User Guide

Spectrum Recorder API

RF Spectrum Data Logger API v1 – Firmware v3.2

Overview

The Spectrum Recorder API allows clients to retrieve sweep data from a Spectrum Recorder device connected to a network. This data includes amplitude information encoded in a base64 string and relevant configuration values such as RBW (Resolution Bandwidth).

Note: This user guide applies to devices using Firmware 3.2 or later. Go to https://www.rfvenue.com/spectrum-recorder-3.2-firmware-update to get the latest firmware upgrade and directions.

Endpoints

API URL	Description
`GET /api/v1/Sweep/GetSweep`	Latest realtime activity, normally updated every 20 seconds with standard span configuration Note this realtime data is only available through the API not from CSV files
`GET /api/v1/Sweep/GetSweep24Avg`	Latest 24hs Average Hold, same as Avg.csv file
`GET /api/v1/Sweep/GetSweep24Max`	Latest 24hs Max Hold, same as MaxHold.csv file
`GET /api/v1/Sweep/GetSweep24Active`	Latest 24hs Activity, same as Active.csv file
`GET /api/v1/Sweep/GetConfig`	Current Configuration settings
`POST /api/v1/Sweep/UpdateFrequency`	Change Start/Stop/Step frequency (new in v3.2)

Sweep Response Format

The response headers include configuration details:

- **`X-RBW:`** `The Resolution Bandwidth (RBW) value in KHz.`
- **`X-StartFreq:`** `The starting frequency of the sweep in MHz.`
- **`X-StopFreq:`** `The stopping frequency of the sweep in MHz.`
- **`X-InputStage:`** `The input stage setting (either` **`Direct`** `or` **`LNA`**`). Note: this is currently fixed in hardware and always operates in Direct mode.`
- **`X-CRC32:`** `Binary sweep checksum`

The sweep response body is packed in a byte array representing amplitude data, which is then compressed with GZip and base64-encoded in a string. This is required to reduce communication latency for the 12K points transmitted. The algorithm to restore the sweep data is as follows:

1. Decode base64 to get the compressed byte array.
2. Decompress the byte array to get the uncompressed byte array.
3. Check that the uncompressed byte array CRC32 matches the one received in the header.
4. Convert the byte array into a float array with amplitude data in dBm/dB using the following calculations:
    a. If sweep data type is not GetSweep24Active , then calculate each iByte element:
    ```
    float fDBM = iByte/-2.0
    ```
    This will provide possible values ranging from -120 to 0 fDBM with 0.5dB resolution.
    b. If sweep data type is GetSweep24Active, then calculate each iByte element
    ```
    float fDB = iByte/2.0
    ```
    This will provide possible values ranging from 0 to +120 fDB with 0.5dB resolution.

Refer to Python code example function **`decompress_string`** to understand the algorithm logic.

Config Response Format

The response body is empty for configuration data request and can be ignored.

The response headers include configuration details:

- **X-RBW:** The Resolution Bandwidth (RBW) value in KHz.
- **X-StartFreq:** The starting frequency of the sweep in MHz.
- **X-StopFreq:** The stopping frequency of the sweep in MHz.
- **X-InputStage:** The input stage setting (either **Direct** or **LNA**).
- **X-SN:** Device Serial Number
- **X-Model:** Device model
- **X-FWVER:** Device Firmware Version
- **X-Date:** Current Date/Time UTC
- **X-TZ:** Current Time zone
- **X-Loc:** Current location
- **X-Files:** Total Files available

Update Frequency Details

Starting with firmware version 3.2, it is now possible to change the frequency configuration using the API as well as the GUI.

The format uses the route: `/StartFrequency(MHz)/StopFrequency(MHz)/StepFrequency(KHz)`

For example, to start at 300.125 MHz and stop at 400 MHz with a 25 KHz step, you can use curl as shown below:

```
curl -i -X POST http://192.168.1.55:80/api/v1/Sweep/UpdateFrequency/300.125/400/25
```

The response headers include configuration details:

- **X-RBW:** The Resolution Bandwidth (RBW) value in KHz.
- **X-StartFreq:** The starting frequency of the sweep in MHz.
- **X-StopFreq:** The stopping frequency of the sweep in MHz.
- **X-InputStage:** The input stage setting (either **Direct** or **LNA**). Note: this is currently fixed in hardware and always operates in Direct mode.

If the input values are invalid, two things may happen:

- **If values are completely out of range** (15–2700 MHz with a minimum span of 10 MHz), the device will return status code 500 to indicate failure, and the command will be ignored.
- **If values are within range but need adjustment**, the firmware will internally adjust them to valid values and return success using the new adjusted values.

Additional Notes:

- In all cases, the caller must check the response headers to confirm that the applied values match the expected configuration.
- RBW is determined automatically based on the requested step frequency.
- A maximum of 50,000 data points is allowed per scan; otherwise, values will be internally adjusted to stay within that limit.
- You can estimate the number of data points with the formula: (StopFrequency - StartFrequency) / StepFrequency
- Note: Frequencies are in MHz, while StepFrequency is in KHz. Be sure to convert units properly when calculating.

Example curl GetSweep Request

```
curl -i -X GET http://LOCALIP/api/v1/Sweep/GetSweep
```

Example Response

- **Status Code**: 200 OK
- **Headers**:
  - Content-Type: application/json; charset=utf-8
  - X-RBW: 25
  - X-StartFreq: 470.000
  - X-StopFreq: 950.000
  - X-InputStage: Direct
- **Body**: "eHhhbXBsZUJhc2U2NEVuY29kZWREYXRheHhhbXBsZUJhc2U2NEVuY29kZWREYXRh"

Error Handling

- **404 Not Found**: The specified resource was not found.
- **500 Internal Server Error**: An unexpected error occurred processing your request.

Python Example

As a fully functional implemented example, the python spectrum_recorder_api_client.py file includes call to each API function, decoding data, CRC32 check and data display on screen.

To run it using Python 3.x, you need to install this lib first

```
pip install requests
```

Then run the example providing IP address of the SpectrumRecorder (local and remote network are ok) and specify port (local port is always 80, use a mapped port if accessing remotely)

```
python3 spectrum_recorder_api_client.py 192.168.1.55 80
```

Example result is listed below:

```
---------- METHOD GetSweep --------------
Resolution BandWidth: 25KHz
Start frequency: 400.0MHz
Stopping frequency: 699.975MHz
Input Stage: Direct
CRC32 Hex: F693098D (received OK)
Total Sweep points: 12000
Minimum amplitude: -110.0 dBm
Maximum amplitude: -57.5 dBm at 627.9 MHz
------------------------------------------------------------
---------- METHOD GetSweep24Max --------------
Resolution BandWidth: 25KHz
Start frequency: 400.0MHz
Stopping frequency: 699.975MHz
Input Stage: Direct
CRC32 Hex: 509D1E3B (received OK)
Total Sweep points: 12000
Minimum amplitude: -93.0 dBm
Maximum amplitude: -40.0 dBm at 433.875 MHz
```

```
------------------------------------------------------------
---------- METHOD GetSweep24Avg --------------
Resolution BandWidth: 25KHz
Start frequency: 400.0MHz
Stopping frequency: 699.975MHz
Input Stage: Direct
CRC32 Hex: 5D16F6D1 (received OK)
Total Sweep points: 12000
Minimum amplitude: -104.0 dBm
Maximum amplitude: -61.0 dBm at 400.025 MHz
------------------------------------------------------------
---------- METHOD GetSweep24Active --------------
Resolution BandWidth: 25KHz
Start frequency: 400.0MHz
Stopping frequency: 699.975MHz
Input Stage: Direct
CRC32 Hex: F9BA23AA (received OK)
Total Sweep points: 12000
Minimum amplitude: 1.0 dB
Maximum amplitude: 60.0 dB at 433.875 MHz
------------------------------------------------------------
---------- METHOD GetConfig --------------
Resolution BandWidth: 25KHz
Start frequency: 400.000MHz
Stopping frequency: 699.975MHz
Input Stage: Direct
Device Serial Number: B34K-55L7-389C-9879
Device model: Spectrum Recorder v1
Device Firmware Version : 3.0.2406.2
Current Date/Time UTC: 2024/06/28 11:51:18
Current Timezone: (UTC+02.00) Europe/Madrid
Current location: Mount Aconcagua
Total Files available: 1584
```