User Manual

USB Digital Step Attenuator

Version 1.0

Nov. 26, 2018

5-3, Beolmal-ro 551beon-gil, Gyeyang-gu, Incheon, 21009, Korea

sales@ga-technologies.com

www.ga-technologies.com

Table of contents

1. Introduction

1.1. Overview

GA Technologies GDSA series is single channel RF Digital Step Attenuator over 9KHz to 6GHz frequency range. The attenuator provides 0 to 95.25dB wide range of attenuation in 0.25dB step.

This manual describes how to setup and use software and hardware of GDSA series Digital Step attenuator.

1.2. Included Accessories

GDSA-069 models are supplied with:
1) USB Cable 1m, A to Mini B male
2) USB flash drive containing GUI, API shared library, sample code and manual

2. Installation and setup

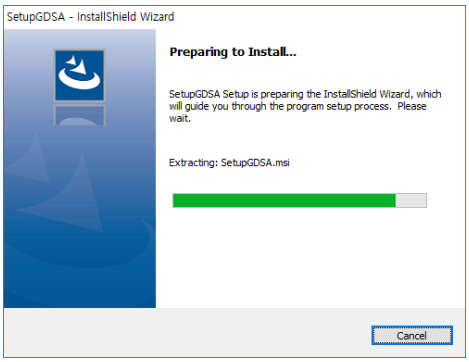
2.1. Software installation

2.1.1. Windows

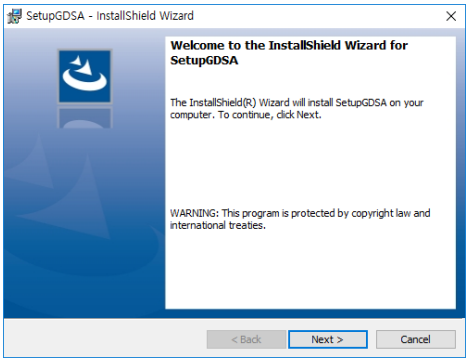
Run “setupGDSA_GUI.exe” from the USB flash drive.
And follow the instruction as following.

1) Double click the file shown in below.

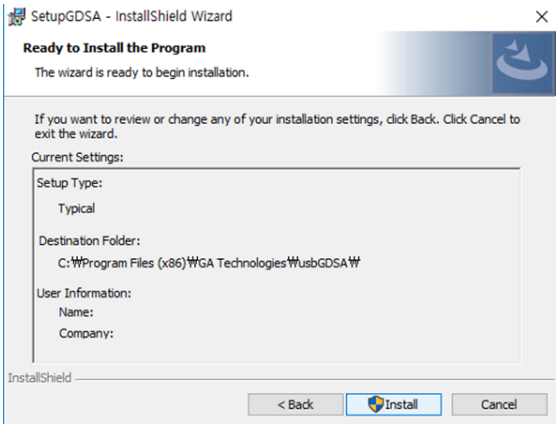
2) Following dialog appears and wait.



3) Click “Next” button.

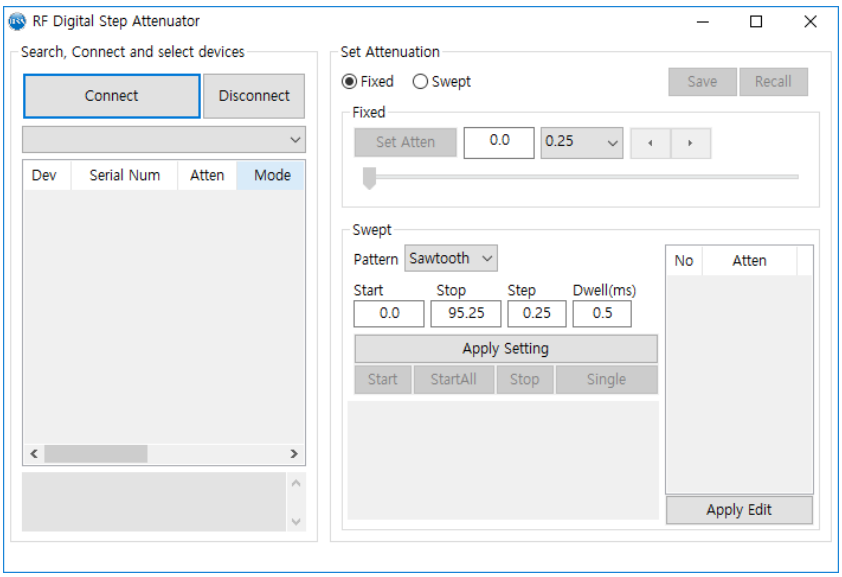


4) Click “Install” button.



5) When installation finished, GUI launch icon appears on desktop screen as below screen capture.

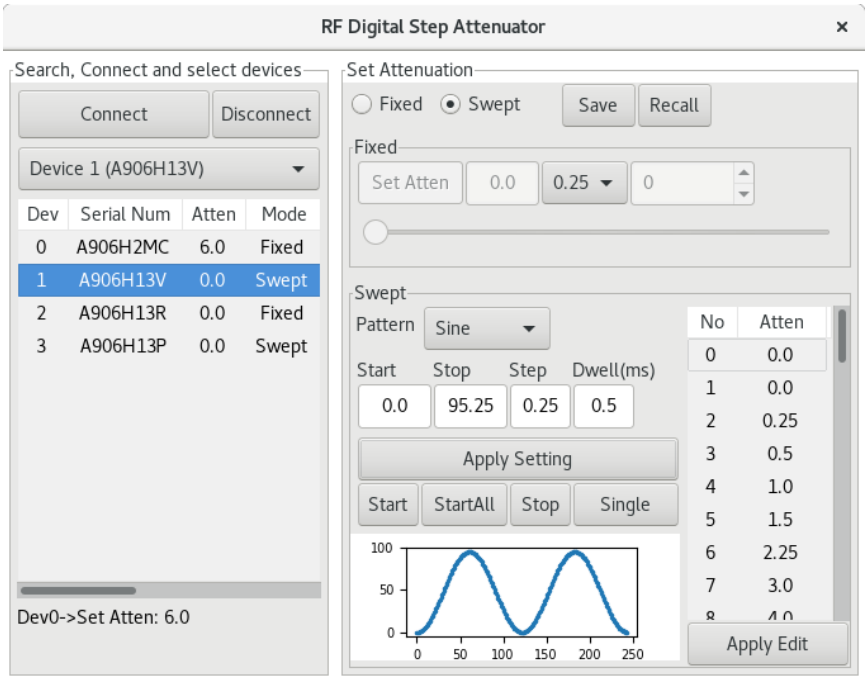
6) Double click the icon to run GUI.



7) API shared libraries are found in the directory as below:
   - Windows 64bit : “C:\Program Files(x86)\GA Technologies\usbGDSA”
   - for Windows 32bit : “C:\Program Files\GA Technologies\usbGDSA”

2.1.2. Linux

1) Extract “*.tar.gz” in a folder as below screen capture.
2) Open “Terminal”, and change directory to the folder.
3) Depending on Linux distribution, run following command:
   - Ubuntu(Version higher than 16.04)
     “sudo ./guiDSA”
   - Debian(Version higher than 9.5)
     “sudo ./guiDSA”
   - CentOS(Version higher than 7)
     “sudo ./guiDSA”
   - OpenSuse(Version higher than 15 Leap)
     “su” then enter root password.
     “xdg-su –c ./guiDSA”



4) API shared libraries are included in the folder.

2.2. Physical setup

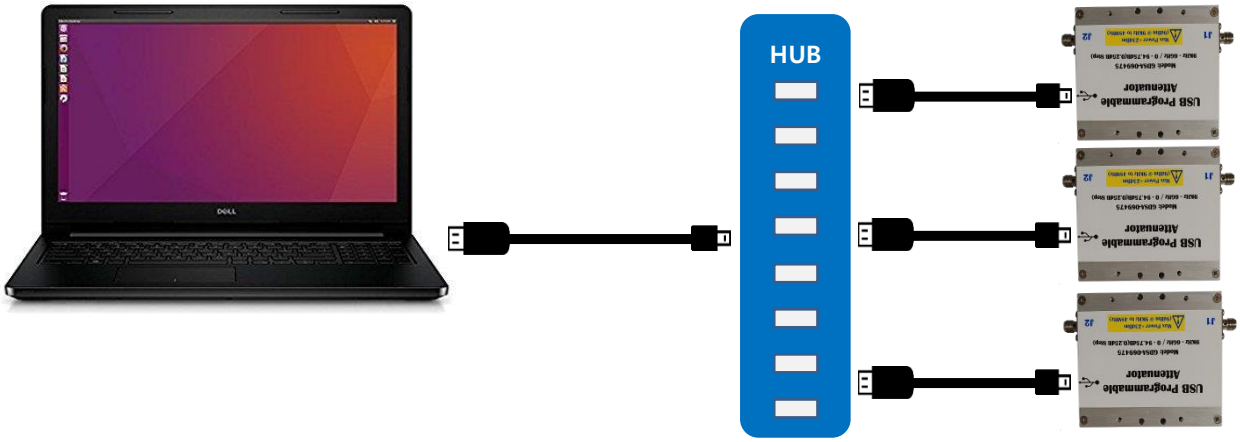
Single device:

GDSA series Digital Step Attenuator is connected to PC with USB 2.0 type A to mini B male cable as below picture.

The attenuator is powered and controlled over the USB cable.
Utilizing GUI or API shared library, user can start to control the attenuator, for example searching, connecting and set attenuation or running attenuation sweep.

Multiple devices:

Multiple devices are connected via USB Hub as below picture.

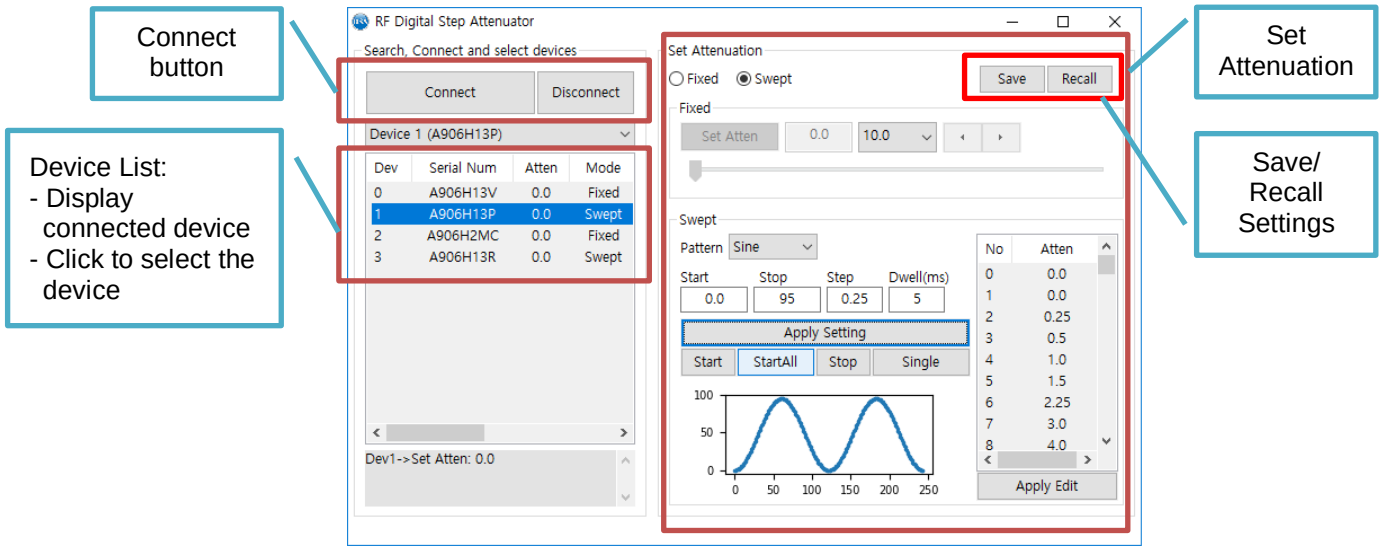


The software, GUI and API shared library is used to detect multiple devices connected via USB hub automatically, and then user can select and control one of the device out of all detected devices.

3. Using software

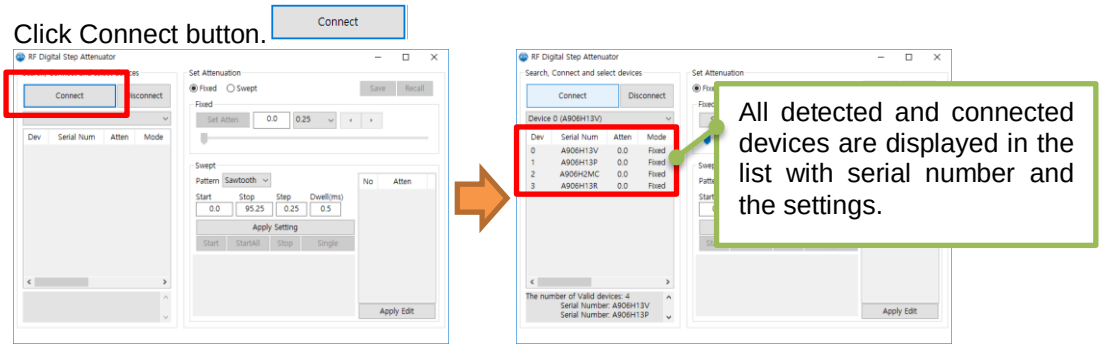
3.1. GUI

GUI consists of mainly 2 parts, “Connect Devices” and “Set Attenuation”.

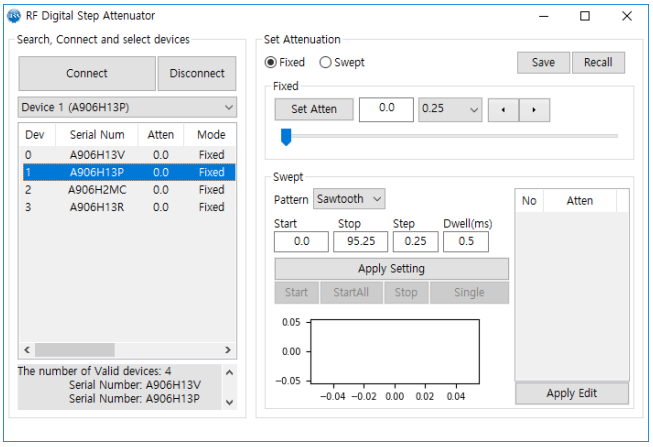


How to use (Operation Procedure):

1) Click Connect button.



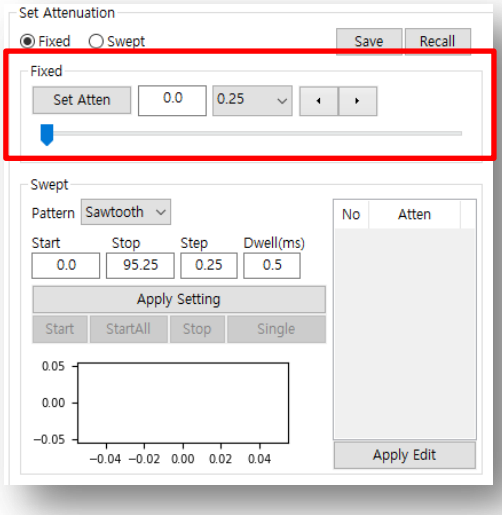
2) Select one of the connected devices on the list



By click on the list, user can choose one of the devices, and then all the parameters from fixed attenuation value to sweep settings are updated accordingly.

3) Choose “Set Attenuation Mode”, “Fixed” or “Swept”.

4) In case of “Fixed”, set attenuation as below.



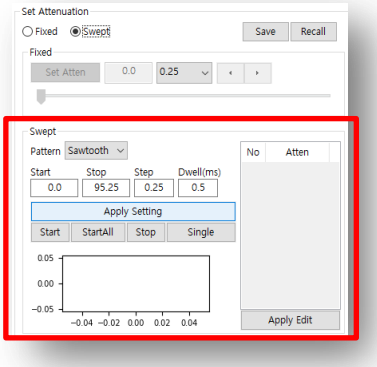
How to set fixed attenuation

I. Edit value in Edit Control + Click “Set Atten” button

II. Choose increment with combo box + click Spin Button to increase or decrease

III. Drag Slider Control with Mouse

5) If “Swept” mode is selected:



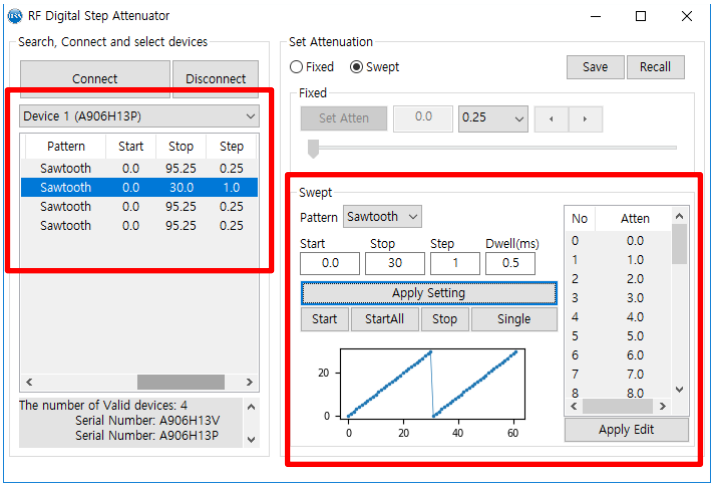
① Choose one of Sweep pattern as below with Combo box
- ✧ Saw Tooth
- ✧ Triangle
- ✧ Sine
- ✧ Random

② Set attenuation “Start”, “Stop”, “Step” and “Dwell” time

6) Click “Apply Setting” button

The parameters for Sweep are updated in the device list.

**All the parameters and settings are displayed when click device on this list.

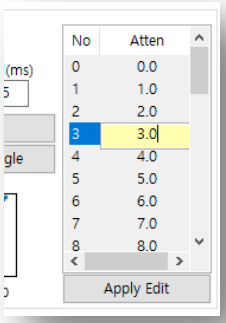


Attenuation list and graph are updated.

**This list is editable. Double click, and then able to type in new value.

7) If necessary edit “Attenuation List” to create user attenuation list:
   ① Double click the list item to enter into edit mode.
   ② Edit the value choose and hit Enter key
   ③ Click “Apply Edit” button to complete editing and save for selected device.

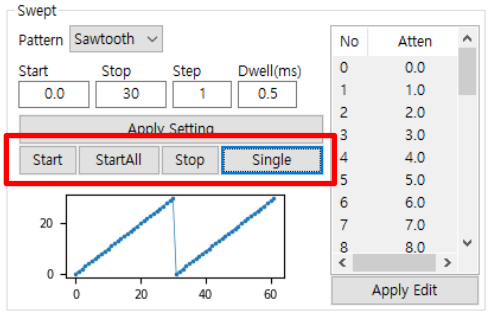
****In this way, custom attenuation list can be created.**



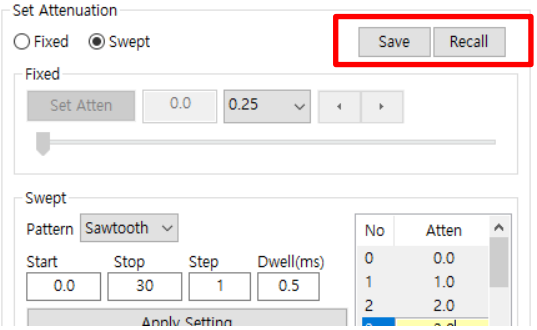
8) Click “Start” button to start Attenuation Sweep of the selected device.
9) Click “Start All” to commence Attenuation Sweep of all devices in “Swept” mode.
   It is necessary to select device and click Stop button to stop running Sweep of the device.
10) “Stop” button is to stop sweep of the selected device.

Attenuation sweep control buttons:

- ✧ **Start**: Start sweep of selected device
- ✧ **StartAll**: Start sweep of all connected device in “Swept” mode.
- ✧ **Stop**: Stop sweep of selected device.
- ✧ **Single**: non continuous sweep, set attenunation when click.



7) “Save” and “Recall” button



This buttons are used to save all necessary settings of the connected devices to *.set file.

This *.set file can be recalled using “Recall” button.

It is very useful when multiple device operation with complicated settings.

Instead of setting all the devices every time, user can finish setting up of the attenuators in short period of time by recalling saved settings.

3.2. Python

3.2.1. Introduction

GDSA series provides API shared library for Python3 as a default. Utilizing this library, test automation can be done with a few lines of code as shown in below script example.

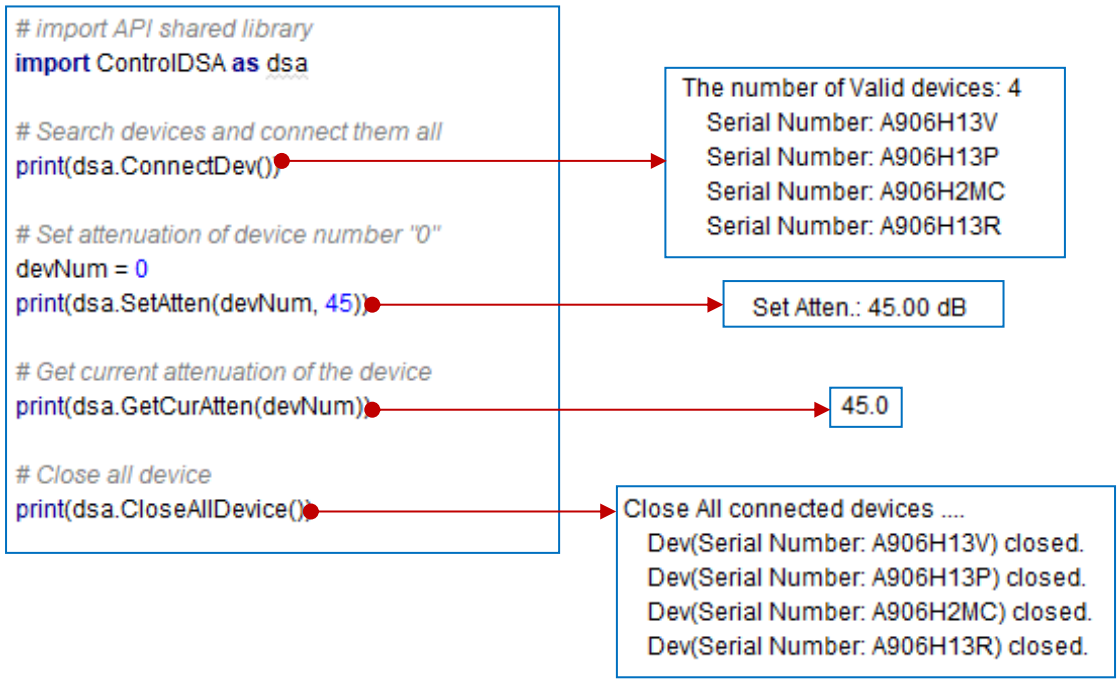
3.2.2. Using API shared library in Terminal or Command Prompt

If PC has Python installed, the library can be used in Terminal (Linux) or Command Prompt (Windows) as below screen capture.

```
C:¥Program Files (x86)¥GA Technologies¥usbGDSA>python
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import ControlDSA as dsa
>>> print(dsa.ConnectDev())
```

3.2.3. Python script example

Following figure shows simple example of Python script and response for each command.



4. Programming

4.1. Search and connect devices

4.1.1.ConnectDev

Summary

Search all valid devices connected directly to PC or via USB hub, and connect them all.

Definition

C, C++ & Python:

```
const char * ConnectDev(void)
```

C#:

```
string ConnectDev()
```

VB.Net:

```
String ConnectDev()
```

Parameters

Return Value

Return number of devices and its serial numbers in char array.

Remarks

This function search devices and connect them if they are valid.

4.2. Close devices

4.2.1.CloseAllDevice

Summary

Close all connected devices..

Definition

C, C++ & Python:

```
const char * CloseAllDevice(void)
```

C#:

```
string CloseAllDevice()
```

VB.Net:

```
String CloseAllDevice()
```

Parameters

Return Value

Return serial numbers of closed devices in char array.

Remarks

4.3. Set fixed attenuation

4.3.1.SetAtten

Summary

Set attenuation of specified device.

Definition

C, C++ & Python:

```
const char * SetAtten(int DevNum, float pAtten)
```

C#:

```
string SetAtten(int DevNum, float pAtten)
```

VB.Net

```
String SetAtten(DevNum As Integer, pAtten As Single)
```

Parameters

DevNum: Device Identification number to set attenuation
pAtten: Attenuation value to set

Return Value

Return result of attenuation setting in char array.

Remarks

4.4. Run sweep or ramps

4.4.1.Generate Attenuation array

4.4.1.1.genAttenListEx

Summary

Generate attenuation command array and set delay time for specified device.

Definition

C & C++:

```
const char * genAttenListEx(int DevNum, float * pList, int lenAtt, float mSecond)
```

Python:

```
const char * genAttenListEx(int DevNum, boost::python::list pList, int lenAtt, float mSecond)
```

C#:

```
string genAttenListEx(int DevNum, float[] pList, int lenAtt, float mSecond)
```

VB.Net:

```
String genAttenListEx(DevNum As Integer pList() As Single, lenAtt As Integer, mSecond
As Single)
```

Parameters

DevNum: Device Identification number to set attenuation
pList: Attenuation value array
mSecond: Dwell time for setting attenuation during sweep

Return Value

Return error messages when occurs.

Remarks

Used to generate and store attenuation command list for one of the connected devices.

4.4.2.Run Sweep

4.4.2.1.StartSweep

Summary

Start sweep of specified device.

Definition

C, C++ & Python:
const char * StartSweep(int devNum)
C#:
string StartSweep(int devNum)
VB.Net:
String StartSweep(devNum As Integer)

Parameters
DevNum: Device Identification number to set attenuation

Return Value
Return sweep start notification of specified device.

Remarks
Prior to run this function, attenuation command list should be generated with "genAttenListEx" function.

4.4.2.2.StartSweepAll

Summary
Start sweep of all connected devices.

Definition
C, C++ & Python:
const char * StartSweepAll(int devNum)
C#:
string StartSweepAll(int devNum)
VB.Net:
String StartSweepAll(devNum As Integer)

Parameters
DevNum: Device Identification number to set attenuation

Return Value
Return error messages when occurs.

Remarks
Prior to run this function, attenuation command list should be generated with "genAttenListEx" function.

4.4.2.3.StopSweep

Summary
Stop sweep of specified device.

Definition
C, C++ & Python:
const char * StopSweep(int devNum)
C#:
string StopSweep(int devNum)
VB.Net:
String StopSweep(devNum As Integer)
Parameters
DevNum: Device Identification number to set attenuation

Return Value

Return sweep stop notification of specified device.

Remarks

4.5. Example code

Python script:

```
#Sweep example
Import ControlDSA as dsa

#Device number to use
dev = 0

#Size of attenuation array
numAtten = 0

#Array for attenuation
swpAtten = []

#Sweep parameters
start = 0
stop = 30
step =1.0
dwell = 2.0

#Calculate number of attenuation values in the array
numAtten = int(stop - start) / step + 0.5) + 1

#Generate attenuation array (Saw Tooth pattern)
tempAtten = start
for j in range(numAtten):
      swpAtten.append(tempAtten)
      tempAtten += step

# Generate Attenuation command list
res = dsa.genAttenListEx(dev, swpAtten, numAtten, dwell)
print(res)

#Start Sweep
res = dsa.StartSweep(int(dev))
print(res)

#Stop sweep of specified device
res = dsa.StopSweep(int(dev))
print(res)
```

C & C++:

```
#include <string>
#include "ControlDSA.h"


int main()
{
   std::string res;

   //...Device to use
   int devNum = 1;

   //1.Connect devices
   res = ConnectDev();
   printf("%s\n\n", res.c_str());
```

```
//1.1.Get valid devices connected
int numValidDev = 0;
numValidDev = GetNumValidDev();
printf("Number of valid devices: %d\n\n", numValidDev);

if (numValidDev > 0)
{
	//1.2.Get Serial Number of specified device
	res = GetSerialNum(devNum);
	printf("SN of dev %d is %s\n\n", devNum, res.c_str());

	//2.Set attenuation
	res = SetAtten(devNum, 40.0f);
	printf("%s\n\n", res.c_str());

	//3.Get current attenuation
	float curAtten = 0.0f;
	curAtten = GetCurAtten(devNum);
	printf("Current Atten of dev# %d:%0.2f\n\n", devNum, curAtten);

	//4.Running Attenuation sweep
	float start, stop, step, mDelay;//Parameters for Attenuation Sweep
	start = 0.0f;
	stop = 95.5f;
	step = 1.0f;
	mDelay = 2.0f;

	//4.1.Calcurate size of the attenuation value array
	int lenAtten = int((stop - start) / step);

	//4.2.Allocate array to store attenuation values
	printf("Generating attenuation list for Sweep...\n\n");
	float * pList = (float*)calloc(lenAtten, sizeof(float));

	//4.2.1. Fill the array with attenuation values according to the parameters
	float temp = 0.0f;
	for (int i = 0; i < lenAtten; i++)
	{
		pList[i] = temp;
		temp += step;
	}

	//4.3.Generate the command list for Sweep
	res = genAttenListEx(devNum, pList, lenAtten, 2.0f);
	printf("%s\n\n", res.c_str());

	//4.4.Start sweep of the device
	res = StartSweep(devNum);
	printf("%s\n\n", res.c_str());

	printf(" Press any key to proceed...\n");
	getchar();

	//4.5.Stop sweep
	res = StopSweep(devNum);
	printf("%s\n\n", res.c_str());
	//Endo of Sweep running

	//5.Close all device
	res = CloseAllDevice();
	printf("%s\n\n", res.c_str());

	//...Free array for attenuation list
	free(pList);
}
```

```
        else
        {
                printf("No valid devices found...\n\n");
        }

        //Exit console
        printf(" Press any key to exit\n");
        getchar();
    }
```

C#:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TestCSconsole
{
    class Program
    {
        static void Main(string[] args)
        {
            dllDSA.dllDSA dsa = new dllDSA.dllDSA();

            string res;
            //...Device to use
            int devNum = 1;

            res = dsa.ConnectDev();
            Console.WriteLine(res);

            //1.1.Get valid devices connected
            int numValidDev = 0;
            numValidDev = dsa.GetNumValidDev();
            Console.WriteLine("Number of valid devices: {0}", numValidDev);

            if (numValidDev > 0)
            {
                //1.2.Get Serial Number of specified device
                res = dsa.GetSerialNum(devNum);
                Console.WriteLine(res);
                Console.WriteLine();

                //2.Set attenuation
                res = dsa.SetAtten(devNum, 40.0f);
                Console.WriteLine(res);
                Console.WriteLine();

                //3.Get current attenuation
                float curAtten = 0.0f;
                curAtten = dsa.GetCurAtten(devNum);
                Console.WriteLine("Current Atten of dev#{0}:{1}", devNum, curAtten);
                Console.WriteLine();

                //4.Running Attenuation sweep
                float start, stop, step, mDelay;//Parameters for Attenuation Sweep
                start = 0.0f;
                stop = 95.5f;
                step = 1.0f;
                mDelay = 2.0f;

                //4.1.Calcurate size of the attenuation value array
                int lenAtten = (int)((stop - start) / step);
```

```
                //4.2.Allocate array to store attenuation values
                Console.WriteLine("Generating attenuation list for Sweep...\n");
                float[] pList = new float[lenAtten];

                //4.2.1. Fill the array with attenuation values according to the parameters
                float temp = 0.0f;
                for( int i = 0; i < lenAtten; i++)
                {
                    pList[i] = temp;
                    temp += step;
                }

                //4.3.Generate the command list for Sweep
                res = dsa.genAttenListEx(devNum, pList, lenAtten, mDelay);
                Console.WriteLine(res);


                //4.4.Start sweep of the device
                res = dsa.StartSweep(devNum);
                Console.WriteLine(res);

                //Exit console
                Console.WriteLine(" Press any key to proceed...\n");
                Console.ReadKey();

                //4.5.Stop sweep
                res = dsa.StopSweep(devNum);
                Console.WriteLine(res);
                //Endo of Sweep running

                //5.Close all device
                res = dsa.CloseAllDevice();
                Console.WriteLine(res);
            }
            else
            {
                Console.WriteLine("No valid devices found...");
            }

            //Exit console
            Console.WriteLine(" Press any key to exit\n");
            Console.ReadKey();
        }
    }
}
```

VB.Net:

```
Imports dllDSA 'Import DSA API for .Net

Module Module1
    Sub Main()
        'Load .Net DSA API
        Dim cDSA As New dllDSA.dllDSA

        'Global variables
        Dim res As String
        Dim devNum As Integer

        'Connect devices
        res = cDSA.ConnectDev()
        Console.WriteLine(res)

        'Toggle Connect and Disconnect button
        Dim numValid As Integer
```

```
        numValid = cDSA.GetNumValidDev()
        Console.WriteLine("Number of valid devices: {0}{1}", numValid, vbCrLf)

        'Set fixed attenuation
        '1. Get attenuation value from the text box
        Dim pAtten As Single
        pAtten = 45.0
        '2. Get device number under control
        devNum = 0
        '3. Set attenuation
        res = cDSA.SetAtten(devNum, pAtten)
        Console.WriteLine(res + vbCrLf)

        'Get current attenuation value to check out the setting
        res = cDSA.GetCurAtten(devNum)
        Console.WriteLine("Current Atten. value: " + res + "dB" + vbCrLf)

        'Example of running Attenuation Sweep
        Dim startAtt As Single
        Dim stopAtt As Single
        Dim attStep As Single
        Dim lenAtten As Int16
        Dim Dwell As Single
        Dim tempAtt As Single

        '1. Get parameters for Sweep
        startAtt = 0.0
        stopAtt = 95.5
        attStep = 1.0
        Dwell = 1.0
        lenAtten = CInt((stopAtt - startAtt) / attStep)
        '2. Generate attenuation array for Sweep
        Dim pList(lenAtten - 1) As Single
        '2.1. Create the attenuation values using the parameters
        tempAtt = startAtt
        For i = 0 To (lenAtten - 1)
            pList(i) = tempAtt
            tempAtt += attStep
        Next i
        '2.2. Get selected device number
        devNum = 0
        '2.3. Generate command list for Sweep
        res = cDSA.genAttenListEx(devNum, pList, lenAtten, Dwell)
        Console.WriteLine(res + vbCrLf + vbCrLf)
        '2.4. Start Sweep
        res = cDSA.StartSweep(devNum)
        Console.WriteLine(res + vbCrLf)
        'Wait for key entry
        Console.WriteLine(" Press any key to proceed...{0}", vbCrLf)
        Console.ReadKey()
        '2.5. Stop Sweep
        res = cDSA.StopSweep(devNum)
        Console.WriteLine(res + vbCrLf)

        'Close All device
        res = cDSA.CloseAllDevice()
        Console.WriteLine(res + vbCrLf)

        'Exit console
        Console.WriteLine(" Press any key to exit\n")
        Console.ReadKey()
    End Sub
End Module
```