NSP32 SDK®
Reference Manual
for C/C++

ver 1.7

nanoLambda

IMPORTANT NOTICE

nanoLambda Korea and its affiliates (“nanoLambda”) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to nanoLambda’s terms and conditions of sale supplied at the time of order acknowledgment. Customers are responsible for their products and applications using any nanoLambda products. nanoLambda does not warrant or represent that any license, either express or implied, is granted under any nanoLambda patent right, copyright, mask work right, or other nanoLambda intellectual property right relating to any combination, machine, or process in which nanoLambda products or services are used. Information published by nanoLambda regarding third-party products or services does not constitute a license from nanoLambda to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from nanoLambda under the patents or other intellectual property of nanoLambda. Reproduction of nanoLambda information in nanoLambda documents or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. nanoLambda is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions. Resale of nanoLambda products is not allowed without written agreement. Decompiling, disassembling, reverse engineering or attempt to reconstruct, identify or discover any source code, underlying ideas, techniques or algorithms are not allowed by any means. nanoLambda products are not authorized for use in safety-critical applications. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of nanoLambda products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by nanoLambda. Further, buyers must fully indemnify nanoLambda and its representatives against any damages arising out of the use of nanoLambda products in such safety-critical applications.

Table of Contents

About This Manual

Document Purpose and Intended User

This document provides you with APIs reference manual for NSP32 SDK® .

What's New in this Manual

This version of the *NSP32 SDK® Reference Manual* is updated for Release 1.7.

Document Summary

Chapter	Description
Chapter 1: *Overview*	Provides tables for low-level and high-level APIs
Chapter 2: *Low-Level API Reference*	Provides a low-level API Reference of the NSP32 SDK®.
Chapter 3: *High-Level API Reference*	Provides a high-level API Reference of the NSP32 SDK®
Chapter 4: *COS-Level API Reference*	Provides a COS(Core-Open-Source) API Reference of the NSP32 SDK®

Chapter 1 Overview

Low-Level API

Base API Functions

EXPORT_DLL char *bsGetErrorString (int error_code)
Convert error code to string.

Device API Functions

EXPORT_DLL int duConnect ()
Connect to physical device(sensor).

EXPORT_DLL int duDisconnect ()
Disconnect physical device(sensor).

EXPORT_DLL int duGetSensorList (char **sensor_list_out)
Get sensor ID list which are connected to system.

EXPORT_DLL int duActivateSensorWithIndex (int sensor_index)
Activate a specific sensor.

EXPORT_DLL int duActivateSensorWithID (const char *sensor_id_str)
Activate a specific sensor.

EXPORT_DLL int duGetTotalSensors ()
Get devices count.

EXPORT_DLL int duGetSensorID (char *sensor_id)
Get sensor ID of physical device(sensor).

EXPORT_DLL int duGetShutterSpeed ()
Get shutter speed.

EXPORT_DLL int duGetShutterSpeedLimits (int *min_limit, int *max_limit)
Get shutter speed limits.

EXPORT_DLL int duShutterSpeedToExposureTime (int master_clock, int shutter_speed, double *exposure_time_val)
Convert shutter speed to exposure time (unit: ms).

EXPORT_DLL int duExposureTimeToShutterSpeed (int master_clock, double exposure_time_val, int *shutter_speed)
Convert exposure time (ms) to shutter speed.

EXPORT_DLL int duSetShutterSpeed (int shutter)
Change current shutter speed of a device(sensor).

EXPORT_DLL int duGetOptimalShutterSpeed ()
Get optimal shutter speed by AE.

EXPORT_DLL int duGetFilterData (double *output, int number_of_averages)
Get raw filter data from physical device(sensor).

EXPORT_DLL int duGetNumOfFilters (int *num_filter)
Get total number of nano-filters.
EXPORT_DLL int duSetSensorParameters (int _adc_gain, int _adc_range)
Set sensor registers.
EXPORT_DLL int duGetSensorParameters (int *_adc_gain, int *_adc_range)
Get settings of sensor registers.

Core Spectrum Functions

EXPORT_DLL int csCreate (void)
Initialize Core Spectrum C API.
EXPORT_DLL int csDestroy (void)
Destroy Core Spectrum C API
EXPORT_DLL int csRegister (const char *sensor_cal_file_path)
Add one sensor to sensor calibration data container (DB).
EXPORT_DLL int csGetSensorList (char **sensor_list_out)
Get sensor ID list.
EXPORT_DLL int csGetSensorID (char *sensor_id_str)
Get sensor ID of sensor calibration data.
EXPORT_DLL int csCapacity(void)
Get total sensors in the sensor data list
EXPORT_DLL int csActivateSensorWithID (const char *sensor_id_str)
Activate one sensor data.
EXPORT_DLL int csGetWavelengthInfo (double *start_wavelength, double *end_wavelength, double *interval_wavelength)
Get wavelength information data.
EXPORT_DLL int csGetResolution (double *resolution)
Get spectrum resolution.
EXPORT_DLL int csGetSpectrumLength (void)
Get size of spectrum data.
EXPORT_DLL int csGetSensorParameters (int *adc_gain, int *adc_range)
Get register settings.
EXPORT_DLL int csSetBackground (double *background_data)
Set background filter data.
EXPORT_DLL int csCalculateSpectrum (double *filter_input, int cur_ss, double *spec_output, double *wavelength_output)
Calculate spectrum.

High-Level API

High-Level API Functions

EXPORT_DLL int sdkCreate (void)
Create SDK Object.

EXPORT_DLL int sdkDestroy ()
Destroy SDK Object.

EXPORT_DLL int sdkAdd (const char *sensor_cal_file_path)
Add one sensor to active sensor list.

EXPORT_DLL int sdkGetSensorIDFromDevice (char *sensor_id_str)
Get sensor ID of physical device(sensor)

EXPORT_DLL int sdkGetSensorIDFromCalData (char *sensor_id_str)
Get sensor ID of sensor calibration data.

EXPORT_DLL int sdkActivateSensorWithIndex (int sensor_index)
Activate a specific sensor with index.

EXPORT_DLL int sdkActivateSensorWithID (const char *sensor_id_str)
Activate a specific sensor.

EXPORT_DLL int sdkGetShutterSpeed ()
Get shutter speed.

EXPORT_DLL int sdkSetShutterSpeed (int shutter)
Change current shutter speed of a device(sensor)

EXPORT_DLL int sdkGetShutterSpeedLimits (int *min_limit, int *max_limit)
Get Shutter Speed Limist.

EXPORT_DLL int sdkShutterSpeedToExposureTime (int master_clock, int shutter_speed_val, double *exposure_time)
Convert shutter speed to exposure time(msec)

EXPORT_DLL int sdkExposureTimeToShutterSpeed (int master_clock, double exposure_time, int *shutter_speed_val)
Convert exposure time (ms) to shutter speed.

EXPORT_DLL int sdkGetOptimalShutterSpeed ()
Get optimal shutter speed by AE.

EXPORT_DLL int sdkGetWavelengthInfo (double *start_wavelength, double *end_wavelength, double *interval_wavelength)
Get wavelength information data.

EXPORT_DLL int sdkGetResolution (double *resolution)
Get spectrum resolution.

EXPORT_DLL int sdkGetSpectrumLength ()
Get size of spectrum data.

EXPORT_DLL int sdkCalculateSpectrum (int cur_ss, int frame_averages, double *spec_output, double *wavelength_output)
Calculate spectrum.

Chapter 2 Low-Level API Reference

Base API

Functions

EXPORT_DLL	char *bsGetErrorString (int error_code)
	Convert error code to string.

Function Documentation

bsGetErrorString()

EXPORT_DLL char* bsGetErrorString(int ***error_code***);)

Convert error code to string.

This function returns one string for a specific error code.

```
int error_code = cc_ec_no_device_in_system;
char* error_string = bsGetErrorString(error_code);
```

Parameters

error_code - error code which defined in 'nsp_error_codes.h' [IN]

Returns

Returns one numeric value of:

- error string

See also

Device API

Functions

EXPORT_DLL	int duConnect ()
	Connect to physical device(sensor).
EXPORT_DLL	int duDisconnect ()
	Disconnect physical device(sensor).
EXPORT_DLL	int duGetSensorList (char **sensor_list_out)
	Get sensor ID list which are connected to system.
EXPORT_DLL	int duActivateSensorWithIndex (int sensor_index)
	Activate a specific sensor.
EXPORT_DLL	int duActivateSensorWithID (const char *sensor_id_str)
	Activate a specific sensor.
EXPORT_DLL	int duGetTotalSensors ()
	Get devices count.
EXPORT_DLL	int duGetSensorID (char *sensor_id)
	Get sensor ID of physical device(sensor).
EXPORT_DLL	int duGetShutterSpeed ()
	Get shutter speed.
EXPORT_DLL	int duGetShutterSpeedLimits (int *min_limit, int *max_limit)
	Get shutter speed limits.
EXPORT_DLL	int duShutterSpeedToExposureTime (int master_clock, int shutter_speed, double *exposure_time_val)
	Convert shutter speed to exposure time (unit: ms).
EXPORT_DLL	int duExposureTimeToShutterSpeed (int master_clock, double exposure_time_val, int *shutter_speed)
	Convert exposure time (ms) to shutter speed.
EXPORT_DLL	int duSetShutterSpeed (int shutter)
	Change current shutter speed of a device(sensor).
EXPORT_DLL	int duGetOptimalShutterSpeed ()
	Get optimal shutter speed by AE.
EXPORT_DLL	int duGetFilterData (double *output, int number_of_averages)
	Get raw filter data from physical device(sensor).
EXPORT_DLL	int duGetNumOfFilters (int *num_filter)
	Get total number of nano-filters.
EXPORT_DLL	int duSetSensorParameters (int _adc_gain, int _adc_range)
	Set sensor registers.
EXPORT_DLL	int duGetSensorParameters (int *_adc_gain, int *_adc_range)
	Get settings of sensor registers.

Function Documentation

duActivateSensorWithID

EXPORT_DLL int duActivateSensorWithID(const char * ***sensor_id_str***)

Activate a specific sensor.

This function activates a specific sensor having a matched sensor ID among multiple sensors if there are multiple sensors in system.

```
char* sensor_id_str = "Y8585-1-85-85-0";
int ret_value = duActivateSensorWithID(sensor_id_str);
```

Parameters

sensor_id_str - sensor ID string [IN]

Returns

Returns one numeric value of:

- sensor index (>=0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duActivateSensorWithIndex

duActivateSensorWithIndex()

EXPORT_DLL int duActivateSensorWithIndex(int ***sensor_index***)

Activate a specific sensor.

This function activates a specific sensor having a matched sensor index among multiple sensors if there are multiple sensors in system.

```
int sensor_index = 0; // default = 0 (1'th sensor)
int ret_value = duActivateSensorWithIndex(sensor_index);
```

Parameters

sensor_index - sensor index start from 0 [IN]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duActivateSensorWithID

duConnect()

EXPORT_DLL int duConnect ()

Connect to physical device(sensor)

This function try to connect to physical device(sensor) and returns the total number of sensors in system.

```
int num_of_sensors = duConnect();
```

Parameters

void

Returns

Returns one numeric value of:

- total number of sensors in system (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_fail_to_create_device_obj
- cc_ec_no_device_in_system

See also

duDisconnect

duDisconnect()

EXPORT_DLL int duDisconnect()

Disconnect physical device(sensor)

This function disconnects physical device(sensor).

```
int ret_value = duDisconnect();
```

Parameters

void

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)

See also

duConnect

duExposureTimeToShutterSpeed()

EXPORT_DLL int duExposureTimeToShutterSpeed(int ***master_clock***,
double ***exposure_time_val***, int * ***shutter_speed***)

Convert exposure time (ms) to shutter speed.

This function converts exposure time to shutter speed based on MASTER CLOCK of sensor.

```
int master_clock = 5; // MCLK = 5-MHz
double exposure_time_val = 100; // 100 msec
int shutter_speed = 0
cur_ss = duExposureTimeToShutterSpeed(master_clock,
                exposure_time_val, &shutter_speed);
```

Parameters

master_clock	- master clock of MCU to sensor [IN]
exposure_time_val	- exposure time value (ms) [IN]
shutter_speed	- int pointer to shutter speed value [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duShutterSpeedToExposureTime

duGetFilterData()

EXPORT_DLL int duGetFilterData (double * ***output***, int ***number_of_averages***)

Get raw filter data from physical device(sensor).

This function acquires one raw filter data from physical device(sensor).

```
double filter_data[SENSOR_DATA_SIZE];
int frame_averages = 50;
int ret_value = duGetFilterData(filter_data, frame_averages);
```

Parameters

output	- double pointer to raw filter data [OUT]

number_of_averages - number of frame averages [IN]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duSetShutterSpeed, duSetSensorParameters

duGetNumOfFilters()

EXPORT_DLL int duGetNumOfFilters (int * ***num_filter***)

Get total number of nano-filters.

This function returns the total number of nano-filters in a physical device(sensor).

```
int num_of_filters = 0;
int ret_value = duGetNumOfFilters(&num_of_filters);
```

Parameters

num_filter - int pointer to the number of nano-filters [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duGetFilterData

duGetOptimalShutterSpeed()

EXPORT_DLL int duGetOptimalShutterSpeed()

Get optimal shutter speed by AE.

This function returns an optimal shutter speed value which found by AE(Auto-Exposure) function.

```
int optimal_ss = 0;
optimal_ss = duGetOptimalShutterSpeed();
```

Parameters

void

Returns

Returns one numeric value of:

- optimal shutter speed (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

csGetSpectrumLength, csGetResolution, csCalculateSpectrum

duGetSensorParameters()

EXPORT_DLL int duGetSensorParameters(int * ***_adc_gain***, int * ***_adc_range***)

Get sensor parameters(registers settings).

This function returns the current settings of sensor registers (ADC gain, ADC range, and ADC resolution).

```
int adc_gain = 1; // 1=1X (default), 0=4X
int adc_range = 132; // 132 (default)
int ret_value = duGetSensorParameters(&adc_gain, &adc_range);
```

Parameters

_adc_gain - integer pointer to ADC gain value [OUT]

_adc_range - integer pointer to ADC range value [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duSetSensorParameters

duGetSensorID()

EXPORT_DLL int duGetSensorID(char * ***sensor_id***)

Get sensor ID of physical device(sensor)

This function returns a sensor ID of currently activated physical device(sensor). Current sensor is an activated one by duActivateSensorWithIndex() function.

```
char sensor_id[SENSOR_ID_STRING_LENGTH];
int ret_value = duGetSensorID(sensor_id);
```

Parameters

pointer to char array for sensor ID data [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_memory_is_null
- cc_ec_device_object_is_null

See also

duActivateSensorWithIndex

duGetSensorList()

EXPORT_DLL int duGetSensorList(char ** ***sensor_list_out***)

Get sensor ID list which are connected to system.

This function returns a string list containing sensor IDs in your system.

int duGetSensorList(char** sensor_list_out);

Parameters

sensor_list_out - char pointer of pointer to sensor list [OUT]

Returns

Returns one numeric value of:

- number of total count of sensors in your system

See also

duConnect, duDisconnect

duGetShutterSpeed()

EXPORT_DLL int duGetShutterSpeed ()

Get shutter speed.

This function returns a shutter speed value of currently activated sensor.

```
int cur_ss = 0;
cur_ss = duGetShutterSpeed();
```

Parameters

void

Returns

Returns one numeric value of:

- current shutter speed (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duSetShutterSpeed

duGetShutterSpeedLimits()

EXPORT_DLL int duGetShutterSpeedLimits (int * ***min_limit***,
int * ***max_limit***
)

Get shutter speed limits.

This function returns two limit values for shutter speed (minimum and maximum).

```
int ss_min, ss_max;
cur_ss = duGetShutterSpeedLimits(&ss_min, &ss_max);
```

Parameters

ss_min - int pointer to minimum shutter speed value [OUT]

ss_max - int pointer to maximum shutter speed value [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duSetShutterSpeed

duGetTotalSensors()

EXPORT_DLL int duGetTotalSensors

Get devices count.

This function returns a total number of devices in system.

```
int total_num_of_sensors = duGetTotalSensors();
```

Parameters

void

Returns

Returns one numeric value of:

- total number of devices (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duSetSensorParameters()

EXPORT_DLL int duSetSensorParameters (int *_adc_gain*, int *_adc_range*)

Set sensor registers.

This function changes sensor parameters(registers) for ADC gain and ADC range.

```
int adc_gain = 1; // 1=1X (default), 0=4X
int adc_range = 132; // 132 (default)
int ret_value = duSetSensorParameters(adc_gain, adc_range);
```

Parameters

_adc_gain - ADC gain value [IN]

_adc_range - ADC range value [IN]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duGetSensorParameters

duSetShutterSpeed()

EXPORT_DLL int duSetShutterSpeed (int ***shutter***)

Change current shutter speed of a device(sensor)

This function changes shutter speed of currently activated device(sensor).

```
int new_ss = 50;
int ret_value = duSetShutterSpeed(new_ss);
```

Parameters

shutter - shutter speed value [IN]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duGetShutterSpeed

duShutterSpeedToExposureTime()

EXPORT_DLL int duShutterSpeedToExposureTime (int ***master_clock***,
int ***shutter_speed***,
double * ***exposure_time_val***
)

Convert shutter speed to exposure time (unit: ms)

This function converts shutter speed to exposure time based on MASTER CLOCK of sensor.

```
int master_clock = 5; // MCLK = 5-MHz
int shutter_speed = 1000
double exposure_time_val = 0;
cur_ss = duShutterSpeedToExposureTime(master_clock,
                                      shutter_speed,
                                      &exposure_time_val);
```

Parameters

master_clock	- master clock of MCU to sensor [IN]
shutter_speed	- shutter speed value [IN]
exposure_time_val	- double pointer to exposure time value [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duExposureTimeToShutterSpeed

Core Spectrum API

Functions

EXPORT_DLL	int csCreate (void)
	Initialize Core Spectrum C API.
EXPORT_DLL	int csDestroy (void)
	Destroy Core Spectrum C API
EXPORT_DLL	int csRegister (const char *sensor_cal_file_path)
	Add one sensor to sensor calibration data container (DB).
EXPORT_DLL	int csGetSensorList (char **sensor_list_out)
	Get sensor ID list.
EXPORT_DLL	int csGetSensorID (char *sensor_id_str)
	Get sensor ID of sensor calibration data.
EXPORT_DLL	int csActivateSensorWithID (const char *sensor_id_str)
	Activate one sensor data.
EXPORT_DLL	int csGetWavelengthInfo (double *start_wavelength, double *end_wavelength, double *interval_wavelength)
	Get wavelength information data.
EXPORT_DLL	int csGetResolution (double *resolution)
	Get spectrum resolution.
EXPORT_DLL	int csGetSpectrumLength (void)
	Get size of spectrum data.
EXPORT_DLL	int csGetSensorParameters (int *adc_gain, int *adc_range)
	Get register settings.
EXPORT_DLL	int csSetBackground (double *background_data)
	Set background filter data.
EXPORT_DLL	int csCalculateSpectrum (double *filter_input, int cur_ss, double *spec_output, double *wavelength_output)
	Calculate spectrum.

Function Documentation

csActivateSensorWithID()

EXPORT_DLL int csActivateSensorWithID (const char * *sensor_id_str*)

Activate one sensor data.

This function activates one sensor data of input sensor ID.

```
char sensor_id_str[SENSOR_ID_STRING_LENGTH];
int ret_val = csActivateSensorWithID(sensor_id_str);
```

Parameters

sensor_id_str - char buffer to contain sensor ID [OUT]

Returns

Returns one numeric value of NSP_RETURN_VALUE_SUCCESS.

- NSP_RETURN_VALUE_SUCCESS (1).
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null
- cc_ec_memory_is_null

See also

csRegister

csRegister()

EXPORT_DLL int csRegister (const char * *sensor_cal_file_path*)

Add one sensor to sensor calibration data container (DB).

This function add one sensor to Core Spectrum object by loading sensor calibration data file.

```
// 'Y8585-1-85-85-0' is a sensor ID of a specific physical sensor.
char* sensor_cal_data_path = "./config/sensor_Y8585-1-85-85-0.dat";
int ret_val = csRegister(sensor_cal_data_path);
```

Parameters

sensor_cal_file_path - sensor calibration file path [IN]

Returns

Returns one numeric values of NSP_RETURN_VALUE_SUCCESS, NSP_RETURN_VALUE_FAILURE, or exception.

- the number of sensors in the sensor cal data container(>0).

- NSP_RETURN_VALUE_FAILURE.
- cc_ec_fail_to_create_core_spectrum_obj.
- cc_ec_sensor_id_is_empty.

See also

csCreate, csDestroy

csCalculateSpectrum()

EXPORT_DLL int csCalculateSpectrum (double * ***filter_input***,
int ***cur_ss***,
double * ***spec_output***,
double * ***wavelength_output***
)

Calculate spectrum.

This function returns a calculated spectrum and wavelength data with input filter data.

```
int cur_ss = 50;
double filter_data[SENSOR_DATA_SIZE];
// acquire raw filter data from sensor with shutter speed 'cur_ss'
// get spectrum length
int spectrum_length = csGetSpectrumLength();
double* spectrum_data = (double*)malloc(sizeof(double)*spectrum_length);
double* wavelength_data =  (double*)malloc(sizeof(double)
                                          * spectrum_length);
int ret_value = csCalculateSpectrum(filter_data,
                                    cur_ss,
                                    spectrum_data,
                                    wavelength_data );
```

Parameters

void

Returns

Returns one numeric value of:

- spectrum data size(length) (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null

See also

csGetSpectrumLength, csGetResolution, csCalculateSpectrum

csCreate()

EXPORT_DLL int csCreate (void)

Initialize Core Spectrum C API.

This function create a core spectrum object.

```
int ret_val = csCreate();
```

Parameters

void

Returns

Returns one numeric values of NSP_RETURN_VALUE_SUCCESS, NSP_RETURN_VALUE_FAILURE, or exception.

- NSP_RETURN_VALUE_SUCCESS.
- NSP_RETURN_VALUE_FAILURE.
- cc_ec_fail_to_create_core_spectrum_obj.

See also

csDestroy

csDestroy()

EXPORT_DLL int csDestroy (void)

Destroy Core Spectrum C API.

This function finalize a core spectrum object.

```
int ret_val = csDestroy();
```

Parameters

void

Returns

Returns one numeric value of NSP_RETURN_VALUE_SUCCESS.

- NSP_RETURN_VALUE_SUCCESS.

See also

csCreate

csGetSensorParameters()

EXPORT_DLL int csGetSensorParameters (int * ***adc_gain***,
int * ***adc_range***
)

Get register settings.

This function returns ADC register settings from physical device (sensor).

```
int adc_gain, adc_range;
int ret_value = csGetSensorParameters(&adc_gain, &adc_range);
```

Parameters

adc_gain - pointer to ADC gain info: 0 or 1(default) [OUT]

adc_range - pointer to ADC range info (0~255) [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null

See also

csGetWavelengthInfo, csGetResolution

csGetResolution()

EXPORT_DLL int csGetResolution (double * ***resolution***)

Get spectrum resolution.

This function returns spectrum resolution information.

```
double spectrum_resolution;
int ret_val = csGetResolution(&spectrum_resolution);
```

Parameters

spectrum_resolution - double pointer to spectrum resolution [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null

See also

csGetWavelengthInfo

csGetSensorID()

EXPORT_DLL int csGetSensorID (char * ***sensor_id_str***)

Get sensor ID of sensor calibration data.

This function returns sensor ID in sensor calibration data file.

```
char sensor_id_str[SENSOR_ID_STRING_LENGTH];
int ret_val = csGetSensorID(sensor_id_str);
```

Parameters

sensor_id_str - char buffer to contain sensor ID [OUT]

Returns

Returns one numeric value of NSP_RETURN_VALUE_SUCCESS.

- the length of sensor ID string (>0).
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null
- cc_ec_memory_is_null

See also

csRegister

csGetSensorList()

EXPORT_DLL int csGetSensorList (char ** ***sensor_list_out***)

Get sensor ID list.

This function returns 2D string array havin sensor IDs.

```
char **sensor_id_list[1][SENSOR_ID_STRING_LENGTH];
int ret_val = csGetSensorList(sensor_id_list);
```

Parameters

sensor_list_out - char pointer of pointer [OUT]

Returns

Returns one numeric values of NSP_RETURN_VALUE_SUCCESS, NSP_RETURN_VALUE_FAILURE, or exception.

- the number of sensors in the sensor cal data container(>0).
- NSP_RETURN_VALUE_FAILURE.
- cc_ec_fail_to_create_core_spectrum_obj.
- cc_ec_sensor_id_is_empty.

See also

csCreate, csDestroy

csCapacity ()

EXPORT_DLL int csCapacity (void)

Get total number of sensors in the sensor data list.

This function returns a total number of sensors in the sensor calibration data list.

```
int total_count = csGetSensorList(sensor_id_list);
```

Parameters

Returns

Returns one numeric values of total count of sensors, NSP_RETURN_VALUE_FAILURE, or exception.

- the number of sensors in the sensor cal data container(>0).
- NSP_RETURN_VALUE_FAILURE.
- cc_ec_fail_to_create_core_spectrum_obj.
- cc_ec_sensor_id_is_empty.

See also

csCreate, csDestroy

csGetSpectrumLength()

EXPORT_DLL int csGetSpectrumLength (void)

Get size of spectrum data.

This function returns the size(length) of spectrum data.

```
int spectrum_length;
spectrum_length = csGetSpectrumLength();
```

Parameters

void

Returns

Returns one numeric value of:

- spectrum data size(length) (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null

See also

csGetWavelengthInfo, csGetResolution, csCalculateSpectrum

csGetWavelengthInfo()

EXPORT_DLL int csGetWavelengthInfo (double * ***start_wavelength***,
double * ***end_wavelength***,
double * ***interval_wavelength***
)

Get wavelength information data.

This function returns wavelength range information for spectrum data.

```
double start_wavelength, end_wavelength, wavelength_interval;
int ret_val = csGetWavelengthInfo(&start_wavelength,
                                  &end_wavelength,
                                  &wavelength_interval);
```

Parameters

start_wavelength	- double pointer to start wavelength [OUT]
end_wavelength	- double pointer to end wavelength [OUT]
wavelength_interval	- double pointer to wavelength interval[OUT]

Returns

Returns one numeric value of NSP_RETURN_VALUE_SUCCESS.

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null

See also

csGetResolution

csSetBackground()

EXPORT_DLL int csSetBackground (double * ***background_data***)

Set background filter data.

This function set input filter data as background data. Background data will be used to correct background signal before spectrum calculation.

```
double filter_data[SENSOR_DATA_SIZE];
// acquire raw filter data at SS=1
int ret_value = csSetBackground((double *)filter_data);
```

Parameters

background_data	- double pointer to raw filter data [IN]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null
- cc_ec_background_buffer_is_null

See also

csCalculateSpectrum

Chapter 3 High-Level API Reference

Functions

EXPORT_DLL	int sdkCreate (void)
	Create SDK Object.
EXPORT_DLL	int sdkDestroy ()
	Destroy SDK Object.
EXPORT_DLL	int sdkAdd (const char *sensor_cal_file_path)
	Add one sensor to active sensor list.
EXPORT_DLL	int sdkGetSensorIDFromDevice (char *sensor_id_str)
	Get sensor ID of physical device(sensor)
EXPORT_DLL	int sdkGetSensorIDFromCalData (char *sensor_id_str)
	Get sensor ID of sensor calibration data.
EXPORT_DLL	int sdkActivateSensorWithIndex (int sensor_index)
	Activate a specific sensor with index.
EXPORT_DLL	int sdkActivateSensorWithID (const char *sensor_id_str)
	Activate a specific sensor.
EXPORT_DLL	int sdkGetShutterSpeed ()
	Get shutter speed.
EXPORT_DLL	int sdkSetShutterSpeed (int shutter)
	Change current shutter speed of a device(sensor)
EXPORT_DLL	int sdkGetShutterSpeedLimits (int *min_limit, int *max_limit)
	Get Shutter Speed Limist.
EXPORT_DLL	int sdkShutterSpeedToExposureTime (int master_clock, int shutter_speed_val, double *exposure_time)
	Convert shutter speed to exposure time(msec)
EXPORT_DLL	int sdkExposureTimeToShutterSpeed (int master_clock, double exposure_time, int *shutter_speed_val)
	Convert exposure time (ms) to shutter speed.
EXPORT_DLL	int sdkGetOptimalShutterSpeed ()
	Get optimal shutter speed by AE.
EXPORT_DLL	int sdkGetWavelengthInfo (double *start_wavelength, double *end_wavelength, double *interval_wavelength)
	Get wavelength information data.
EXPORT_DLL	int sdkGetResolution (double *resolution)
	Get spectrum resolution.
EXPORT_DLL	int sdkGetSpectrumLength ()
	Get size of spectrum data.
EXPORT_DLL	int sdkCalculateSpectrum (int cur_ss, int frame_averages, double *spec_output, double *wavelength_output)
	Calculate spectrum.
EXPORT_DLL	int sdkCalculateColor (double *spectrum, double *wavelength, int spectrum_length, double *X, double *Y, double *Z, double *r, double *g, double *b, double *x, double *y, double *z, double *cct)

Function Documentation

sdkActivateSensorWithID()

EXPORT_DLL int sdkActivateSensorWithID (const char * ***sensor_id_str***)

Activate a specific sensor.

This function activates a specific sensor having a matched sensor index among multiple sensors if there are multiple sensors in system.

```
const char* sensor_id_str = "Y8585-1-85-85-0";
int ret_value = sdkActivateSensorWithID(sensor_id_str);
```

Parameters

sensor_id_str - sensor ID string [IN]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duConnect, duDisconnect

sdkActivateSensorWithIndex()

EXPORT_DLL int sdkActivateSensorWithIndex (int ***sensor_index***)

Activate a specific sensor with index.

This function activates a specific sensor having a matched sensor index among multiple sensors if there are multiple sensors in system.

```
int sensor_index = 0; // default = 0 (1'th sensor)
int ret_value = sdkActivateSensorWithIndex(sensor_index);
```

Parameters

sensor_index - sensor index start from 0 [IN]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duConnect, duDisconnect

sdkAdd()

EXPORT_DLL int sdkAdd (const char * ***sensor_cal_file_path***)

Add one sensor to active sensor list.

This function add one sensor to active sensor list by loading sensor calibration data.

```
const char* sensor_cal_file_path = "./config/sensor_Y8585-1-85-85-0.dat";
int total_count_of_active_sensors = sdkAdd(sensor_cal_file_path);
```

Parameters

sensor_cal_file_path - file path for sensor calibration data [IN]

Returns

Returns one numeric value of:

- total count of active sensors
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_no_device_in_system
- cc_ec_fail_to_load_sensor_cal_data
- cc_ec_fail_to_initialize_color_object

See also

sdkCreate, sdkDestroy

sdkCalculateSpectrum()

EXPORT_DLL int sdkCalculateSpectrum (int ***cur_ss***,
int ***frame_averages***,
double * ***spec_output***,
double * ***wavelength_output***
)

Calculate spectrum.

This function returns a calculated spectrum and wavelength data with input filter data.

```
int cur_ss = 50;
int frame_averages = 50;
// get spectrum length
int spectrum_length = sdkGetSpectrumLength();
double* spectrum_data = (double*)malloc(sizeof(double)*spectrum_length);
double* wavelength_data = (double*)malloc(sizeof(double)*
                                  spectrum_length);
int ret_value = sdkCalculateSpectrum(cur_ss, frame_averages,
                                 spectrum_data, wavelength_data );
```

Parameters

cur_ss	- current shutter speed [IN]
frame_averages	- frame count for averaging [IN]
spec_output	- double pointer to spectrum data [OUT]
wavelength_output	- double pointer to wavelength data [OUT]

Returns

Returns one numeric value of:

- spectrum data size(length) (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_memory_for_spectrum_data_buffer_is_null
- cc_ec_memory_for_wavelength_data_buffer_is_null

See also

sdkGetSpectrumLength, sdkGetResolution, sdkGetWavelengthInfo

sdkCreate()

EXPORT_DLL int sdkCreate (void)

Create SDK Object.

This function create one SDK object.

```
int ret_value = sdkCreate();
```

Parameters

void

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_no_device_in_system
- cc_ec_fail_to_load_sensor_cal_data
- cc_ec_fail_to_initialize_color_object

See also

sdkDestroy

sdkDestroy()

EXPORT_DLL int sdkDestroy ( )

Destroy SDK Object.

This function destroy SDK object.

```
int ret_value = sdkDestroy();
```

Parameters

void

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)

See also

sdkCreate

sdkExposureTimeToShutterSpeed()

EXPORT_DLL int sdkExposureTimeToShutterSpeed (int *master_clock*,
double *exposure_time*,
int * *shutter_speed_val*
)

Convert exposure time (ms) to shutter speed.

This function converts a exposure time (unit: msec) to shutter speed value based on sensor's master clock.

```
int master_clock = 5; // 5-MHz
double exposure_time = 100.0;
int shutter_speed = 0;
int ret_value = sdkShutterSpeedToExposureTime(cur_ss, &exposure_time );
```

Parameters

master_clock - master clock of MCU to sensor (IN)

exposure_time - exposure time value (msec) (IN)

shutter_speed_val - int pointer to shutter speed (OUT)

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)

See also

sdkGetShutterSpeed, sdkSetShutterSpeed, sdkGetShutterSpeedLimits, sdkShutterSpeedToExposureTime

sdkGetOptimalShutterSpeed()

EXPORT_DLL int sdkGetOptimalShutterSpeed ( )

Get optimal shutter speed by AE.

This function returns an optimal shutter speed value which found by AE(Auto-Exposure) function.

```
int optimal_ss = 0;
optimal_ss = sdkGetOptimalShutterSpeed();
```

Parameters

void

Returns

Returns one numeric value of:

- optimal shutter speed (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

sdkGetShutterSpeed, sdkSetShutterSpeed

sdkGetResolution()

EXPORT_DLL int sdkGetResolution (double * ***resolution***)

Get spectrum resolution.

This function returns spectrum resolution information.

```
double spectrum_resolution;
int ret_val = sdkGetResolution(&spectrum_resolution);
```

Parameters

spectrum_resolution - double pointer to spectrum resolution [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null

See also

sdkGetSpectrumLength

sdkGetSensorIDFromCalData()

EXPORT_DLL int sdkGetSensorIDFromCalData (char * ***sensor_id_str***)

Get sensor ID of sensor calibration data.

This function returns sensor ID in sensor calibration data file.

```
char sensor_id_str[SENSOR_ID_STRING_LENGTH];
```

```
int ret_val = sdkGetSensorIDFromCalData(sensor_id_str);
```

Parameters

sensor_id_str - char buffer to contain sensor ID [OUT]

Returns

Returns one numeric value of NSP_RETURN_VALUE_SUCCESS.

- the length of sensor ID string (>0).
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null
- cc_ec_memory_is_null

See also

sdkGetSensorIDFromDevice

sdkGetSensorIDFromDevice()

EXPORT_DLL int sdkGetSensorIDFromDevice (char * ***sensor_id_str***)

Get sensor ID of physical device(sensor)

This function returns a sensor ID of currently activated physical device(sensor). Current sensor is an activated one by sdkActivateSensorWithID() or sdkActivateSensorWithIndex() functions.

```
char sensor_id[SENSOR_ID_STRING_LENGTH];
int ret_value = sdkGetSensorIDFromDevice(sensor_id);
```

Parameters

pointer to char array for sensor ID data [OUT]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_memory_is_null
- cc_ec_device_object_is_null

See also

sdkGetSensorIDFromCalData

sdkGetShutterSpeed()

EXPORT_DLL int sdkGetShutterSpeed ( )

Get shutter speed.

This function returns a shutter speed value of currently activated sensor.

```
int cur_ss = 0;
```

```
cur_ss = sdkGetShutterSpeed();
```

Parameters

void

Returns

Returns one numeric value of:

- current shutter speed (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

sdkSetShutterSpeed

sdkGetShutterSpeedLimits()

EXPORT_DLL int sdkGetShutterSpeedLimits (int * ***min_limit***,
int * ***max_limit***
)

Get Shutter Speed Limist.

This function returns two shutter speed limits (minimum and maximum).

```
int ss_min, ss_max;
int ret_value = sdkGetShutterSpeedLimits(&ss_min, &ss_max);
```

Parameters

ss_min - int pointer to the minimum shutter speed (OUT)

ss_max - int pointer to the maximum shutter speed (OUT)

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)

See also

sdkGetShutterSpeed, sdkSetShutterSpeed, sdkShutterSpeedToExposureTime

sdkGetSpectrumLength()

EXPORT_DLL int sdkGetSpectrumLength ( )

Get size of spectrum data.

This function returns the size(length) of spectrum data.

```
int spectrum_length;
spectrum_length = sdkGetSpectrumLength();
```

Parameters

void

Returns

Returns one numeric value of:

- spectrum data size(length) (>0)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null

See also

sdkGetWavelengthInfo, sdkGetResolution, sdkCalculateSpectrum

sdkGetWavelengthInfo()

EXPORT_DLL int sdkGetWavelengthInfo (double * ***start_wavelength***,
double * ***end_wavelength***,
double * ***interval_wavelength***
)

Get wavelength information data.

This function returns wavelength range information for spectrum data.

```
double start_wavelength, end_wavelength, wavelength_interval;
int ret_val = sdkGetWavelengthInfo(&start_wavelength,
                                   &end_wavelength,
                                   &wavelength_interval);
```

Parameters

start_wavelength - double pointer to start wavelength [OUT]

end_wavelength - double pointer to end wavelength [OUT]

wavelength_interval - double pointer to wavelength interval[OUT]

Returns

Returns one numeric value of NSP_RETURN_VALUE_SUCCESS.

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_spectrum_core_object_is_null

See also

sdkGetResolution

sdkSetShutterSpeed()

EXPORT_DLL int sdkSetShutterSpeed (int ***shutter***)

Change current shutter speed of a device(sensor)

This function changes shutter speed of currently activated device(sensor).

```
int new_ss = 50;
int ret_value = sdkSetShutterSpeed(new_ss);
```

Parameters

shutter - shutter speed value [IN]

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)
- cc_ec_device_object_is_null

See also

duGetShutterSpeed

sdkShutterSpeedToExposureTime()

```
EXPORT_DLL int sdkShutterSpeedToExposureTime ( int      master_clock,
                                               int      shutter_speed_val,
                                               double * exposure_time
                                             )
```

Convert shutter speed to exposure time(msec)

This function converts a shutter speed value to exposure time (unit: msec).

```
int master_clock = 5; // 5 MHz
int cur_ss = 50;
double exposure_time = 0.0;
int ret_value = sdkShutterSpeedToExposureTime(master_clock, cur_ss,
&exposure_time );
```

Parameters

master_clock - master clock of MCU to sensor (IN)

cur_ss - current shutter speed (IN)

exposure_time - double pointer to exposure time (OUT)

Returns

Returns one numeric value of:

- NSP_RETURN_VALUE_SUCCESS (1)
- NSP_RETURN_VALUE_FAILURE (-1)

See also

sdkGetShutterSpeed, sdkSetShutterSpeed, sdkGetShutterSpeedLimits, sdkExposureTimeToShutterSpeed

Chapter 4 COS API Reference

COS API will soon be available to developers.
Please contact nanoLambda for COS API.

info@nanolambda.net