STS Developer's Kit
Programming Guide

For Products: STS Developer's Kit
Document: DEV-RASPI-KIT-12-201508

A HALMA COMPANY

AMERICAS & WORLD HEADQUARTERS

Phone: +1 727-733-2447
Fax: +1 727-733-3962

Sales: info@oceanoptics.com
Orders: orders@oceanoptics.com
Support: techsupport@oceanoptics.com

830 Douglas Ave.
Dunedin, FL 34698
USA

Manufacturing & Logistics
4301 Metric Dr.
Winter Park, FL 32792
USA

EUROPE, MIDDLE EAST & AFRICA

Phone: +31 26-319-0500
Fax: +31 26-319-0505
Email: info@oceanoptics.eu

Germany : +49 711-341696-0
UK : +44 1865-811118
France : +33 442-386-588

Sales & Support
Geograaf 24
6921 EW Duiven
The Netherlands

Manufacturing & Logistics
Maybachstrasse 11
73760 Ostfildern
Germany

ASIA

Phone: +86 21-6295-6600
Fax: +86 21-6295-6708
Email: asiasales@oceanoptics.com

Japan & Korea: +82 10-8514-3797

Ocean Optics Asia
666 Gubei Road
Kirin Tower Suite 601B
Changning District
Shanghai
PRC, 200336

www.oceanoptics.com

Table of Contents

About This Manual

Document Purpose and Intended Audience

This document provides you with instructions to install and use the Application Programming Interface (API) for the STS Development Kit.

What's New in this Document

This version of the *STS Developer's Kit Programming Guide* updates information for Version 2.

Document Summary

Chapter	Description
Chapter 1: *Introduction*	Provides an overview of the STS Developer's Kit web API.
Chapter 2: *Examples*	Provides three example scripts.
Chapter 3: *Web Reference*	Contains a list of parameters.

Product-Related Documentation

- *STS Developer's Kit Installation and Operation Manual* – Information for installing and operating the STS Developer's Kit for single use with no need to make any alterations to the network capabilities of the device.
- *STS Developer's Kit Quick Start* – Information needed to get the STS Developer's Kit up and running
- *SeaBreeze* – Detailed instructions for the SeaBreeze software
- *STS Data Sheet* – Engineering-level information for the STS Spectrometer

You can access documentation for Ocean Optics products by visiting our website at http://www.oceanoptics.com. Select *Support→ Technical Documents*, then choose the appropriate document from the available drop-down lists.

ISO Certification

Ocean Optics, the industry leader in miniature photonics, has been certified for ISO 9001:2008 certification applicable to the design and manufacture of electro-optical equipment since 2009.

Chapter 1

Introduction

Overview

The STS Developer's Kit comes with a selection of easy to use web pages that are intended to provide commonly required functions for managing and controlling the spectrometer. These functions will not be sufficient for every application so a web Application Programming Interface (API) is available to users that will allow them to produce their own scripts to meet their own needs. The supplied web pages make use of the Web API and are a good source of examples for the use of the API.

The API can be used with any programming language that can communicate with the Raspberry Pi in the STS Developer's Kit via Hypertext Transfer Protocol (HTTP). Most of the code samples in the STS Development Kit documentation use PHP, and the API itself is written in PHP.

Software Architecture

The web API is provided by a set of PHP scripts that run in the context of the web server on the Raspberry Pi. These scripts communicate with a daemon service running on the Raspberry Pi which forwards commands and queries to the spectrometer using the SeaBreeze library and returns responses to the scripts.

Version 2 of the STS Developer's Kit allows more than one spectrometer to be connected to the Raspberry Pi. Most of the functions documented here allow a "channel" number to be specified that is associated with each spectrometer, i.e., if there are two connected spectrometers their channel numbers will be 0 and 1. This means that queries and commands can be directed to a specific spectrometer.

All of the functions in the API described here are accessible as a **URL http://{IP address}/cgi-bin/{function.php}.**

Where:

"{IP address}" = the current IP address of the Raspberry Pi, and

"{function.php}" = the appropriate script name documented later in this manual.

Many of the functions have set and get functionality for the parameters that control the spectrometer or the sequence of acquisition. In each of these cases the function name takes the form set{parameter} or get{parameter}

Where:

"{parameter}"= the parameter name

Functions that set a parameter return an integer value that indicates the result of the set operation:

1 indicates the operation has succeeded.

A value greater than 1 indicates failure (with an error message using the script getcurrentstatus.php). Functions that get a parameter or a result return the response as a string (where the result is an array e.g. wavelengths or a spectrum, the values are whitespace separated).

Most of the functions are provided in two forms:

1. A PHP function intended to be included in other PHP scripts, taking zero or more parameters depending on the specific operation being performed. These functions return either the success or failure result or the parameter/result being requested as a string.
2. A PHP script taking an argument on the URL that is passed to the corresponding PHP function using a PHP $_GET variable. The result is output with the PHP echo command. These scripts can be used directly and the results seen in e.g. a web browser.

The naming convention for these two types of script are as follows:

- type 1 begins with an underscore
- type 2 has the same name without the leading underscore

For example, the related functions to get the spectrometer serial number are _getserial.php and getserial.php.

Chapter 2

Examples

Overview

Three worked examples are included in this chapter, starting with a very simple example of setting basic spectrometer acquisition parameters and building up to customizing the scripts on the Raspberry Pi itself to provide a simple spectral peak finding application.

Example 1: A Simple Web Page to Configure and Start a Short Sequence of Acquisitions

This example shows a simple HTML page that can be created and stored on the computer to be connected to the Raspberry Pi:

1. Sets the integration time of the spectrometer, sets scans to average to 10, and turns off boxcar smoothing.
2. Configures a sequence of 5 acquisitions with 60 seconds between each acquisition.
3. Starts the sequence with a button.

Setting the maximum number of acquisitions terminates the sequence. The results will be saved on to the SD card on the Raspberry Pi to be retrieved later using the web pages described in the *STS Developer's Kit Installation & Operation Manual* (see *Product-Related Documentation*).

► *Procedure*

1. Create a directory on a Windows computer called C:\Raspberry.
2. Using notepad as the text editor, create a file simple.html in this directory.
3. Copy and paste the text shown below in Figure 1 into the file.

The web page uses several of the scripts described in this manual. To keep the web page simple, HTML iframes is used to load (execute) each script from this web page. To run the scripts, load the web page into your browser by typing "C:\Raspberry\simple.html" in the browser address bar (or just double-click on the file). The IP address of the Raspberry Pi is assumed to be the default 192.168.42.1.

```
<html>

<p>
We are setting up a sequence of acquisitions with an integration time of 500ms, 10
scans to average and
no boxcar smoothing.
</p>

<p>
There will be 5 spectra with a 60 second interval between acquisitions. The spectra
will be saved into the directory /home/ocean on the SD card with one spectra per
file. Each file will begin with the prefix EXAMPLE
followed by the numeric position in the sequence.
</p>

<p>
We will retrieve the results at a later time using the "Manage Results" page on the
Raspberry Pi.
</p>

<p>
Implementation detail: we must use several of our scripts to do all of this. The
quick way of doing this
in a single web page is to use iframes. The scripts will run sequentially - we
don't care what order
they are executed in, only that they all complete before we start the sequence of
acquisitions with
the submit button below.
</p>

<iframe src='http://192.168.42.1/cgi-bin/setintegration.php?time=500000'></iframe>
<iframe src='http://192.168.42.1/cgi-bin/setaverage.php?scans=10'></iframe>
<iframe src='http://192.168.42.1/cgi-bin/setboxcar.php?width=0'></iframe>
<iframe src='http://192.168.42.1/cgi-bin/setprefix.php?prefix=EXAMPLE'></iframe>
<iframe src='http://192.168.42.1/cgi-bin/setmaxacquisitions.php?acquisitions=5'>-
</iframe>
<iframe src='http://192.168.42.1/cgi-
bin/setsavelocation.php?location=/home/ocean'></iframe>
<iframe src='http://192.168.42.1/cgi-bin/setsavemode.php?savemode=multi'></iframe>
<iframe src='http://192.168.42.1/cgi-
bin/setsequenceinterval.php?interval=60000'></iframe>

<form action='http://192.168.42.1/cgi-bin/startsequence.php'>

This form has a single push button to start the sequence of acquisitions.

<input type=submit value='Push to start'>

</form>

</html>
```

Figure 1: Simple Example Web Page

To run the scripts make sure you are connected to the Raspberry Pi wifi network and load the web page into your web browser. Note that each iframe is displaying the result “1” – this is the result returned by the scripts to indicate successful completion. Start the sequence of acquisitions by pressing the “Push to start” button. A single character, “1”, should appear in your browser, indicating a successful result from the script. You can look for the results as they begin to appear using the Manage Results web page provided on the Raspberry Pi and download them upon completion.

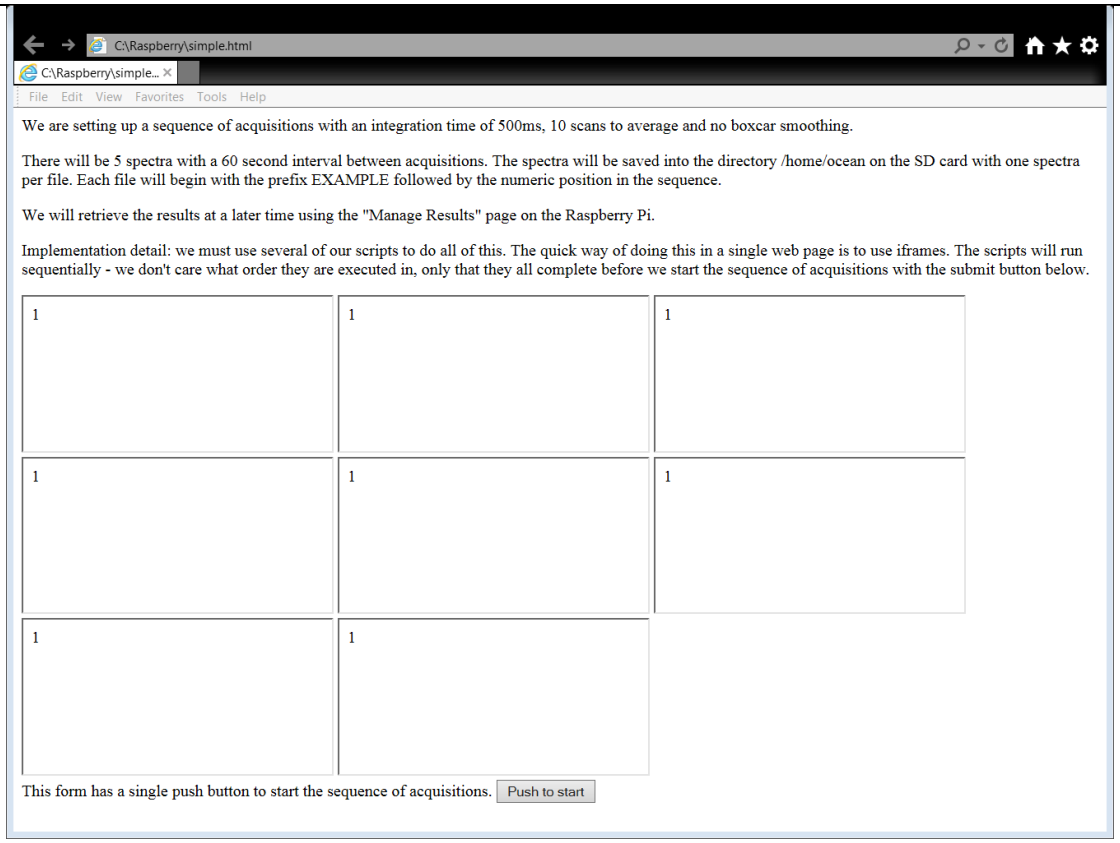

Figure 2: Web page showing successful configuration results

Example 2: Script to Acquire a Spectrum at Regular Intervals, Saving Spectra with Maximum Intensity over a Threshold

This example produces a script called maximum.php containing the text shown in Figure 3 on a Windows computer to acquire a spectrum at regular intervals and save spectra that contain a maximum intensity over a fixed threshold into a local directory C:\Raspberry. PHP is used for this script but any one of a number of other languages can be used to produce the same application. This example assumes that PHP has been installed on the Windows computer, the directory C:\Raspberry exists, that we have connected to the Raspberry Pi WIFI, and the Raspberry Pi has the default IP address. Testing for errors has been restricted to the results of the Web API scripts only for the sake of illustration.
There are several points to note about this script:

- The PHP function file_get_contents is used to run each of the scripts on the Raspberry Pi. This function opens the specified URL and returns the result as a string.
- The result of each script is tested. Most of the scripts return the value “1” as a result, indicating successful completion.

- If the result of a script is unsuccessful, a meaningful error message can be retrieved using the getcurrentstatus.php script. This script returns a string containing the error message resulting from the last executed script.
- The script getspectrum.php returns the spectrum as a space separated list of pixel values in a string. This is easily converted into an array with the PHP explode function.
- Since getspectrum.php returns the spectrum, not a result code like most other scripts, the status message for the result “Success” must be tested to determine if any error occurred.
- The script getspectrum.php does not return a result until the spectrometer has completed the acquisition. This must be taken into consideration if you have a long integration time and/or a large number of scans to average.

```
<?php

// Set the threshold we are interested in to 5000 counts
$threshold = 5000.0;

// set the interval we will leave between acquisition to 10 seconds
$interval = 10;

// Set the maximum number of spectra we will acquire to 1000
$numAcquisitions = 1000;

// Now set up the spectrometer: integration time 600ms, no boxcar, no scans to
average
// On each step we check for an error and print the error message if something went
wrong
$result = file_get_contents("http://192.168.42.1/cgi-
bin/setintegration.php?time=600000");
if ($result != "1") {
    $message = file_get_contents("http://192.168.4.2.1/cgi-
bin/getcurrentstatus.php");
    echo "An error occurred: " . $message . "\n";
}

$result = file_get_contents("http://192.168.42.1/cgi-bin/setboxcar.php?width=0");
if ($result != "1") {
    $message = file_get_contents("http://192.168.4.2.1/cgi-
bin/getcurrentstatus.php");
    echo "An error occurred: " . $message . "\n";
}

$result = file_get_contents("http://192.168.42.1/cgi-bin/setaverage.php?scans=0");
if ($result != "1") {
    $message = file_get_contents("http://192.168.4.2.1/cgi-
bin/getcurrentstatus.php");
    echo "An error occurred: " . $message . "\n";
}

// finally begin the sequence
for ($acq = 0; $acq < $numAcquisitions; $acq++) {

    // Acquire a spectrum, returned as a space separated list of pixels.
    // Check for errors and report if any. Note getspectrum.php returns the
spectrum not
    // an error code so we must test the current status for "Success" instead this
time.
    $result = file_get_contents("http://192.168.42.1/cgi-bin/getspectrum.php");
    $message = file_get_contents("http://192.168.42.1/cgi-
bin/getcurrentstatus.php");
    if ($message == "Success") {
        // Convert the string of counts to an array.
        $pixels = explode(" ", $result);

        // Now test for the threshold value and save the spectrum to a local file
if
        // the threshold is exceeded
        if (max($pixels) > $threshold) {
            $filename = "c:\\Raspberry\\" . $acq . ".txt";
            $ok = file_put_contents($filename, $result);
        }
    }
    else {
        echo "An error occurred: " . $message . "\n";
    }
```

```
    // finally pause for the specified time before the next acquisition
    sleep($interval);
}
?>
```

Figure 3: Script to save spectra with maxima over a threshold

Run the script from the Windows command prompt with the command "php maximum.php". The results of the acquisitions should appear in the directory C:\Raspberry.

Example 3: Customized Scripts on the Raspberry Pi: a Peak Finding Script

This example shows how the scripts on the Raspberry Pi can be extended. It assumes that you have some familiarity with logging on to a Linux computer, navigating the directory structure and using text editors such as vi or nano. You can now take advantage of the ability to "include" some of the scripts containing functions on the Raspberry Pi. For this example, it is assumed that you are connected to the Raspberry Pi on the default IP address and have logged in as user "root" using e.g. PuTTY.

This example sets the spectrometer parameters, acquires a spectrum, and then presents a table of peaks over a threshold value versus wavelength as a web page. Go to the directory /var/www/cgi-bin and create a file peaks.php in the chosen editor, e.g., vi, containing the text shown in Figure 4 below.

Notes on this example:

- Error checking is not used in this example to keep the code short and clear. This is not a robust implementation of peak detection; it is only intended as an example to illustrate what can be achieved.
- You can now include some PHP functions available on the Raspberry Pi directly into your script. Doing this enables you to avoid writing lengthy URLs in your code.
- A boxcar smoothing value of 3 is used in this example to try to smooth out some of the noise. This is to reduce the chances of identifying noise spikes as peaks.
- The algorithm estimates the gradient of the spectrum and looks for zero crossings of the gradient. The gradient is estimated using forward differences for simplicity rather than accuracy.
- Wherever zero crossings are found in the gradient, the corresponding intensity in the spectrum is checked to see if it is above the desired threshold and added to the HTML table if so. Troughs above the threshold can also be misidentified by this algorithm. The checking necessary to avoid this is omitted to keep the example short.

```
<html>
<?php
// include all of the functions we will be using
include '_getwavelengths.php';
include '_getspectrum.php';
include '_setintegration.php';
include '_setaverage.php';
include '_setboxcar.php';

// set the basic spectrometer parameters and the threshold we are setting
$result = setintegration(200000);
$result = setboxcar(3);
$result = setaverage(0);
$threshold = 2000.0;

// get the wavelengths and acquire the spectrum
$wl = getwavelengths();
$spec = getspectrum();

// convert the wavelengths and intensities to arrays
$wavelengths = explode(" ", $wl);
$spectrum = explode(" ", $spec);
$numPixels = count($wavelengths);

// Now compute the gradient of the spectrum by simple forward differences
$gradient = array();
for ($i = 0; $i < $numPixels-1; $i++) {
       $forwardDiff = ($spectrum[$i+1] - $spectrum[$i]) / ($wavelengths[$i+1] -
$wavelengths[$i]);
       array_push($gradient, $forwardDiff);
}
?>
A table of peaks above the threshold <?php echo $threshold; ?>
<table>
<tr>
<th>Wavelength (nm)</th>
<th>Intensity</th>
</tr>
<?php
for ($i = 0; $i < $numPixels-2; $i++) {
       // Look for zero crossing of the gradient to identify a peak
       if ($gradient[$i+1] * $gradient[$i] <= 0.0) {
              // then check if the corresponding peak is over the threshold
              if ($spectrum[$i+1] > $threshold) {
                     // add the peak to the table
                     echo "<tr><td>" . $wavelengths[$i+1] . "</td><td>" .
$spectrum[$i+1] . "</td></tr>";
              }
       }
}
?>
</table>
</html>
```

Figure 4: Customized Peak Finding Script on the Raspberry Pi

If you are connected to the Raspberry Pi wireless network with the SD card loaded, you can access the scripts at the URL **http://192.168.42.1/cgi-bin/peaks.php**. Example output from the script is shown below in Figure 5.

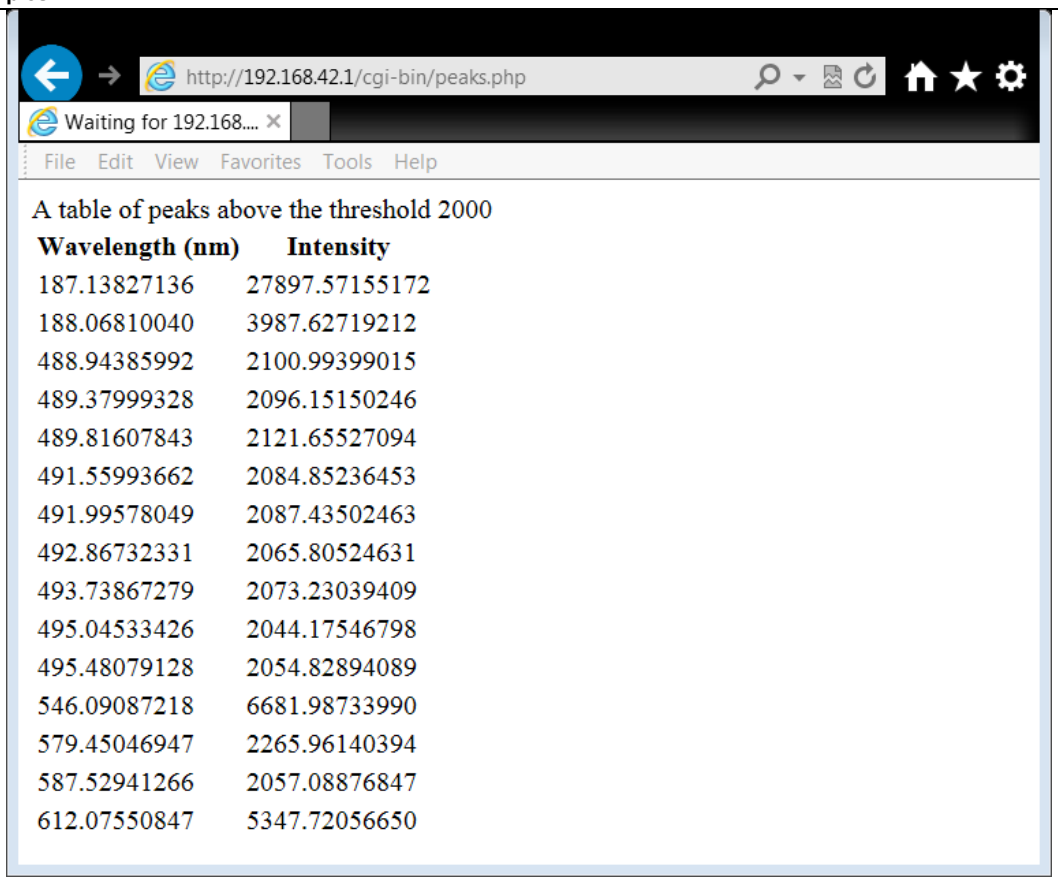

Figure 5: Example Output from peaks.php

Chapter 3

Web Reference

Overview

The web reference is organized into the following sections:

- *Spectrometer Parameters* -- functions that query spectrometer parameters such as minimum integration time and maximum intensity.
- *Acquisition Parameters* -- the control and query of each individual acquisition e.g. integration time and boxcar smoothing.
- *Acquisition Sequence Parameters* -- functions that configure and query a sequence of acquisitions e.g. interval between acquisitions.
- *Sequence Control* -- starting, stopping and pausing of a sequence.
- *Miscellaneous* -- Other functions not covered previously such as query the software version.

Note

In the following sections, parameters shown in { } are optional.

Spectrometer Parameters

Get Minimum Integration Time

Return the minimum integration time for the spectrometer in microseconds.

Script: `_getminintegration.php`

Function Specification:

`function getminintegration($channel = 0)`

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `getminintegration.php`

`Parameters:`

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

`http://192.168.42.1/cgi-bin/getminintegration.php{?channel=0}`

Get Maximum Integration Time

Return the maximum integration time in microseconds.
Script: `_getmaxintegration.php`
Function Specification:
`function getmaxintegration($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `getmaxintegration.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/getmaxintegration.php{?channel=0}`

Get Maximum Intensity

Return the maximum intensity (counts) for the spectrometer.
Script: `_getmaxintensity.php`
Function Specification:
`function getmaxintensity($channel = 0`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `getmaxintensity.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/getmaxintensity.php{?channel=0}`

Get Spectrometer Name (Type)

Return the spectrometer name.
Script: `_getname.php`
Function Specification:
`function getname($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `getname.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/getname.php{?channel=0}`

Get Spectrometer Serial Number

Return the spectrometer serial number.
Script: `_getserial.php`
Function Specification:
`function getserial($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `getserial.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/getserial.php{?channel=0}`

Get Spectrometer Wavelengths

Return the wavelengths in nm for each pixel in the spectrometer. This is returned as a space-separated array of real numbers.
Script: `_getwavelengths.php`
Function Specification:
`function getwavelengths($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `getwavelengths.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/getwavelengths.php{?channel=0}`

Acquisition Parameters

Get Scans to Average

Return the number of scans averaged to produce the resulting spectrum.
Script: `_getaverage.php`
Function Specification:
`function getaverage($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: getaverage.php
Parameters:
channel – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
http://192.168.42.1/cgi-bin/getaverage.php{?channel=0}

Set Scans to Average

Set the number of scans to average to produce the resulting spectrum.
Script: _setaverage.php
Function Specification:
function setaverage($scans, $channel = 0)
Parameters:
$scans – On entry, an integer specifying the number of scans to average. If $scans is zero averaging is turned off. This value must be greater or equal to zero.
$channel – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: setaverage.php
Parameters:
scans – An integer specifying the number of scans to average. If scans is zero averaging is turned off. This value must be greater or equal to zero.
channel – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
http://192.168.42.1/cgi-bin/setaverage.php?scans=4{&channel=0}

Get Pixel Binning Factor

Return the pixel binning factor (if the spectrometer supports pixel binning). When pixel binning is enabled, a number of adjacent pixels are summed depending on the value of the binning factor. A binning factor of zero means no binning is applied. A binning factor of 1 means pairs of adjacent pixels in the spectra are summed, effectively halving the number of pixels returned. A binning factor of 2 means that each 4 adjacent pixels are summed, returning a quarter of the pixels. Similarly a binning factor of 3 means each 8 adjacent pixels are summed and an eighth of the number of pixels returned.
Script: _getbinning.php
Function Specification:
function getbinning($channel = 0)
Parameters:
$channel – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: getbinning.php
Parameters:

channel – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

http://192.168.42.1/cgi-bin/getbinning.php{?channel=0}

Set Pixel Binning Factor

Set the pixel binning factor (if the spectrometer supports pixel binning). When pixel binning is enabled, a number of adjacent pixels are summed depending on the value of the binning factor. A binning factor of zero means no binning is applied. A binning factor of 1 means pairs of adjacent pixels in the spectra are summed, effectively halving the number of pixels returned. A binning factor of 2 means that each 4 adjacent pixels are summed, returning a quarter of the pixels. Similarly a binning factor of 3 means each 8 adjacent pixels are summed and an eighth of the number of pixels returned.

Script: _setbinning.php

Function Specification:

```
function setbinning($bin, $channel = 0)
```

Parameters:

$bin – On entry, an integer specifying the pixel binning factor. This value must be in the range 0, 1, 2, 3.

$channel – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: setbinning.php

Parameters:

bin – An integer specifying the pixel binning factor. This value must be in the range 0, 1, 2, 3.

channel – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

http://192.168.42.1/cgi-bin/setbinning.php?bin=2{&channel=0}

Get Boxcar Smoothing Width

Return the boxcar smoothing width. When boxcar smoothing is turned on each pixel is replaced with the average of its own value and *n* pixels either side where *n* is the boxcar width. If the width is set to zero then boxcar smoothing is turned off.

Script: _getboxcar.php

Function Specification:

```
function getboxcar($channel = 0)
```

Parameters:

$channel – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: getboxcar.php

Parameters:

channel – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

http://192.168.42.1/cgi-bin/getboxcar.php{?channel=0}

Set Boxcar Smoothing Width

Set the boxcar smoothing width. When boxcar smoothing is turned on each pixel is replaced with the average of its own value and *n* pixels either side where *n* is the boxcar width. If the width is set to zero then boxcar smoothing is turned off.

Script: _setboxcar.php

Function Specification:

function setboxcar($width, $channel = 0)

Parameters:

$width – On entry, an integer specifying the boxcar smoothing width. If this value is zero then boxcar smoothing is turned off. This value must be >= 0.

$channel – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: setboxcar.php

Parameters:

width – An integer specifying the boxcar smoothing width. If this value is zero then boxcar smoothing is turned off. This value must be >= 0.

channel – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

http://192.168.42.1/cgi-bin/setboxcar.php?width=2{&channel=0}

Get Electric Dark Correction On/Off

Return the electric dark correction on/off status for spectrometers that support this feature. Electric dark correction automatically subtracts the average of several optically masked pixels from a spectrum to compensate for noise.

Script: _getedcorrect.php

Function Specification:

function getedcorrect($channel = 0)

Parameters:

$channel – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: getedcorrect.php

Parameters:

channel – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

http://192.168.42.1/cgi-bin/getedcorrect.php{?channel=0}

Set Electric Dark Correction On/Off

Set the electric dark correction on/off status for spectrometers that support this feature. Electric dark correction automatically subtracts the average of several optically masked pixels from a spectrum to compensate for noise.

Script: `_setedcorrect.php`

Function Specification:

```
function setedcorrect($electric, $channel = 0)
```

Parameters:

`$electric` – On entry an integer 0 or 1. A value of 0 turns off electric dark correction. A value of 1 turns on electric dark correction.

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `setedcorrect.php`

Parameters:

`electric` – An integer 0 or 1. A value of 0 turns off electric dark correction. A value of 1 turns on electric dark correction.

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-
bin/setedcorrect.php?electric=1{&channel=0}
```

Get Integration Time

Return the current integration time for the spectrometer in microseconds.

Script: `_getintegration.php`

Function Specification:

```
function getintegration($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `getintegration.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-bin/getintegration.php{?channel=0}
```

Set Integration Time

Set the integration time of the spectrometer in microseconds.

Script: `_setintegration.php`

Function Specification:

```
function setintegration($time, $channel = 0)
```

Parameters:

`$time` – On entry, an integer specifying the integration time for the spectrometer in microseconds. This value must be in the valid range for the spectrometer i.e. between the minimum and maximum integration time for the spectrometer.

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `setintegration.php`

Parameters:

`time` - An integer specifying the integration time for the spectrometer in microseconds. This value must be in the valid range for the spectrometer i.e. between the minimum and maximum integration time for the spectrometer.

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-
bin/setintegration.php?time=1000000{&channel=0}
```

Acquisition Sequence Parameters

Get Current Spectrum in a Sequence

Return the last acquired spectrum in a sequence of acquisitions. If no sequence has been started then this function returns zero for every pixel.

Script: `_currentspectrum.php`

Function Specification:

```
function currentspectrum($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `currentspectrum.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-bin/currentspectrum.php{?channel=0}
```

Get a Spectrum

Acquire and return a spectrum. This function will only return on completion of an acquisition so if the integration time is set to a large value and/or the number of scans to average is set to a large value then this can take some time.

Script: `_getspectrum.php`

Function Specification:

```
function getspectrum($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `getspectrum.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

`http://192.168.42.1/cgi-bin/getspectrum.php{?channel=0}`

Sequence Control

Get the Current Error Status

Return the error status of the last command as a human readable string. The error status is reset by calling any other function so if an error is detected this function must be used to determine the cause before any other function is called.

Script: `_getcurrentstatus.php`

Function Specification:

```
function getcurrentstatus($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `getcurrentstatus.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

`http://192.168.42.1/cgi-bin/getcurrentstatus.php{?channel=0}`

Get Maximum Acquisitions Limit

Return the current maximum number of acquisitions for a sequence of acquisitions. A sequence of acquisitions will stop automatically when it reaches this value to prevent the SD card from being filled accidentally. A value of zero means this limit is turned off and a sequence of acquisitions will continue until it is deliberately stopped or the SD card runs out of space.

Script: `_getmaxacquisitions.php`

Function Specification:

```
function getmaxacquisitions($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `getmaxacquisitions.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

`http://192.168.42.1/cgi-bin/getmaxacquisitions.php{?channel=0}`

Get the Saved File Prefix

Return the current prefix for the saved files. The sequence of spectra saved onto the SD card will have this string prepended to their names, allowing multiple sequences to be distinguished.

Script: `_getprefix.php`

Function Specification:

```
function getprefix($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `getprefix.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

`http://192.168.42.1/cgi-bin/getprefix.php{?channel=0}`

Get the Saved Spectra Location

Return the current path to the location that spectra will be saved into.

Script: `_getsavelocation.php`

Function Specification:

```
function getsavelocation($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `getsavelocation.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

`http://192.168.42.1/cgi-bin/getsavelocation.php{?channel=0}`

Get the File Save Mode

Return the file save mode. If the sequence is set to save to multiple files this returns the string "multi". If the sequence is set to save to a single file this returns the string "single".

Script: `_getsavemode.php`

Function Specification:
`function getsavemode($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `getsavemode.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/getsavemode.php{?channel=0}`

Get the Scope Mode Refresh Interval

Return the current interval in seconds between refreshes of the scope mode page.
Script: `_getscopeinterval.php`
Function Specification:
`function getscopeinterval($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `getscopeinterval.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/getscopeinterval.php{?channel=0}`

Get the Scope Mode On/Off

Return the string “on” if scope mode is turned on or the string “off” if it is turned off.
Script: `_getscopemode.php`
Function Specification:
`function getscopemode($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `getscopemode.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/getscopemode.php{?channel=0}`

Get the Interval Between Saved Acquisitions

Return the current time interval in milliseconds between each acquisition and save of a spectrum in a sequence.

Script: `_getsequenceinterval.php`

Function Specification:

```
function getsequenceinterval($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `getsequenceinterval.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-
bin/getsequenceinterval.php{?channel=0}
```

Get the Acquisition Sequence State

Return a string specifying the state of the current acquisition sequence:

1. "not_yet_configured" if no sequence parameters, e.g., sequence interval have been set.
2. "not_yet_started" if the sequence has been configured but is not yet started.
3. "paused" if the sequence has been paused.
4. "active" if the sequence is currently running.

Script: `_getsequencestate.php`

Function Specification:

```
function getsequencestate($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `getsequencestate.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-bin/getsequencestate.php{?channel=0}
```

Pause a Running Sequence

Pause a currently running sequence to allow an ad-hoc measurement e.g. a reference spectrum to be taken.

Script: `_pausesequence.php`

Function Specification:
`function pausesequence($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `pausesequence.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/pausesequence.php{?channel=0}`

Resume a Paused Sequence

Resume a sequence of acquisitions that has been paused.
Script: `_resumesequence.php`
Function Specification:
`function resumesequence($channel = 0)`
Parameters:
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.
Script: `resumesequence.php`
Parameters:
`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.
Example URL:
`http://192.168.42.1/cgi-bin/resumesequence.php{?channel=0}`

Save a Spectrum

Acquire and save a single spectrum to the current save location.
Script: `_savespectrum.php`
Function Specification:
`function savespectrum($location, $channel = 0)`
Parameters:
`$location` – On entry, a valid (writeable) directory on the Raspberry Pi.
`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Set the Maximum Number of Acquisitions

Set the upper limit on the number of acquisitions in a sequence. If this value is set to zero then no upper limit is set and the sequence can continue until it is deliberately stopped or the SD card runs out of space.
Script: `_setmaxacquisitions.php`

Function Specification:

```
function setmaxacquisitions($acquisitions, $channel = 0)
```

Parameters:

`$acquisitions` – On entry an integer >= 0 specifying the maximum number of acquisitions in a sequence. If this is set to zero then no upper limit is applied.

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `setmaxacquisitions.php`

Parameters:

`acquisitions` – An integer >= 0 specifying the maximum number of acquisitions in a sequence. If this is set to zero then no upper limit is applied.

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-
bin/setmaxacquisitions.php?acquisitions=1000{?channel=0}
```

Set the Saved File Prefix

Set the current prefix for the saved files. The sequence of spectra saved onto the SD card will have this string prepended to their names, allowing multiple sequences to be distinguished.

Script: `_setprefix.php`

Function Specification:

```
function setprefix($prefix, $channel = 0)
```

Parameters:

`$prefix` – On entry a string that will be prepended to the names of the sequence of saved files. This must contain only characters that are valid within Linux filenames.

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `setprefix.php`

Parameters:

`prefix` – a string that will be prepended to the names of the sequence of saved files. This must contain only characters that are valid within Linux filenames.

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-bin/setprefix.php?prefix=OO-
TEST{&channel=0}
```

Set the Saved File Location

Set the current path to the location that spectra will be saved into. This must be a valid, writeable directory on the Raspberry Pi.

Script: `_setsavelocation.php`

Function Specification:

```
function setsavelocation($location, $channel = 0)
```

Parameters:

`$location` – On entry, a valid writeable directory on the Raspberry Pi.

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `setsavelocation.php`

Parameters:

`location` - A valid writeable directory on the Raspberry Pi.

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-
bin/setsavelocation.php?location=/home/ocean{&channel=0}
```

Set the Save Mode

Set the file save mode. Set the parameter to "multi" if the sequence is to save to multiple files. Set the parameter to "single" if the sequence is to save to a single file.

Script: `_setsavemode.php`

Function Specification:

```
function setsavemode($mode, $channel = 0)
```

Parameters:

`$mode` – On entry, a string set to "multi" if the sequence should be saved to one file per spectrum or "single" if each spectrum in the sequence should be appended to a single file.

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `setsavemode.php`

Parameters:

`savemode` - A string set to "multi" if the sequence should be saved to one file per spectrum or "single" if each spectrum in the sequence should be appended to a single file.

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-
bin/setsavemode.php?savemode=multi{&channel=0}
```

Set the Scope Interval

Set the current interval in seconds between refreshes of the scope mode page.

Script: `_setscopeinterval.php`

Function Specification:

```
function setscopeinterval($interval, $channel = 0)
```

Parameters:

`$interval` – On entry, an integer specifying the interval in seconds between refreshes of the scope mode page. This value must be > 0.

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `setscopeinterval.php`

Parameters:

`interval` – An integer specifying the interval in seconds between refreshes of the scope mode page. This value must be > 0.

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-
bin/setscopeinterval.php?interval=10{&channel=0}
```

Set the Scope Mode On/Off

Set the scope mode on or off. Set the parameter to the string "on" if scope mode is to be turned on or the string "off" if it is to be turned off.

Script: `_setscopemode.php`

Function Specification:

```
function setscopemode($mode, $channel = 0)
```

Parameters:

`$mode` – On entry, set this to "on" if scope mode should b turned on or "off" if scope mode should be turned off.

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `setscopemode.php`

Parameters:

`mode` – Set this to "on" if scope mode should b turned on or "off" if scope mode should be turned off.

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-
bin/setscopemode.php?mode=on{&channel=0}
```

Set the Acquisition Interval

Set the current time interval in milliseconds between each acquisition and save of a spectrum in a sequence. This value must be > 0 but in practice there is a minimum value due to the time taken to save the results to the SD card. The acquisition interval must be greater than the total acquisition time i.e. the integration time multiplied by the number of scans to average (or only the integration time if scans to average is turned off).

Script: `_setsequenceinterval.php`

Function Specification:

```
function setsequenceinterval($interval, $channel = 0)
```

Parameters:

`$interval` – On entry, an integer > 0 specifying the interval in milliseconds between each acquisition and save.

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `setsequenceinterval.php`

Parameters:

`interval` – An integer > 0 specifying the interval in milliseconds between each acquisition and save.

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-
bin/setsequenceinterval.php?interval=500{&channel=0}
```

Start a Sequence of Acquisitions

Start a sequence of acquisitions and saves with the current values of spectrometer and acquisition parameters.

Script: `_startsequence.php`

Function Specification:

```
function startsequence($channel = 0)
```

Parameters:

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `startsequence.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-bin/startsequence.php{?channel=0}
```

Stop a Sequence of Acquisitions

Stop a running sequence of acquisitions and saves.

Script: `_stopsequence.php`

Function Specification:

```
function stopsequence($channel = 0)
```

`$channel` – On entry, an integer specifying the channel number (spectrometer) to query. If this parameter is omitted the default value of 0 is used.

Script: `stopsequence.php`

Parameters:

`channel` – An integer specifying the channel number (spectrometer) to query. This parameter is optional. If it is omitted, channel number 0 is queried.

Example URL:

```
http://192.168.42.1/cgi-bin/stopsequence.php{?channel=0}
```

Miscellaneous

Get the Daemon Software Version

Return the version of the daemon software currently installed.
Script: _getversion.php
Function Specification:
function getversion()

Script: getversion.php
Example URL:
http://192.168.42.1/cgi-bin/getversion.php

Index