Spectral Products

Spectrometers · Spectrophotometers · Color Instruments · Spectrographs · Monochromators

SM32Pro SDK
Spectrometer Operating Software

USER MANUAL

(SM301/SM301EX)

Spectral Products

Spectrometers · Spectrophotometers · Color Instruments · Spectrographs · Monochromators

111 Highland Drive · Putnam, CT · 06260 · USA
PHONE (860) 928-5834· FAX (860) 928-2676
http://www.spectralproducts.com

Table Of Contents

Spectral Products

Warranty And Liability

This SM product is warranted against defects in material and workmanship for a period of one year from the date of shipment. During the warranty period, Spectral Products (SP) will, without charge, repair or replace, at its discretion, the defective product or component parts.

For warranty service or repair, this product must be returned to a service facility designated by SP. For products returned under warranty, the Buyer shall prepay shipping charges (including shipping charges, duties, and taxes for products returned to SP from another country), and SP will pay for shipping charges to return the product to the Buyer.

This warranty does not apply in the event of misuse or abuse of the product or as a result of unauthorized alterations, modifications or repairs, if the serial number is altered, defaced or removed, the improper or inadequate maintenance by the Buyer, Buyer-supplied software or interfacing, operation outside of the environmental specifications for the product, or improper site preparation or maintenance. No other warranty is expressed or implied.   SP shall not be liable for any consequential damages, including without limitation, damages resulting from loss of use, as permitted by law.

Quick Start Installation Guide

This quick start installation guide will walk you through the quick and easy installation of the SM software package you have chosen. Please follow the instructions closely to complete the installation of your product.

1. Insert the SM installation CD into the appropriate CD-Rom drive.

2. When the auto run starts, cancel the installation.

3. Explore the contents of the CD and copy the "\SDKs\SDK_SM301" folder to any folder or directory on your local drive.

Spectral Products

Spectrometers · Spectrophotometers · Color Instruments · Spectrographs · Monochromators

111 Highland Drive · Putnam, CT · 06260 · USA
PHONE (860) 928-5834· FAX (860) 928-2676
http://www.spectralproducts.com

System Requirements

Check that your computer meets the minimum requirements for the SM32Pro system.

Requirements for the Software

- Any IBM compatible computer
- A hard drive with at least 50 MB free space
- A CD-ROM
- A VGA or compatible display
- 32 MB RMA or higher
- A mouse or other pointing device
- Microsoft Windows® 98, 98SE, ME, NT, 2000, and XP

Check System Package Contents

Check that your SM system package contains all of the required components.

Common system packages contain the fallowing:

- Spectrometer
- A/D board (unless a different interface was requested)
- Cable and Adapter
- Any accessories ordered

Note: Package contents may vary from unit to unit and order to order. If you have any questions about the contents of your package, please contact the support team referred to in the back of this manual.

Spectral Products

Spectrometers · Spectrophotometers · Color Instruments · Spectrographs · Monochromators

Getting Started

Welcome to the Spectral Products **SM32Pro Software Development Kit**.

This kit documents a library of functions for accessing the USB Data Acquisition board used by the PbS/PbSe detector units, SM301 and SM301EX. Two fundamental detector imaging concepts are the Collection of Data and Generating Meaningful Values from that data.

Collection of Data

In our PbS/PbSe detector spectrometer, the light is dispersed across PbS/PbSe 256 pixel detector array. Data is collected by each pixel and converted to a relative value by the analog-to-digital (A/D) converter ranging from 0 to 65535 (16bit) and representing the intensity of the light at each pixel. We are able to control the amount of time that the pixels collect light and thus read signals of varying strengths. Using this library of functions you will be able to adjust the signal capture time (Integration Time) and collect the converted signals from each pixel. The criteria to adjust the integration is to try different lengths of time with a reference signal representing the maximum possible signal level during the measurement until the peak value across the range of pixels is close to, but not at 65535. We recommend trying to get the peak about 65000. This way you can be sure that you are getting an optimized measurement condition with no saturation of PbS/PbSe detector elements.

Generating Meaningful Values

The information you collect from the system is only the relative signal size at each detector pixel until it has been calibrated to absolute (certified reference) values. That is in the raw data, the pixel location represents spatial distribution of the PbS/PbSe detector elements and the corresponding intensity is represented by a digital value. Since we can adjust the strength of the light source and the integration time of the PbS/PbSe detector, this A/D value only provides a relative representation of the real world parameters.

We address both of these problems. We connect pixels to wavelengths by use of known emission lines (e. g. SP calibration light sources or narrow band

filters) across the range of pixels together with a table relating pixels to wavelengths in nanometers. Curve fitting functions can then be applied to generate a polynomial function for the conversion of all other pixels to wavelengths.

We address the second problem in one of two ways. Both ways involve "normalization." Normalization involves measurements of signal strength based on its ratio with respect to reference signal intensity. If we are only interested in the spectral distribution of the sample signal, then we can normalize a sample signal scan to one reference value by, for example, intensities of all the array elements divided by the peak intensity. If we want to measure percent transmission, the light can be measured first with only air in the light path as 100% reference and then the sample can be inserted into the light path and a sample scan followed. Consequently, the divisions of sample scan readings by the 100% reference yields the relative transmission values, or percent transmission when multiplied by 100%. For reflectance measurement similar practice can be applied and a high reflector can be used as a reference air in many cases.

Background Scan: It is strongly recommended that a measurement be taken prior to any sample scan, in which no external light but only the detector noise will be sensed and subsequently subtracted from all the following measurements. This will subtract thermally generated "background" DC offset level and statistically establish a baseline of zero for all subsequent measurements.

Spectral Products

Spectrometers · Spectrophotometers · Color Instruments · Spectrographs · Monochromators

Using the SP Libraries

For Visual C/C++, you must include the correct "include" file and "lib" file for the functions you are calling.

For Visual Basic, please include the correct "BAS" file. The function declarations listing in this file demonstrate the correct data types. To pass a "pointer" or array to a DLL from Visual basic, simply pass the first element of the array (IE SampleData[0]).

Please refer to the samples or contact us if you need any further technical assistance.

Procedure for using the .dll with VC++

1.   Copy "SPdbUSBIRCS.dll" from "\ SDKs\SDK_SM301" to your work or windows\system folder.

2.   Copy "SPdbUSBIRCS.lib" from "\ SDKs\SDK_SM301" to your work or release folder.

3.   Add this "SPdbUSBIRCS.lib" to the project/settings/link/object/library module.

General Overview

Some basic fundamentals of utilizing the SP SDK functions

All SP's DLL functions are based on the ADICLib.dll that the manufacture provided. To get more detailed information about all DLL functions in the ADICLib.dll, refer to the help file (ADICLib256Help.chm) for it.

First spTestBoard() should be called to get the appropriate information of the board. You can trace some basic setting values from this function. To turn on or off the TE cooler, run spSetTECoolerEx(). With spReadTemp(), you can check the current cooling temperature status. The spSetIntTimeEx() is for setting the integration time. You can set it from 10usec to ~220msec.

Integration time is the time period the detector pixels are exposed to light before the resulting charges are read out. A longer integration time can allow you to detect a lower light level signal. The longer your integration time is the more background signal will accumulate.

The PbS/PbSe detector should be balanced before getting the signals from the detector. spBalanceEx() is for the balance of the detector. Please be sure that this spBalanceEx() is for setting the stable base line. The input light should be turned off or blocked completely before running this function. Reading the signals from the detector is carried out by spReadDataEx().

By using the spStoreAllEx(), you can store all detector array operational data and settings currently resident in the PIC processor's run-time memory to the PIC processor's NVRAM for future retrieval to a known operating state after a mux controller board power cycle.

Using Curve Fitting to Calibrate SM32Pro

Curve fitting is used in SM32Pro to correlate the physical locations of pixels on the detector with the known wavelength of the radiation falling on them.

This is done by identifying the pixel locations where the maximal of the known wavelength is at. These peak intensity wavelengths and pixels are used

by spPolyCalc() and spPolyFit() to generate a correlating polynomial function which best represents all the data points. We have found that using a third order polynomial function produced the most desirable results for most cases. In cases where high dispersion elements are used, lower order polynomial functions may have to be utilized due to the limited known wavelengths available from the calibration lamps.

Calibration Files

Each unit's calibration set is included in the SM32Pro software settings. This text file contains calibration data of the form "DataX=Wavelength;Pixel". The file also contains the regression coefficients "A0 value", "A1 value", … , "B3 value" that satisfy the equations.

$$l_i = A_0 + A_1 P_i + A_2 P_i^2 + A_3 P_i^3$$
$$P_i = B_0 + B_1 l_i + B_2 l_i^2 + B_3 l_i^3.$$

The A values are the coefficients for conversions from a pixel number to a wavelength in "nm" by use of the above first third order polynomial function. The B values allow the conversions from a desired wavelength in "nm" to a pixel number.

Spectral Products

SDK Functions

Data Acquisition

spTestBoard

Used by the program to test the acquisition board.

spSetTECoolerEx

Used by the program to turn on or off the TE cooler.

spReadTemp

Used by the program to read the current TE cooler temperature status.

spSetIntTimeEx

Used by the program to set integration time.

spBalanceEx

Used by the program to balance the PbS/PbSe detector.

spReadDataEx

Used by the program to collect spectral data.

spStoreAllEx

Used by the program to store all operational data and settings.

spCloseEx

Used by the program to reset all board options upon resetting the software.

Calibration

spPolyFit

This function calculates the coefficients for a polynomial curve fitting function given an array of independent variables and a corresponding array of dependent variables.

Used by the program to generate a calibration function from pixels to wavelength.

spPolyCalc

This function calculates a polynomial function given the independent variable and a coefficient array.

Used for determining the wavelength for a given pixel location.

Procedure of getting data

spTestBoard

Test the detector board to see if it is working properly.

spSetTECoolerEx

Turn TE cooler on/off.

spSetIntTimeEx

Set integration time.

spBalanceEx

Balance the PbS/PbSe detector.

spReadDataEx

Read acquired data.

spStoreAllEx

Store all operational data and settings.

spCloseEx

Reset all the values to default values (should be called at the exiting of program).

Spectral Products

Alphabetical Function Reference

spTestBoard

long spTestBoard
(
long *lIntTime;   // Integration time
long *lPxlNum;   // Pixel number of the detector
long *lWellIdx;   // Charge well size index, 0: 1pF, 1: 4pF, 2: 7pF and 3:10pF
long lArrayMode;   // Array mode (0: Sens Array, 1: Cal Sensors)
)

This function is used to test the resources of the IO board. If the communication with the PbS/PbSe detector board works well, some basic operational data & settings will be retrieved from the EEPROM (NVRAM) on the board.

lIntTime is the integration time saved in the NVRAM of the detector board.

lPxlNum is the total pixel number saved in the NVRAM of the detector board. Usually it is 256.

lWellIdx is the index of the charge well size saved in the NVRAM of the detector board. It indicates the index of the assigned charge well size. Valid charge well sizes are 1pF (index: 0), 4pF (index: 1), 7pF (index: 2), 10pF (index: 3), 11pF (index: 4), 14pF (index: 5), 17pF (index: 6) and 20pF (index: 7).

The three arguments above will be retrieved when the spTestBoard() work with no problem. You can use the retrieved data.

lArrayMode is the array or mux order mode. The Cal Sensors' PbS/PbSe detector uses "array" order mode and the Sens Array's PbS/PbSe uses "mux" order mode. It can be "1" (for Cal Sensor's) and "0" (Sens Array's). Usually SP uses the Cal Sensor's PbS/PbSe detectors so if there is no special note for this value, just set it as "1".

RETURN:
If the board works successfully the function will return "1".
If the board doesn't work properly the function will return other number and error message.

spSetTECoolerEx

long spSetTECoolerEx

(

long const sTECoolerOn;   **// TE Cooler On/Off option. 1:On, 0:Off**

)

This function is used to turn on or off the TE cooler.

lTECoolerOn Informs whether the TE Cooler will be run or not. If this value is 1, then the TE cooler will be on. If 0, the cooler will be off.

RETURN:

If there is no problem on running, it returns 1. Otherwise, it returns other value and error message.

spReadTemp:

long spReadTemp

(

long *lTemp; **// TE cooler temperature status**

)

This function is used to read the TE cooler temperature status. This function has to be run only when the TE cooler is ON.

lTemp is the TE cooler temperature status returned. The returned value will be between -100 and 100. If the TE cooler is running properly, it is supposed to be ~0.

RETURN:

If there is no problem on running, it returns 1. Otherwise, it returns other value and error message.

spSetIntTimeEx:

long spSetIntTimeEx
(
 long const lIntegTime;   //new integration time (3-65535)
)

This function is used to change the integration time.

lIntegTime is the integration time. The real integration time to be set in the detector board is "lIntegTime"X10/3 microseconds. This number should range between 3 and 65535. The minimum integration time possible is 10usec (lIntegTime = 3) and the maximum is ~220msec (lIntegTime = 65535). To set 1msec, set the lIntegTime as "300".

RETURN:

If there is no problem on running, it returns 1. Otherwise, it returns other value and error message.

spBalanceEx:

long spBalanceEx
(
long const lTECoolerOn;   // TE Cooler On/Off (1/0)
long const lWellIndex;   // Charge well size index (0-3)
)

This function is used to balance the detector. The input light should be turned off or blocked completely before running this function.

lTECoolerOn Informs whether the TE Cooler will be run or not. If this value is 1, then the TE cooler will be on. If 0, the cooler will be off.

lWellIndex is the index of the charge well size saved in the NVRAM of the detector board. It indicates the index of the assigned charge well size. Valid charge well sizes are 1pF (index: 0), 4pF (index: 1), 7pF (index: 2), 10pF (index: 3), 11pF (index: 4), 14pF (index: 5), 17pF (index: 6) and 20pF (index: 7).

RETURN:

If there is no problem on running, it returns 1. Otherwise, it returns other value and error message.

Spectral Products

Spectrometers · Spectrophotometers · Color Instruments · Spectrographs · Monochromators

spReadDataEx

long spReadDataEx
(
long const lBit;   **// A/D conversion resolution. 16bit**
long lNumAve;   **// Number of data samples to average (1, 2, or 4)**
long *plPixArray;   **// The array in which spectral data is stored**
)

lBit is the A/D conversion resolution bit. The current ADIC board has 16bit.

lNumAve indicates the number of data samples to average prior to returning data to the calling procedure. Valid values are 1, 2, or 4 samples.

lpPixArray points to a read memory address with the long integers of the same pixel number as that of the detector. The detector is usually installed inversely in the SM301 spectrometer so the memory location has to be also reversed.

RETURN:
If there is no problem on running, it returns 1. Otherwise, it returns other value and error message.

spStoreAllEx:

long spStoreAllEx
(
)

This function is used to store all mux/detector array operational data & settings currently resident in the PIC processor's run-time memory to the PIC processor's NVRAM for future retrieval to a known operating state after a mux controller board power cycle.

RETURN:
If there is no problem on running, it returns 1. Otherwise, it returns other value and error message.

spCloseEx:

short spCloseEx
(
)

This function is called to reset all default valued for spTestBoard() upon closing of the application

RETURN:
If there is no problem on running, it returns 1. Otherwise, it returns other value and error message.

spPolyFit:

long spPolyFit
(
float *x;   **// Array of independent variables**
float *y;   **// Array of dependent variables**
short numPts;   **// Number of points in independent and dependent arrays**
float *coefs;   **// Pointer to array to hold calculated coefficients**
                                                // [index from 0 to order]
short order;   **// Order of polynomial**
)

This curve fitting function is used to find a polynomial function to calculate the wavelength of a given pixel.

This function is used for calibration purposes. Either a calibration light source or a series of narrow band filters are scanned and the pixel locations of all known peaks are identified along with the known wavelength at that peak. These peak locations and wavelengths are stored in the arrays x and y, respectively. The arrays indices should range from 0 to [Number_of_Points - 1]. They are passed to the function along with a requested order for the polynomial fitting function and an array large enough to hold the coefficients (). This array is then used with **spPolyCalc** to calculate wavelength from pixels.

x is an array containing the independent variables. It should range from 0 to (numPts-1).

y is an array containing the dependent variables. It should range from 0 to (numPts-1).

numPts is the number of points in the variable arrays

coefs is a pointer to the array that will contain the polynomial coefficients. It should range from 0 to (order-1).

order is the desired order of the polynomial. We have determined third order to be the optimum for wavelength calibration for most cases.

RETURN:
This function will return 1 if the function is successful. Otherwise it will return negative.

spPolyCalc

**void spPolyCalc
(
float *coefs,   // pointer to an array containing the polynomial coefficients
short order,   // the order of the polynomial equation
float x,// the pixel number
float *y   // the value to be calculated
)**

This function calculates for the following formula:

$y = a0 + a1*x^1 + a2*x^2 + ... + aN*x^N$, where * specified multiplication and ^ specifies "to the power of."

coefs is a pointer to an array containing the polynomial coefficients. These can be calculated using *spPolyFit*.

order specified the order of the polynomial equation and must be less than or equal to the number of elements in coefs.

x is the independent variable, in this case, the pixel number.

y is the value to be calculated.

RETURN:

None