备

To activate a spectrometer device, the following steps must be performed:

1) First, call LC_Init() to initialize the library and search for USB spectrometer devices. This method will return the number of available spectrometer devices.

2) You can then call LC_GetList() to retrieve the operation index of the connected spectrometers and their corresponding serial numbers. The status of the device can be determined by using the spectrometer serial number.

3) Use LC_Activate() to select the spectrometer to be activated. During activation, the license file for the selected spectrometer needs to be loaded, which contains information such as spectrometer-specific corrections and usage permissions. Only after successfully loading the license file can the device execute related functional methods.

4) In the subsequent chapters, we will discuss how to configure relevant parameters to achieve fast and reliable testing of related data.

5. Exported Functions 方法说明

5.1. LC_Init

Function 方法	**int LC_Init** (void)
Group 方法组	Blocking control function
Description 描述	The communication interface for initializing the spectrometer and internal data structures. Once the LC_Init() function is called, the software internally instantiates the required number of spectrometers. There is no need to instantiate them multiple times.
Parameters 参数	None
Return 返回值	**Success:** the return value will be the number of connected or discovered devices. If no devices are connected, the return value will be **0.** **Failure:** When encountering a failure, the return value will be ERR_UNKNOWN=**-99**.

5.2. LC_DoneAll

Function 方法	**int LC_DoneAll** (void)
Group 方法组	Blocking control function
Description 描述	This function will close the communication interface with all spectrometers and release the internal storage.
Parameters 参数	None
Return 返回值	**Success:**: ERR_SUCCESS = 0 **Failure**: ERR_UNKNOWN = -99

5.3. LC_Done

Function 方法	**int LC_Done** (int l_Index,)
Group 方法组	Blocking control function
Description 描述	To close the communication with a specific spectrometer, identified by its operation index, and release the internal storage.
Parameters 参数	**l_Index**: An integer representing the selected spectrometer index
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT= -19 ERR_UNKNOWN = -99

5.4. LC_Activate

Function 方法	**int LC_Activate** (int l_Index, string l_File,)
Group 方法组	Blocking control function
Description 描述	Activate the selected spectrometer for communication purposes.
Parameters 参数	**l_Index**: An integer representing the selected spectrometer index **l_File**: licenseFilePath: A string representing the full path of the license file (.spt file) for the selected spectrometer

Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT= -19 ERR_FILE_NO_EXIST = -24 ERR_INVALID_DEVICE_TYPE = -7 ERR_INVALID_DEVICE_ID = -4 ERR_UNKNOWN = -99

5.5. LC_GetList

Function 方法	**int LC_GetList** (int I_Index, ref string I_SerialNumber,)
Group 方法组	Blocking control function
Description 描述	This function will return the device information associated with the spectrometer connected to the specified port.
Parameters 参数	**I_Index**: An integer representing the selected spectrometer index. **I_SerialNumber**: A string representing the serial number of the selected spectrometer.
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT= -19 ERR_UNKNOWN = -99

5.6. LC_GetParameters

Function 方法	**int LC_GetParameters** (int l_Index, int l_mpiParametersNo, ref string l_Parameters,)
Group 方法组	Blocking control function
Description 描述	This function will return a data structure specific to the spectrometer.
Parameters 参数	**l_Index**: An integer representing the selected spectrometer index. **l_mpiParametersNo**: An integer representing the mode selection for retrieving spectrometer device parameters. **0**: Get the spectrometer model. **1**: Get the total number of pixels of the spectrometer. **2**: Get the starting wavelength of the spectrometer. **3**: Get the ending wavelength of the spectrometer. **l_Parameters**: A string representing the parameter value for the selected mode.
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_INVALID_PARAMETER = -1 ERR_UNKNOWN = -99

5.7. LC_AutoDark

Function 方法	**int LC_AutoDark** (int l_Index, double l_Integration,)
Group 方法组	Blocking control function
Description 描述	To retrieve the auto dark spectrum data for the selected spectrometer
Parameters 参数	**l_Index**: An integer representing the selected spectrometer index. **l_Integration**: A double precision floating-point number representing the upper limit of integration time for the dark spectrum.
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_INVALID_INT_TIME = -11 ERR_UNKNOWN = -99

5.8. LC_OnceDark

Function 方法	**int LC_OnceDark** (int I_Index, double I_Integration, int I_Averaging,)
Group 方法组	Blocking control function
Description 描述	This function will perform a single measurement to capture the dark spectrum data from the spectrometer. The dark spectrum represents the background noise or dark signal of the spectrometer without any light source.
Parameters 参数	**I_Index:** An integer representing the selected spectrometer index. **I_Integration**: A double precision floating-point number representing the upper limit of integration time for the dark spectrum. **I_Averaging**: An integer representing the number of times the dark spectrum data is averaged.
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_UNKNOWN = -99

5.9. LC_AutoIntegration

Function 方法	**int LC_AutoIntegration** (int I_Index, double I_Saturation, ref double I_Integration, ref int I_Averaging,)
Group 方法组	Blocking control function
Description 描述	To obtain the automatic integration time and automatic averaging count of a spectrometer under different light conditions.
Parameters 参数	**I_Index:** An integer representing the selected spectrometer index. **I_Saturation**: A double precision floating-point number representing the target saturation level for automatic integration. **I_Integration**: A double precision floating-point number representing the

	automatic integration time obtained. **l_Averaging**: An integer representing the automatic averaging count obtained.
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_INVALID_SATURATION = -13 ERR_INVALID_AUTO_INT= -22 ERR_UNKNOWN = -99

5.10. LC_SetIntegration

Function 方法	**int LC_SetIntegration** (int l_Index, double l_Integration, int l_Averaging,)
Group 方法组	Blocking control function
Description 描述	To set the integration time and averaging count for the selected spectrometer.
Parameters 参数	**l_Index**: An integer representing the selected spectrometer index. **l_Integration**: A double precision floating-point number representing the integration time for the selected spectrometer. **l_Averaging**: An integer representing the averaging count for the selected spectrometer.
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_INVALID_INT_TIME = -11 ERR_INVALID_AVERAGING= -12 ERR_UNKNOWN = -99

5.11. LC_GetSaturation

Function 方法	**int LC_GetSaturation** (int l_Index, double l_Integration, int l_Averaging, ref double l_Saturation,)
Group 方法组	Blocking control function
Description 描述	To obtain the saturation level of a spectrometer's response at a specific integration time.
Parameters 参数	**l_Index**: An integer representing the selected spectrometer index. **l_Integration**: A double precision floating-point number representing the integration time for the selected spectrometer. **l_Averaging**: An integer representing the averaging count for the selected spectrometer. **l_Saturation:** A double precision floating-point number representing saturation level of the spectrometer
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_UNKNOWN = -99

5.12. LC_SetAutoMaxIntegration

Function 方法	**Int LC_SetAutoMaxIntegration** (int l_Index, ref double l_MaxIntegration, ref int l_MaxAveraging,)
Group 方法组	Blocking control function 阻塞功能方法
Description 描述	To set the maximum integration time and maximum number of averages for the selected spectrometer.
Parameters 参数	**l_Index**: An integer representing the selected spectrometer index. **l_MaxIntegration**: A double precision floating-point number representing the Maximum integration time for the selected spectrometer. **l_MaxAveraging**: An integer representing the Maximum averaging count for the selected spectrometer.
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_UNKNOWN = -99

5.13. LC_GetSpectrum

Function 方法	**int LC_GetSpectrum** (int l_Index, int l_DarkMode, double l_Integration, int l_Averaging, ref double[] l_Spectrum,)
Group 方法组	Blocking control function
Description 描述	To obtain the response data of the selected spectrometer in a specified mode.
Parameters 参数	**l_Index:** Integer, the selected spectrometer's index **l_Mode**: Integer, mode selection 0 - Obtain raw response data from the spectrometer 1 - Obtain raw response data from the spectrometer with automatic dark spectrum subtraction 2 - Obtain raw response data from the spectrometer with single dark spectrum subtraction **l_Integration**: Double precision floating-point, integration time **l_Averaging**: Integer, number of averages **l_Spectrum**: One-dimensional array of double precision floating-point, the

	obtained spectral response values.
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_INVALID_PARAMETER = -1 ERR_UNKNOWN = -99

5.14. LC_Almp

Function 方法	**int LC_Almp** (int I_Index, int I_DarkMode, double I_Integration, int I_Averaging, double[] I_AlmpSp, double[] I_AlmpWave,)
Group 方法组	Blocking control function
Description 描述	To calibrate the spectrometer system, you can use the spectral radiance data from a standard lamp.
Parameters 参数	**I_Index**: Integer, the selected spectrometer's index **I_Mode**: Integer, mode selection 1 - Obtain raw response data from the spectrometer with automatic dark spectrum subtraction 2 - Obtain raw response data from the spectrometer with single dark spectrum subtraction **I_Integration**: Double precision floating-point, integration time **I_Averaging**: Integer, number of averages **I_AlmpSp**: One-dimensional array of double precision floating-point, spectral radiance values of the standard lamp **I_AlmpWave**: One-dimensional array of double precision floating-point, wavelength values of the standard lamp's spectral radiance
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_INVALID_ARRAY = -14 ERR_UNKNOWN = -99

5.15. LC_ReadFbr

Function 方法	**int LC_ReadFbr** (int l_Index, string l_File,)
Group 方法组	Blocking control function
Description 描述	To import the calibration file for the spectrometer's standard lamp
Parameters 参数	**l_Index**: Integer, the selected spectrometer's index **l_File**: String, the complete path of the calibration file (.txt) for the selected spectrometer's standard lamp
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_FILE_NO_EXIST = -24 ERR_INVALID_DEVICE_TYPE = -7 ERR_INVALID_DEVICE_ID = -4 ERR_UNKNOWN = -99

5.16. LC_SaveFbr

Function 方法	**int LC_SaveFbr** (int l_Index, int l_UsageMode, string l_Path,)
Group 方法组	Blocking control function
Description 描述	To save the calibration file for the spectrometer's standard lamp.
Parameters 参数	**l_Index:** Integer, the selected spectrometer's index **l_UsageMode:** Integer, the measurement usage mode of the spectrometer **l_Path**: String, the save path for the calibration file (.txt) of the spectrometer's standard lamp (automatically set filename as: Sp_spectrometer_serial_number.txt)
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_FILE_NO_EXIST = -24 ERR_INVALID_PARAMETER = -1 ERR_UNKNOWN = -99

5.17. LC_SetZoomFactor

Function 方法	**int LC_SetZoomFactor** (int l_Index, double l_Factor,)
Group 方法组	Blocking control function
Description 描述	To set the overall scaling factor for a spectrum.
Parameters 参数	**l_Index:** Integer, the selected spectrometer's index **l_Factor:** Double precision floating-point number, the overall scaling factor for the spectrum
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_INVALID_OPTICAL_PARAMETER = -10 ERR_UNKNOWN = -99

5.18. LC_Measure

Function 方法	**int LC_Measure** (int l_Index, double l_Integration, int l_Averaging, int l_DarkMode, bool l_Aux, Int l_Smooth)
Group 方法组	Blocking control function
Description 描述	To perform a spectral measurement with the spectrometer.

Parameters 参数	**l_Index**: Integer, the selected spectrometer's index **l_Integration**: Double precision floating-point number, the integration time **l_Averaging**: Integer, the number of measurements to average **l_DarkMode**: Integer, the selection of dark spectrum mode to use: **1** - Obtain the spectrometer's original response data after subtracting the automatically acquired dark spectrum **2** - Obtain the spectrometer's original response data after subtracting a single acquired dark spectrum **l_Aux**: Boolean, whether to use auxiliary lamp compensation factors for correction **l_Smooth**: Integer, the number of pixels for rolling smoothing of the spectral response values
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_UNKNOWN = -99

5.19. LC_MeasureDate

Function 方法	**int LC_MeasureDate** (int l_Index, int l_mpiTestDataItem, ref double l_Data, ref double[] l_DateArray,)
Group 方法组	Blocking control function
Description 描述	To perform a spectral measurement
Parameters 参数	**l_Index:** Integer, the selected spectrometer's index **l_mpiTestDataItem**: Integer, mode selection: 0 - Spectral total energy 1 - Luminous flux (different modes represent different parameters) 2 - CIE 1931 tristimulus value - X 3 - CIE 1931 tristimulus value - Y 4 - CIE 1931 tristimulus value - Z 5 - CIE 1931 chromaticity coordinate - Cx 6 - CIE 1931 chromaticity coordinate - Cy 7 - CIE 1960 chromaticity coordinate - u 8 - CIE 1960 chromaticity coordinate - v 9 - CIE 1976 chromaticity coordinate - u' 10 - CIE 1976 chromaticity coordinate - v' 11 - Correlated color temperature (in Kelvin)

	12 - Color difference level between Duv and blackbody radiation color temperature 13 - Peak wavelength (in nm) 14 - Full width at half maximum (in nm) 15 - Energy at peak wavelength 16 - Pixel value at peak wavelength 17 - Wavelength at pixel value peak (in nm) 18 - Energy at pixel value peak 19 - Existence of dominant wavelength 20 - Dominant wavelength/complementary dominant wavelength (in nm) 21 - Purity 31 - Color rendering index Ra 32~46 - Color rendering index R1~R15 801 - Spectral wavelength 802 - Spectral response value 901 - Integration time for this test (in ms) 902 - Number of averages for this test 903 - Spectral response saturation for this test 904 - Time taken for this test (in ms) 905 - Spectrometer mode used for this test **l_Data:** Double precision floating-point number, the value returned for the selected mode **l_DateArray**: One-dimensional array of double precision floating-point numbers, the values returned for the selected mode (valid for 801/802)
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_INVALID_ACTIVATE = -2 ERR_UNKNOWN = -99

5.20. CheckCASError

Function 方法	**int CheckCASError** (int I_Index, out string I_ErrorInformation,)
Group 方法组	Blocking control function
Description 描述	To retrieve error information from a spectrometer.
Parameters 参数	**I_Index**: Integer, the selected spectrometer's index **I_ErrorInformation**: String, the obtained error content information from the current spectrometer
Return 返回值	**Success**: ERR_SUCCESS = 0 **Failure**: ERR_INDEX_EXCEED_LIMIT = -19 ERR_UNKNOWN = -99

6. Data Elements 数据元素

Here are several data types commonly used in libraries and required by application programming interfaces (APIs):

Type/ 类型	Format 格式	Value/Range 值/范围	Description 描述
StdSpectralData （测量后得到的数据）	Class		
	{		
	double d_IntegrationTime	0.06-10000 ms	Integration time for this test
	int d_Averaging	1-1000	Number of averages for this test
	double d_Saturation	0%-100%	Spectral response saturation for this test
	string d_Date	-	Time taken for this test
	double d_CostTime	ms	Time cost for this test
	double d_TotalPower	-	Spectral total energy
	double d_Intensity	-	Luminous flux
	double d_UsageMode	-	Spectrometer mode used for this test
	double d_X	-	CIE 1931 tristimulus value – X
	double d_Y	-	CIE 1931 tristimulus value – Y
	double d_Z	-	CIE 1931 tristimulus value - Z
	double d_Cx	-	CIE 1931 chromaticity coordinate - Cx
	double d_Cy	-	CIE 1931 chromaticity coordinate - Cy
	double d_U	-	CIE 1960 chromaticity coordinate - u
	double d_V	-	CIE 1960 chromaticity coordinate - v
	double d_Uc	-	CIE 1976 chromaticity coordinate - u'
	double d_Vc	-	CIE 1976 chromaticity coordinate - v'
	double d_Dominant	380nm-780nm	Dominant wavelength
	double d_Pure	0%-100%	Purity
	double d_HasDominant	Y/N	Existence of dominant wavelength
	double d_PeakCount	-	Energy at peak wavelength
	double d_PeakHalf	nm	Full width at half maximum
	double d_PeakWave	nm	Peak wavelength
	int d_PeakPix	-	Pixel value at peak wavelength
	double d_PeakPixWave	nm	Wavelength at pixel value peak
	double d_PeakPixCount	-	Energy at pixel value peak
	double d_CCT	1000K-10000K	Correlated color temperature (CCT)
	double d_Ds	-	Duv relative to color temperature
	double d_Ra	-	Color rendering index Ra
	double[] d_Wavelengths	-	Spectral wavelength
	double[] d_Spectrums	-	Spectral response value
	double[] d_CRI	-	Color rendering index R1~R15
	}		

7. Return Value Constants 返回值常量

The following table outlines possible integer return codes:

Return code 返回码	Value 值	Description 描述
ERR_SUCCESS	0	operation is successful
ERR_INVALID_PARAMETER	-1	Invalid input parameter value used
ERR_INVALID_ACTIVATE	-2	Device not properly activated
ERR_INVALID_DEVICE_ID	-4	Serial number in the file does not match the device serial number
ERR_INVALID_DEVICE_TYPE	-7	Device type in the file does not match
ERR_INVALID_OPTICAL_PARAMETER	-10	Invalid optical data parameter used
ERR_INVALID_INT_TIME	-11	Integration time setting exceeds the limit
ERR_INVALID_AVERAGING	-12	Number of averages setting exceeds the limit
ERR_INVALID_SATURATION	-13	Automatic integration saturation rate setting exceeds the limit
ERR_INVALID_ARRAY	-14	Spectrum or wavelength array error
ERR_INDEX_EXCEED_LIMIT	-19	Data range exceeds the limit
ERR_INVALID_AUTO_INT	-22	Exceeded time limit for automatic integration
ERR_FILE_NO_EXIST	-24	The path used does not exist
ERR_UNKNOWN	-99	Program code error

8. Examples for Windows

The example source code can be found in the "examples" directory. The provided 32-bit sample program (including header files and link libraries) offers a programming environment for the following:

-Microsoft C# 2017 (Managed code for .NET)

In the DLL directory, you can find both x64 and x86 versions of the drivers. Users can choose the appropriate dynamic library file based on their needs.

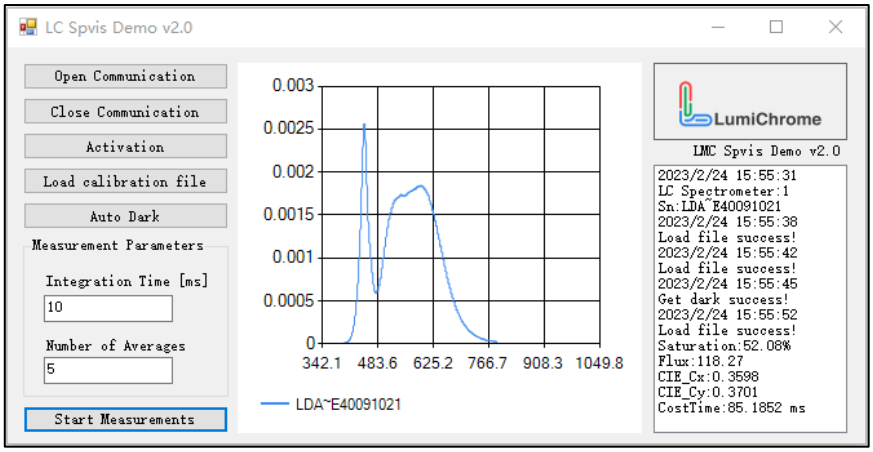
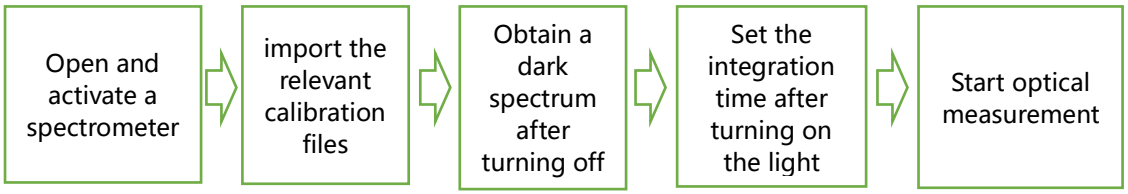


图 8.1 Demo software interface

8.1. General operation process 通常操作流程

The general operation process of a spectrometer is as follows:



8.2. Initialization and activation of the spectrometer 初始化及激活光谱仪

After launching the full-featured demo program, the main window will be displayed. By clicking the "Open Communication" button, the LC_Init() function is called. If successful, the LC_GetList() function is called to collect the serial numbers of the connected spectrometers. By clicking the "Close Communication" button, the LC_DoneAll() function is called. If executed successfully, the communication with the spectrometer will be closed, and internal storage will be released.

By clicking the "Activation" button, the LC_Activate() function is called to select the corresponding activation license file (.spt file) for the spectrometer. If the activation of the spectrometer is successful.

8.3. Import calibration files of the spectrometer 光谱仪相关标定文件导入

To load the calibration file, which includes the "Std Calibration" standard lamp calibration file, the LC_ReadFbr() function is called. By clicking the "Load Calibration File" button, you can select the appropriate calibration file (.txt file) for the corresponding spectrometer.

The "Integration Time [ms]" refers to the integration time for the spectrometer's measurements, which represents the exposure time when capturing the spectrum. A longer

exposure time allows the sensor to receive more energy, while a shorter exposure time results in less energy being received by the sensor. The device's maximum integration time is 10 seconds, and the minimum integration time is 0.06 milliseconds.

By adjusting the integration time, you can ensure that the sensor's spectral response remains within a reasonable range, avoiding overexposure (when the original response exceeds 65535). Overexposure can lead to spectral line errors, resulting in inaccurate calculation of related optical parameters. Typically, to maintain a linear response range, it is recommended to aim for a saturation rate of around 80%, which is approximately 65535 × 60% = 39,321.

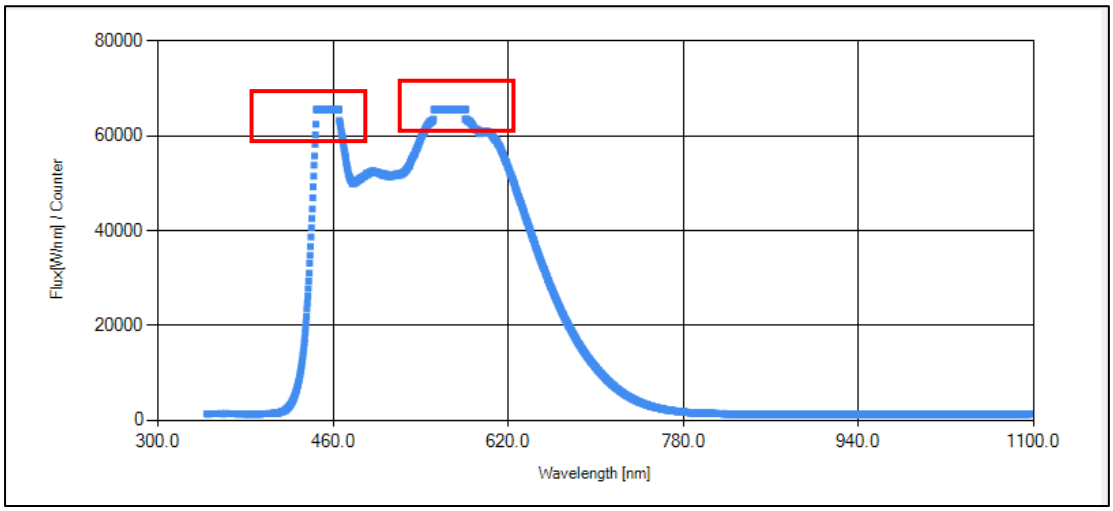


图 8.4 The original response values of the spectrometer are experiencing overexposure (indicated by the red box).

The "Average" refers to the number of consecutive measurements performed by the internal chip of the spectrometer, followed by an arithmetic mean calculation to obtain the measured response value.

Typically, it is recommended to set the average value differently based on the integration time:

When the integration time is greater than 500 ms, it is suggested to set the average value to 1.When the integration time is less than 500 ms, the formula "average value × integration time = 500 ms" can be used to determine the average value. The average value should be rounded to the nearest integer.

In the case of very short integration times, such as 2.5 ms, the above formula suggests an average value of 200. However, in practical testing, it has been observed that averaging beyond 40-50 times does not significantly improve the measurement.

8.4. Basic Functions of the Spectrometer 光谱仪的基本功能

Auto Integration Time 自动积分时间: By calling the LC_AutoIntegration() function, the program interacts with the spectrometer to test the current brightness level. If successful, it retrieves the integration time and average value corresponding to the desired saturation level. This feature helps in automatically determining the optimal integration time and average value based on the current lighting conditions.

Counter Original Response: By using the LC_GetSpectrum() function, the program communicates with the spectrometer to test the current brightness level. If successful, it retrieves

the raw response values from the spectrometer's sensor. These raw response values indicate the sensor's original response and can be used to assess if the integration time is set too high, resulting in overexposure. It's important to note that when a very long integration time, such as 10 seconds, is set, the program needs to wait for the full duration before the spectrometer can receive new instructions. During this waiting period, the thread remains blocked.

Dark Spectrum: The dark spectrum refers to the sensor's response when there is no light incident on the spectrometer's sensor. This represents the intrinsic noise response of the sensor itself. During actual measurements, it is necessary to subtract this dark spectrum to obtain accurate response spectra. Therefore, it is essential to acquire the dark spectrum before conducting measurements to avoid erroneous results.

Additionally, in certain scenarios where the ambient light is not completely dark, it is advisable to subtract the background light to mitigate its influence on the measurements. However, it is crucial that the background light level remains relatively stable and sufficiently dim (below 3%) compared to the measured light source.

Since the dark spectrum response is also dependent on the integration time, it increases with longer integration times. Therefore, during measurements, it is necessary to subtract the dark spectrum data corresponding to the specific integration time used. To fulfill this requirement, two modes are provided for acquiring the dark spectrum: Auto Dark and Once Dark.

Auto Dark: By clicking the "Auto Dark" button and calling the LC_AutoDark() function, you can retrieve the dark spectra within the set integration time (Integration Dark [ms]). As long as the integration time used during measurements does not exceed the integration time set for auto dark, you can use the acquired dark spectra to obtain accurate measurement results.

Since the program needs to collect and process the response states from 1 ms up to the set integration time, the execution time can be relatively long. It is recommended to set the integration dark time between 100 ms and 500 ms, which can be adjusted based on your specific needs. However, it is advisable to keep the maximum integration dark time within 1000 ms.

8.5. Starting a Measurement 开始测量

By clicking the "Start Measurement" button and calling the LC_Measure() function, you can initiate the measurement process. The program will test the current brightness level, and if successful, it will obtain the relative spectral radiance of the current lighting conditions.

After obtaining the relative spectral radiance spectrum, you can use the LC_MeasureData() function to calculate relevant optical parameters based on the acquired relative spectral radiance spectrum. The function will return the corresponding optical data, which can include parameters such as intensity, wavelength, color temperature, or any other relevant optical measurements.