INSION®

integrated spectral solutions

Description of the

Interface Dll

3.28

1. Description of the function library

Convention for calling :	__stdcall with sdcm_std.dll
	__cdecl with sdcm_c.dll

Your compiler needs the "convention calling" to call up the function in the dll. The "convention calling" for the dll is defined by the compiler which created the dll. Do not mistake this convention call with the convention call in the header file of the dll. For more information read also the "help menue" of your compiler.

2. List of function calls

Name:	Ordinal no:
int SDCM_Close (void)	1
int SDCM_CycleMeas (…)	3
int SDCM_GetAdRange(void)	4
int SDCM_GetDefPixel (int DefPixel[5])	5
int SDCM_GetDelay(void)	6
float SDCM_GetDetectorTemp (void)	7
int SDCM_GetDetectorTemperature(void)	8
int SDCM_GetExpanded (XPARAM *ExpParam)	9
int SDCM_GetExpandedSingle (...)	10
int SDCM_GetFit (int FitNo,float *Fit)	11
int SDCM_GetGainBimNir (void)	12
int SDCM_GetGeneral (PARAM *GenParam)	13
int SDCM_GetGeneralSingle (...)	14
float SDCM GetIfactor (void)	15
float SDCM_GetNominalTemp (void)	16
float SDCM_GetOffset(void)	17
float SDCM_GetPfactor (void)	18
int SDCM_GetReceiveTime(void)	19
int SDCM_GetStatus (void)	21
int SDCM_GetTempConfig (void)	23
int SDCM_IllumOff (void)	24
int SDCM_IllumOn (void)	25
int SDCM_Init (int *iComPortNr, int iSearchComPort)	26
int SDCM_ReadCycleMeas (int *Values)	28
int SDCM_ReadDark (int *Data)	29
int SDCM_ReadDiff (int *Data)	30
int SDCM_ReadSpec (int *Data)	21
int SDCM_Save(void)	33
int SDCM_SetAdRange(int iVolt)	34
int SDCM_SetExpanded (XPARAM ExpParam)	35
int SDCM_SetExpandedSingle (...)	36
int SDCM_SetFit (int FitNo,float Fit)	37
int SDCM_SetGain (float Gain)	38
int SDCM_SetGainBimNir (int Gain)	39
int SDCM_SetGeneral (PARAM GenParam)	40
int SDCM_SetGeneralSingle (...)	41
int SDCM_SetIfactor (float Ifactor)	42

Name:	Ordinal no:
int SDCM_SetOffset(float Offset)	44
int SDCM_SetPfactor (float Pfactor)	45
int SDCM_SetTempConfig (int Mode)	49
int SDCM_StartCycleMeas (…)	50
int SDCM_StartMeasDark (int Tint, int Scans)	51
int SDCM_StartMeasDiff (int Tint, int Scans)	52
int SDCM_StartMeasSpec (int Tint, int Scans)	53
int SDCM_StopCycleMeas (void)	55
int SDCM_SystemInfo(…)	56
Int SDCM_TimeProgress(void)	57

NOTE: Some compilers like MS Visual Basic or Borland Delphi needs a ordinal number additional to the function name to identify a function in a dll.

Example for Visual Basic: `Declare Function SDCM_Init& Lib "sdcm_std.dll" Alias "#23" (ByRef ComPortNo As Long, ByVal SearchComPort As Long)`

The “Alias” represents the ordinal number of the function. In this example “#23” for the function “SDCM_Init”.

3. Initialise functions

int SDCM_Init (int *iComPortNr, int iSearchComPort)

Open the specified COM-Port and search for the COM-Port number where the Microspectrometer is connected.
To initialize the spectrometer, the correct port-number (e.g. 1 for COM1) has to be set. To find the spectrometer automatically the parameter “iSearchComPort” must be set to 0.

Inputs:

Variable	Type	Description	Call
iComPortNr	int	Returns the COM-Port number where the device was found	By reference
iSearchComPort	int	Specifies the COM/USB-Port number to search (#1...#999). For automatic search set to 0.	By value

Return Value:

Type	Description
Integer	0 = OK -1 = no device found

NOTE: For spectromter with USB-port the dll is searching for a virtual com port generated by the usb driver.

int SDCM_Close (void)

Close the com port that was specified in SDCM_Init function.

Return Value:

Type	Description
Integer	0 = OK -1 = function failed

4. Parameter functions

NOTE: All parameters will be flashed temporary into the eeprom. If you want to save the parameters permanent use the SDCM_Save() function.

int SDCM_GetGeneral (PARAM *GenParam)

```
struct PARAM
{
char electronic[9]         // type of electronic (SDCM_VIS, MTI_VIS,
                              BIM_NIRP)
int channels;              // number of channels (standard setting 1)
int pixels;                // number of detector pixels
int preheat;               // lamp pre-heat time in ms
int tint;                  // integration time for fast scan (standard setting
                              100)
float fit[5];              // polynomial fit values
}
```

Inputs:

Name	Type	Description	Call
GenParam	PARAM*	Pointer to a structure containing the values	By reference

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = could not read data

int SDCM_GetGeneralSingle (char *electronic[9], int *channels, int *pixels, int *preheat, int *tint, float fit[5])

Use this function instead of "SDCM_GetGeneral" when your compiler not supported structure data types.

int SDCM_SetGeneral (PARAM GenParam)

Sets the general parameter values of the connected spectrometer.

NOTE: Be careful not to set incorrect values! This can cause unexpected measuring data.

Input:

Name	Type	Description	Call
GenParam	PARAM	The structure containing the general parameter values.	By value

Return Value:

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = count of pixel invalid -4 = could not write parameter

int SDCM_SetGeneralSingle (int channels, int pixels, int preheat, int tint, float fit[5])

Use this function instead of “SDCM_SetGeneral” when your compiler not supported structure data types.

int SDCM_GetExpanded (XPARAM *ExpParam)

Gets the expanded parameter values of a connected device.

```
struct XPARAM
{
float fastscan;      // time [ms] to next fastscan
float sensor; /      // choice of image sensor
float gain;          // gain factor of the ADC (min = 1.0 / max = 5.5)
float offset;        // offset of the ADC in mV (factory adjusted)
int parallel;        // do not change
int spi;             // do not change
int lowgain;         // change the capacitor of the detector to low or high
int lamp;            // set the lamp function to enable (1) or disable (0)
int lamplow;         // set the trigger level to low (1) or high (0)
int led;             // not supported. Standard setting 0
}
```

Input:

Name	Type	Description	Call
ExpParam	XPARAM*	Pointer to a structure containing the values	By reference

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = could not read data

int SDCM_GetExpandedSingle (float *fastscan, float *sensor, float *gain, float *offset, int *parallel, int *spi, int *lowgain, int *lamp, int *lamplow, int *led)

Use this function instead of "SDCM_GetExpanded" when your compiler not supported structure data types.

int SDCM_SetExpanded (XPARAM ExpParam)

Sets the expanded parameter values of a connected device.

NOTE: Be careful not to set incorrect values! This can cause unexpected measuring data.

Input:

Name	Type	Description	Call
ExpParam	XPARAM	The structure containing the expanded parameter values	By value

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = could not write parameter

int SDCM_SetExpanded (float fastscan, float sensor, float gain, float offset, int parallel, int spi, int lowgain, int lamp, int lamplow, int led)

Use this function instead of "SDCM_SetExpanded" when your compiler not supported structure data types.

int SDCM_SetGain(float Gain)

Set the gain value of the ADC (Min = 1.0 / Max = 5.5)

Input:

Name	Type	Description	Call
Gain	float	Gain in volt of the ADC	By value

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = could not write parameter

int SDCM_SetAdRange(int iVolt)

Set the input range of the ADC to 2V or 4V.

Input:

Name	**Type**	**Description**	**Call**
iVolt	Int	Input range of the ADC (2 or 4)	By value

Return Values:

Type	**Description**
Integer	0 = OK -1 = no device connected -2 = device busy -3 = could not write parameter -4 = incorrect voltage

int SDCM_GetAdRange(void)

Get the input range of the ADC (2V or 4V).

Return Values:

Type	**Description**
Integer	>0 = range of ADC -1 = no device connected -2 = device busy -3 = could not read adc range

int SDCM_GetDelay(void)

Returns the lamp preheat time in ms.

Return Values:

Type	**Description**
Integer	>0 = lamp preheat time in ms -1 = no device connected -2 = device busy -3 = could not read lamp preheat

int SDCM_SetOffset(float Offset)

Sets the offset of the ADC.

Input:

Name	**Type**	**Description**	**Call**
Offset	float	Offset of the ADC (min= -300; max=+300)	By value

Return Values:

Type	**Description**
Integer	0 = OK -1 = no device connected -2 = device busy

	-3 = could not set offset

float SDCM_GetOffset(void)

Returns the offset of the ADC.

Return Values:

Type	Description
float	-300 < value > +300 = offset value -400 = no device connected -401 = device busy -402 = could not read offset value

Int SDCM_SetFit(int FitNo, float Fit)

Sets the polynomial fit with the exponent "FitNo".

The fits will be used in the formula: wavelength = a * x^2 + b * x^1 + c * x^0

Note: This function is available recently up to firmware version 2.21 with electronic SDCM_VIS. On other electronics at all versions.

Input:

Name	Type	Description	Call
FitNo	Int	Exponent of the fit (min=0;max=4)	By value
Fit	Float	Coefficient of the selected Exponent	By value

Return Values:

Type	Description
Integer	-1 = no device connected -2 = device busy -3 = could not set coefficient 6 = value acknowledged 21 = value not acknowledged

Int SDCM_GetFit(int FitNo, float *Fit)

Returns the coefficient of the polynomial fit with the exponent "FitNo".

Note: This function is available recently up to firmware version 2.21 with electronic SDCM_VIS. On other electronics at all versions.

Return Values:

Type	Description
Integer	-1 = no device connected -2 = device busy -3 = could not read coefficient

int SDCM_Save(void)

Save the parameters non-volatile into the eeprom.

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = could not save parameters

5. Measurement functions

int SDCM_StartMeasDark (int tint, int scans)

Starts the dark measurement. Must be called before SDCM_ReadDark() can be called.

Inputs:

Name	Type	Description	Call
tint	int	Integration time	By value
scans	int	Average scans	By value

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = integration time invalid -4 = count of scans invalid -5 = could not start measurement

int SDCM_ReadDark (int *Data)

Returns the values of the dark measurement. Use the SDCM_GetStatus() function to check when the measurement is finished.

Input:

Name	Type	Description	Call
Data	int*	Pointer to an array (size depends on the number of pixels)	By reference

Return Value:

Type	Description
Integer	0 = OK -1 = no device connected -2 = no dark measurement started -3 = measurement not ready -4 = could not read data

int SDCM_StartMeasSpec (int tint, int scans)

Starts the measurement of a raw spectrum and switch the lamp on. Must be called before SDCM_ReadSpec() can be called.

Input:

Name	Type	Description	Call
tint	int	Integration time	By value
scans	int	Average scans	By value

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = integration time invalid -4 = count of scans invalid -5 = could not start measurement

int SDCM_ReadSpec (int *Data)

Returns the values of the MeasSpec measurement. Use the SDCM_GetStatus() function to check when the measurement is finished.

Input:

Name	Type	Description	Call
Data	int*	Pointer to an array (size depends on the number of pixels)	By reference

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = no reference measurement started -3 = measurement not ready -4 = could not read data

int SDCM_StartMeasDiff (int tint, int scans)

Starts the measurement of difference spectrum and switch the lamp on. Must be called before SDCM_ReadDiff() can be implemented.

Input:

Name	Type	Description	Call
tint	int	Integration time	By value
scans	int	Average scans	By value

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected

	-2 = device busy -3 = no dark measurement started -4 = integration time invalid -5 = count of scans invalid -6 = could not start measurement

int SDCM_ReadDiff (int *Data)

Returns the values of a spectrum that has been subtracted by the measured dark values from the function StartMeasDark. Use the SDCM_GetStatus() function to check when the measurement is finished.

Input:

Name	Type	Description	Call
Data	int*	Pointer to an array (size depends on the number of pixels)	By reference

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = no diff measurement started -3 = measurement not ready -4 = could not read data

int SDCM_GetStatus (void)

Returns the actual state of the connected microspectrometer.

Return Values:

Type	Description
Integer	0 = device idle 1 = dark measurement ready 2 = dark measurement in process 3 = spectrum measurement ready 4 = spectrum measurement in process 5 = diff spectrum measurement ready 6 = diff spectrum measurement in process

6. Miscellaneous functions

int SDCM_IllumOn (void)

Switch the internal light source to on (other than dark measurement). When the microspectrometer is equipped with a trigger interface you get a –5V signal when the parameter “lamp low active” is set to *low* or a +5V signal when the parameter “lamp low active” is set to *high* in the “ExpandedValues” (other than dark measurement).

Return Values:

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = could not send command

bint SDCM_IllumOff (void)

Switch the internal light source to off. When the microspectrometer is equipped with a trigger interface the output signal is 0V.

Return Value

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = could not send command

int SDCM_GetReceiveTime (void)

Returns the time period in ms of a measurement operation.

Return Value

Type	Description
Integer	>0 = time period in ms -1 = invalid time period

**Int SDCM_SystemInfo(char SerialNumberSpectrometer[9],
char SerialNoElectronic[5], char Electronictype[9],
char Firmwareversion[5],char Baudrate[7],
char SerialPort[4],char VersionInterfaceDll[5])**

Returns the hardware information of the spectrometer and the using version of the interface dll.

Input:

Name	Type	Description	Call
SerialNumberSpec-trometer	Char[9]	Pointer to an array	By reference
SerialNoElectronic	Char[5]	Pointer to an array	By reference
Electronic type	Char[9]	Pointer to an array	By reference
Firmware version	Char[5]	Pointer to an array	By reference
Baudrate	Char[7]	Pointer to an array	By reference
Serial port	Char[4]	Pointer to an array	By reference

Version interface dll	Char[5]	Pointer to an array	By reference

Return Value

Type	Description
Integer	0 = OK -1 = no device connected -2 = device busy -3 = error during readout procedure

Int SDCM_TimeProgress(void)

Returns the time in ms that is elapsed since the measurement was started

Return Value

Type	Description
Integer	≥0 = Time in ms -1 = Past time is less than I-time

7. Electronic “BIM_NIRP” functions (Electronic implemented the MSM-NIR 1.7)

int SDCM_ SetGainBimNir (int Gain)

change the capacitor of the detector. (Low = 10 pF / High = 0.5 pF)

Input:

Name	Type	Description	Call
Gain	Int	0 = low 1 = high	By value

Return Values:

Type	Description
Integer	-1 = no device connected -2 = device busy -3 = could not send command 6 = value acknowledge 21 = value not acknowledge

int SDCM_ GetGainBimNir (void)

Returns the gain status of the detector.

Return Values:

Type	Description
Integer	0 = low 1 = high -1 = no device connected -2 = device busy -3 = could not read parameter

int SDCM_ SetTempConfig (int Mode)

Switch the temperature control to on or off.

Input:

Name	**Type**	**Description**	**Call**
Mode	Int	0 = off 2 = on	By value

Return Values:

Type	**Description**
Integer	-1 = no device connected -2 = device busy -3 = could not send command 6 = value acknowledge 21 = value not acknowledge

int SDCM_ GetTempConfig (void)

Returns the mode of the temperature control.

Return Values:

Type	**Description**
Integer	0 = off 2 = on -1 = no device connected -2 = device busy -3 = could not read parameter

int SDCM_ SetNominalTemp (float Temp)

Sets the nominal temperature in °C for regulation the detector temperature.

Input:

Name	**Type**	**Description**	**Call**
Temp	Float	Nominal temperature	By value

Return Values:

Type	**Description**
Integer	-1 = no device connected -2 = device busy -3 = could not send command 6 = value acknowledge 21 = value not acknowledge

float SDCM_ GetNominalTemp (void)

Returns the nominal temperature in °C.

Return Values:

Type	Description
Float	>0 = nominal temperature -1 = no device connected -2 = device busy -3 = could not read parameter

float SDCM_ GetDetectorTemp (void)

Returns the detector temperature.

Return Values:

Type	Description
Float	>0 = detector temperature -1 = no device connected -2 = device busy -3 = could not read parameter

int SDCM_GetDetectorTemperature(float* DetectorTemp)

Returns the detector temperature as call by reference in the pointer "DetectorTemp".

Return Values:

Type	Description
Int	>0 = detector temperature -1 = no device connected -2 = device busy -3 = could not read parameter

int SDCM_ SetPfactor (float Pfactor)

Sets the proportional rate of the detector temperature regulation.

Input:

Name	Type	Description	Call
Pfactor	Float	Proportional factor	By value

Return Values:

Type	Description
Integer	-1 = no device connected -2 = device busy -3 = could not send command 6 = value acknowledge 15 = value not acknowledge

float SDCM_ GetPfactor (void)

Returns proportional rate of the detector temperature regulation.

Return Values:

Type	Description
Float	>0 = Proportional factor -1 = no device connected -2 = device busy -3 = could not read parameter

int SDCM_ SetIfactor (float Ifactor)

Sets the integral rate of the detector temperature regulation.

Input:

Name	Type	Description	Call
Ifactor	Float	Integral factor	By value

Return Values:

Type	Description
Integer	-1 = no device connected -2 = device busy -3 = could not send command 6 = value acknowledge 15 = value not acknowledge

float SDCM_ GetIfactor (void)

Returns the integral rate of the detector temperature regulation.

Return Values:

Type	Description
Float	>0 = Integral factor -1 = no device connected -2 = device busy -3 = could not read parameter

int SDCM_ GetDefPixel (int DefPixel[5])

Returns the number of defective pixels and the defective pixel positions.

Input:

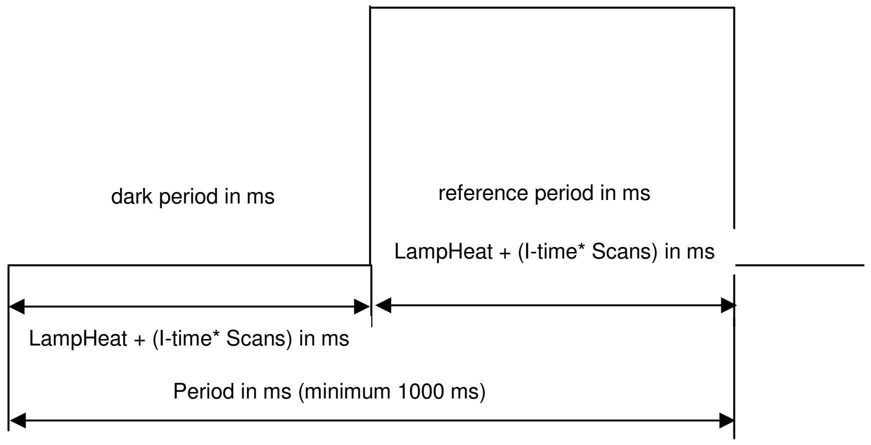
Name	Type	Description	Call
DefPixel[5]	Int	Defective pixel position	By reference

Return Values:

Type	Description
Int	≥0 = Number of defective pixels -1 = no device connected -2 = device busy -3 = could not read parameter

8. Measurement cycle(Only implemented in the electronic "BIM_NIRP")

The measurement cycle automatically repeats itself taking a dark- and a reference measurement.


dark period in ms
reference period in ms
LampHeat + (I-time* Scans) in ms
LampHeat + (I-time* Scans) in ms
Period in ms (minimum 1000 ms)

int SDCM_ StartCycleMeas (int Itime,int Scans,int CycleTime)

Sends the parameter and starts the measurement cycle function.

Input:

Name	Type	Description	Call
Itime	Int	Integration time in ms	By value
Scans	Int	Average scans	By value
CycleTime	Int	Period in ms	By value

Return Values:

Type	Description
Integer	-1 = no device connected -2 = device busy -3 = could not read LampHeat -4 = valid parameters -5 = could not send parameters 6 = OK

NOTE: The result of "LampHeat + (I-time* Scans)" must be smaller than CycleTime/2 !!

int SDCM_ ReadCycleMeas (int *Values)

Readout the values during a measurement cycle. Call up this function in a timer loop with a sample rate of CycleTime/4.

Input:

Name	Type	Description	Call
Values	Int	Dark or reference values	By reference

Return Values:

Type	Description
Integer	-1 = no device connected -2 = device busy -3 = No parameters available -4 = No values received -5 = Incorrect checksum -5 = could not send parameters 0 = Dark values received 1 = Reference values received

int SDCM_ StopCycleMeas (void)

Stops the measurement cycle.
Call up this function until the return value is 6

Return Values:

Type	Description
Integer	-1 = no device connected -2 = Could not stop measurement 6 = OK

int SDCM_ CycleMeas (int Itime,int Scans,int Averages,float* Dark,float* Ref)

Starts a cycles measurements that takes “Averages” reference scans and “Averages+1” dark scans. The function returns the mean dark and the mean reference data of the scans.

Input:

Name	Type	Description	Call
Itime	Int	Integration time in ms	By value
Scans	Int	Internal Average scans	By value
Averages	Int	Number of reference signals	By value
Dark	Float	Mean dark	By reference
Ref	Float	Mean reference	By reference

Return Values:

Type	Description
Integer	-1 = no device connected -2 = device busy -3 = Averages <1 or >256 -4 = error during sending para-meters -5 = not enough memory -6 = time limit exceeded 6 = OK