

# LibreVNA Device Protocol

Version 1.3

February 23, 2023

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Hardware interface</b>	<b>3</b>
2.1	USB device	3
2.2	Ethernet interface	3
2.2.1	Device Discovery	3
<b>3</b>	<b>General packet structure</b>	<b>3</b>
<b>4</b>	<b>Packet types</b>	<b>4</b>
4.1	SweepSettings	5
4.2	ManualStatus	7
4.2.1	Hardware Version 0x01	7
4.2.2	Hardware Version 0xFF	8
4.3	ManualControl	8
4.3.1	Hardware Version 0x01	8
4.3.2	Hardware Version 0xFF	10
4.4	DeviceInfo	12
4.5	FirmwarePacket	12
4.6	Ack	13
4.7	ClearFlash	13
4.8	PerformFirmwareUpdate	13
4.9	Nack	13
4.10	Reference	13
4.11	Generator	13
4.12	SpectrumAnalyzerSettings	14
4.13	SpectrumAnalyzerResult	15
4.14	RequestDeviceInfo	15
4.15	RequestSourceCal	15
4.16	RequestReceiverCal	15
4.17	SourceCalPoint	15
4.18	ReceiverCalPoint	16
4.19	SetIdle	16
4.20	RequestFrequencyCorrection	16
4.21	FrequencyCorrection	16
4.22	RequestDeviceConfig	16
4.23	DeviceConfig	16
4.23.1	Hardware Version 0x01	17

4.23.2	Hardware Version 0xFF	17
4.24	DeviceStatus	18
4.24.1	Hardware Version 0x01	18
4.24.2	Hardware Version 0xFF	19
4.25	RequestDeviceStatus	19
4.26	VNADatapoint	19
4.27	SetTrigger	21
4.28	ClearTrigger	21
4.29	StopStatusUpdates	21
4.30	StartStatusUpdates	21
4.31	InitiateSweep	21

## 1 Introduction

This document describes the device protocol of the LibreVNA. This is the protocol used by the LibreVNA to communicate with the LibreVNA-GUI (or other custom implementations). In the context of this document, the LibreVNA is also referred to as the “device” and the LibreVNA-GUI as the “host”.

## 2 Hardware interface

Depending on the LibreVNA, different hardware interfaces may be used for the implementation of this protocol.

### 2.1 USB device

The LibreVNA implements a “custom class” USB device. It uses a VID of 0x0483 and a PID of 0x4121. The custom class contains a single interface with three bulk endpoints:

- **Endpoint 0x01:** Communication data from the USB host to the LibreVNA
- **Endpoint 0x81:** Communication data from the LibreVNA to the USB host
- **Endpoint 0x82:** Debug messages from the LibreVNA

Endpoint 0x82 is exclusively used for debug messages. They are transmitted in ASCII format. All protocol packets described in this document are always transmitted over endpoints 0x01 and 0x81.

### 2.2 Ethernet interface

The ethernet interface implements two TCP servers, one for protocol data and one for debug messages:

- **Port 19544:** Data interface
- **Port 19545:** Debug interface

Each server only supports a single connection. If another connection request is received, the existing connection is closed before accepting the new one.

Incoming data on the debug interface is ignored. Debug messages are transmitted in ASCII format. All protocol packets described in this document are always transmitted over the data interface.

#### 2.2.1 Device Discovery

Initially, the IP addresses of connected LibreVNAs may not be known. To automatically detect any devices, the LibreVNA implements SSDP and responds to M-SEARCH packets looking for either

```
ssdp:all
```

or

```
urn:schemas-upnp-org:device:LibreVNA:1
```

services.

## 3 General packet structure

The data traffic can be viewed as a stream of bytes. The communication between the LibreVNA and the host is done in packets. To detect the packets within the data stream, some framing is needed. This general package structure is described in this section.

Each packet consists of the following fields:

1. **Header:** 1 byte, always 0x5A

2. **Length:** 2 bytes, length of the overall packet in bytes, including the header and the checksum
  3. **Type:** 1 byte, defines the type of packet and subsequently the data encoding within the payload
  4. **Payload:** Any amount of bytes, content depends on the packet type
  5. **CRC:** 4 bytes, CRC32 over all other packet bytes (header, length, type and payload)
- All values in the device protocol are little-endian.

## 4 Packet types

The following packet types are available:

Type	Name	Dir <sup>a</sup>	Description	Answer <sup>a</sup>
2	SweepSettings	H→D	Sets the sweep parameters and starts the sweep in VNA mode	27 <sup>c</sup>
3	ManualStatus	D→H	Contains the hardware status when in manual control mode	None
4	ManualControl	H→D	Transfers the manual control configuration, switches the device into manual control mode	3 <sup>c</sup>
5	DeviceInfo	D→H	Contains the device information (firmware/hardware version, capabilities,...)	None
6	FirmwarePacket	H→D	Contains a piece of firmware data	None
7	Ack	D→H	Sent as a response to every successfully received and handled packet	None
8	ClearFlash	H→D	Triggers the flash erase procedure. Must be issued before transferring firmware data	None
9	PerformFirmwareUpdate	H→D	Triggers the firmware update once all firmware data has been transferred	None
10	Nack	D→H	Sent as a response to every unknown command or failure to execute the requested command	None
11	Reference	H→D	Configure the external/internal reference	None
12	Generator	H→D	Switches the VNA into generator mode and configures the generator output	None
13	SpectrumAnalyzerSettings	H→D	Sets the sweep parameters and starts the sweep in spectrum analyzer mode	14 <sup>c</sup>
14	SpectrumAnalyzerResult	D→H	Sent for every sampled frequency within the sweep in spectrum analyzer mode	None
15	RequestDeviceInfo	H→D	Makes the device send the DeviceInfo packet	5
16	RequestSourceCal	H→D	Makes the device send the source calibration packets	18 <sup>c</sup>
17	RequestReceiverCal	H→D	Makes the device send the receiver calibration packets	19 <sup>c</sup>
18	SourceCalPoint	D↔H	Contains a single source amplitude calibration point	None
19	ReceiverCalPoint	D↔H	Contains a single receiver amplitude calibration point	None

Type	Name	Dir <sup>a</sup>	Description	Answer <sup>a</sup>
20	SetIdle	H→D	Stops all device activity	None
21	RequestFrequencyCorrection	H→D	Makes the device send the frequency calibration packet	22
22	FrequencyCorrection	D↔H	Contains the frequency calibration factor	None
23	RequestDeviceConfig	H→D	Makes the device send its device configuration	24
24	DeviceConfig	D→H	Contains the configuration of various global device settings	None
25	DeviceStatus	D→H	Contains the hardware device status (lock, temperatures,...)	None
26	RequestDeviceStatus	H→D	Makes the device send the device status	25
27	VNADatapoint	D→H	Sent for every sampled frequency within the sweep in VNA mode	None
28	SetTrigger	D↔H	Updates the trigger status for synchronization over the data interface	None
29	ClearTrigger	D↔H	Updates the trigger status for synchronization over the data interface	None
30	StopStatusUpdates	H→D	Stops the automatic transmission of device status packets	None
31	StartStatusUpdates	H→D	Starts the automatic transmission of device status packets	None
32	InitiateSweep	H→D	Initiates a single sweep when configured for standby operation	None

<sup>a</sup> Direction of packet transfer:

- **D→H:** Device to host
- **H→D:** Host to device
- **D↔H:** Both directions used

<sup>b</sup> Packet type that will be sent in response to this packet

<sup>c</sup> The response will be sent multiple times

An Ack is transmitted by the device for every received command after it has been handled successfully.

Received packets from the device are not acknowledged by the host; the host never sends an Ack packet.

#### 4.1 SweepSettings

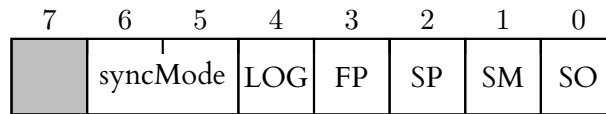
Transmitting this packet will switch the LibreVNA into VNA mode and start the sweep. During the sweep, VNADatapoint packets are generated for each completed point in the sweep.

The sampling for each frequency (or power) point in the sweep is done in stages. In each stage, the stimulus can be active at another port. A typical full two-port sweep would therefore use two stages, with the stimulus being active on port 1 during stage 0 and on port 2 during stage 1. For faster measurements, this could be reduced to a single stage if only a subset of the S-parameters is required. Similarly, more stages than the number of ports can be used (with the stimulus inactive during some) when multiple devices are synchronized. Another device in the setup will have to generate the stimulus during the inactive stages.

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	8	UINT64	f_start	Start frequency in Hz
8	8	UINT64	f_stop	Stop frequency in Hz
16	2	UINT16	points	Number of points in the sweep
18	4	UINT32	IF_bandwidth	Bandwidth of the IF sampling in Hz
22	2	INT16	cdbm_excitation_start	Stimulus power at the first point in $\frac{1}{100}$ dBm
24	1	UINT8	Configuration	Bitmap for configuration, see below
25	2	UINT16	Stages	Bitmap for stage configuration, see below
27	2	INT16	cdbm_excitation_stop	Stimulus power at the last point in $\frac{1}{100}$ dBm

**Configuration:**



- **syncMode:** Synchronization mode when multiple devices are used together

Setting	Synchronization
00	Disabled
01	Protocol
10	Reserved
11	External trigger

- **LOG:** Set for a logarithmic sweep (only for frequency, power adjustment during the sweep is always linear)
- **FP:** Fixed power setting. This must be enabled for power sweeps (when cdbm\_excitation\_start  $\neq$  cdbm\_excitation\_stop)

Setting	Behavior
0	Attenuator setting is fixed during the sweep. This will result in inaccurate stimulus level but prevent discrete jumps in output power.
1	Attenuator setting is changed during the sweep. This will result in more accurate stimulus level but also create discrete jumps in output power.

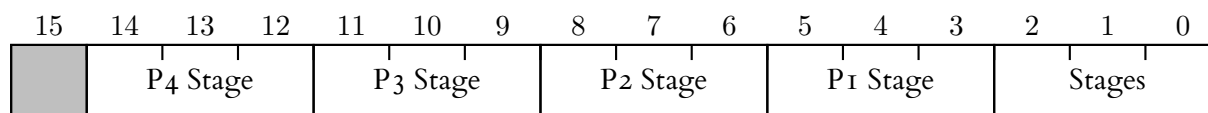
- **SP:** Suppress peaks. Recommended setting: always enabled.

Setting	Behavior
0	2.LO is adjusted to compensate for limited frequency resolution in 1.LO. Slight decrease in maximum sweep speed.
1	2.LO is kept at its nominal value. Slightly faster sweep but this will result in peaks at frequencies where the 1.LO is too far off the ideal frequency.

- **SM:** Sync Master. Must be set to 1 at exactly one device when multiple devices are synchronized. Set to 0 when synchronization is disabled.
- **SO:** Standby Operation. Indicates whether the VNA will begin sweep immediately, or wait in the configured state to be triggered manually by InitiateSweep packets. Standby operation allows for lower latency of intermittent single sweeps.

Setting	Behavior
0	VNA will begin sweep immediately and timeout to idle mode 100ms after sweep is completed or 100ms after entering the halted state.
1	VNA will wait in a configured state for InitiateSweep packets. The host application is responsible for putting the VNA into idle mode with a SetIdle packet.

### Stages:



- **P1 Stage:** Sets the stage number when the stimulus is active at port 1. Stage number indices start at 0.
- **P2 Stage:** Sets the stage number when the stimulus is active at port 2. Stage number indices start at 0.
- **P3 Stage:** Sets the stage number when the stimulus is active at port 3. Stage number indices start at 0.
- **P4 Stage:** Sets the stage number when the stimulus is active at port 4. Stage number indices start at 0.
- **Stages:** Sets the number of used stages. The number of stages is one more than this value. E.g. set to 1 for 2 stages

## 4.2 ManualStatus

This packet is generated by the LibreVNA when in manual control mode. It is transmitted in regular intervals on its own.

The content of this packet varies according to the hardware version reported in the DeviceInfo packet. Each hardware version sends a different ManualStatus packet according to the available hardware information. As the different content is implemented as a “union” in the protocol layer, the packet size always matches the largest content possible. For hardware versions whose content is smaller, the extra bytes can be ignored.

### 4.2.1 Hardware Version 0x01

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	2	INT16	port1min	Minimum value of the ADC at port 1
2	2	INT16	port1max	Maximum value of the ADC at port 1
4	2	INT16	port2min	Minimum value of the ADC at port 2
6	2	INT16	port2max	Maximum value of the ADC at port 2
8	2	INT16	refmin	Minimum value of the ADC at the reference receiver
10	2	INT16	refmax	Maximum value of the ADC at the reference receiver
12	4	FLOAT	port1real	Real part of the complex signal at port 1
16	4	FLOAT	port1imag	Imaginary part of the complex signal at port 1
20	4	FLOAT	port2real	Real part of the complex signal at port 2
24	4	FLOAT	port2imag	Imaginary part of the complex signal at port 2
28	4	FLOAT	refreal	Real part of the complex signal at the reference receiver

Offset	Length	Type	Name	Description
32	4	FLOAT	refimag	Imaginary part of the complex signal at the reference receiver
36	1	UINT8	temp_source	Temperature of the source PLL in °C
37	1	UINT8	temp_LO	Temperature of the LO PLL in °C
38	1	UINT8	Lock status	Bit 0: lock status of source PLL. Bit 1: lock status of LO PLL

#### 4.2.2 Hardware Version 0xFF

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	2	INT16	port1min	Minimum value of the ADC at port 1
2	2	INT16	port1max	Maximum value of the ADC at port 1
4	2	INT16	refmin	Minimum value of the ADC at the reference receiver
6	2	INT16	refmax	Maximum value of the ADC at the reference receiver
8	4	FLOAT	port1real	Real part of the complex signal at port 1
12	4	FLOAT	port1imag	Imaginary part of the complex signal at port 1
16	4	FLOAT	refreal	Real part of the complex signal at the reference receiver
20	4	FLOAT	refimag	Imaginary part of the complex signal at the reference receiver
24	1	UINT8	Lock status	Bit 0: lock status of source PLL. Bit 1: lock status of LO PLL

### 4.3 ManualControl

This packet switches the LibreVNA to manual control mode. As long as the manual control mode is active, the LibreVNA will generate ManualStatus packets and send them to the host.

The content of this packet varies according to the hardware version reported in the DeviceInfo packet. Each hardware version expects a different ManualControl packet according to the available hardware information.

#### 4.3.1 Hardware Version 0x01

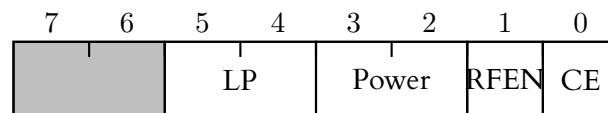
The packet contains the following fields:

Offset	Length	Type	Name	Description
0	1	UINT8	Source High Config	Configuration of the highband source
1	8	UINT64	Source High Frequency	Frequency of the highband source in Hz
9	1	UINT8	Source Low Config	Configuration of the lowband source
10	4	UINT32	Source Low Frequency	Frequency of the lowband source in Hz
14	2	UINT16	Source Path Config	Configuration of the source signal from the PLLs to the ports
16	1	UINT8	1.LO Config	Configuration of the 1.LO
17	8	UINT64	1.LO Frequency	Frequency of the 1.LO in Hz
25	1	UINT8	2.LO Enable	Set to 1 to enable the 2.LO. Set to 0 to disable the 2.LO
26	4	UINT32	2.LO Frequency	Frequency of the 2.LO in Hz



Offset	Length	Type	Name	Description
30	1	UINT8	Receiver enable	Bit 0: Enable port 1 receiver Bit 1: Enable port 1 receiver Bit 2: Enable reference receiver
31	4	UINT32	Samples	Number of ADC samples for each complex wave calculation
32	1	UINT8	WindowType	Window selection for the complex wave calculation

### Source High Config:



- **LP:** Lowpass setting

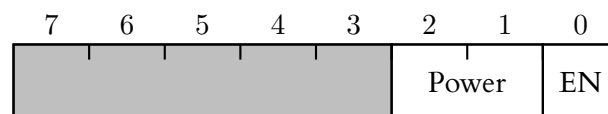
Setting	Cut-off frequency
00	947 MHz
01	1.88 GHz
10	3.5 GHz
11	No filter

- **Power:** Power output of the highband source PLL

Setting	Power
00	-4 dBm
01	-1 dBm
10	2 dBm
11	5 dBm

- **RFEN:** RF output enable
- **CE:** Chip enable

### Source Low Config:

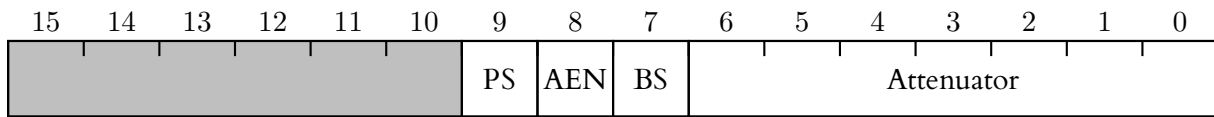


- **Power:** Power output of the lowband source PLL

Setting	Drive Strength
00	2 mA
01	4 mA
10	6 mA
11	8 mA

- **EN:** Lowband source enable

### Source Path Config:



- **PS:** Port switch. Set to 1 to route the source signal to port 2, set to 0 to route the source signal to port 1.
- **AEN:** Amplifier enable.
- **BS:** Band select. Set to 1 to use the highband source, set to 0 to use the lowband source.
- **Attenuator:** Attenuation of the source signal in 0.25 dBm.

### 1.LO Config:



- **RFEN:** RF output enable
- **CE:** Chip enable

### WindowType:

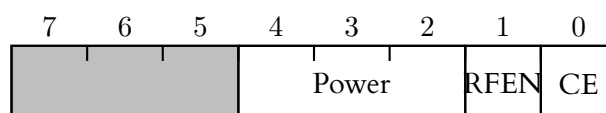
Setting	Window
0	None
1	Kaiser
2	Hann
3	Flattop

### 4.3.2 Hardware Version 0xFF

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	1	UINT8	Source Config	Configuration of the source
1	8	UINT64	Source Frequency	Frequency of the source in Hz
9	1	UINT8	Source Path Config	Configuration of the source signal from the PLL to the port
10	1	UINT8	LO Config	Configuration of the LO
11	8	UINT64	LO Frequency	Frequency of the LO in Hz
19	2	UINT16	Acquisition Config	Configuration of the acquisition hardware
21	2	UINT16	Samples	Number of ADC samples for each complex wave calculation

### Source Config:

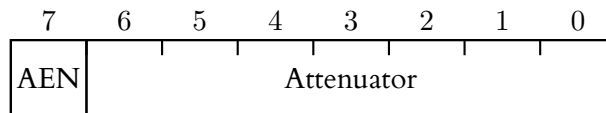


- **Power:** Power output of the source PLL

Setting	Power
000	-1 dBm
001	1 dBm
010	2.5 dBm
011	3.5 dBm
100	4.5 dBm
101	5.5 dBm
110	6.5 dBm
111	7 dBm

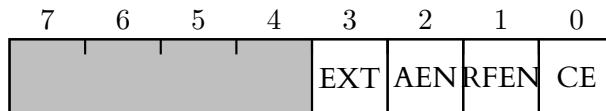
- **RFEN:** RF output enable
- **CE:** Chip enable

#### Source Path Config:



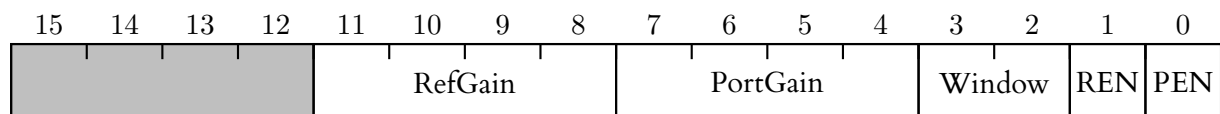
- **AEN:** Amplifier enable.
- **Attenuator:** Attenuation of the source signal in 0.25 dBm.

#### LO Config:



- **EXT:** Use external LO input.
- **AEN:** Amplifier enable.
- **RFEN:** RF output enable
- **CE:** Chip enable

#### Acquisition Config:



- **RefGain/PortGain:** Gain setting of the PGA in the frontend of the port or reference receiver.

Setting	Gain
0	1 V/V
1	10 V/V
2	20 V/V
3	30 V/V
4	40 V/V
5	60 V/V
6	80 V/V
7	120 V/V
8	157 V/V
9	0.25 V/V

- **Window:**

Setting	Window
0	None
1	Kaiser
2	Hann
3	Flattop

- **REN:** Reference receiver enable.
- **PEN:** Port receiver enable.

#### 4.4 DeviceInfo

This packet contains information about the connected device. It can be requested by sending a RequestDeviceInfo packet. This request is the first thing that should happen after the device has been enumerated to make sure the right protocol version is used.

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	2	UINT16	ProtocolVersion	Set to 12. If another value is reported, refer to the corresponding protocol description.
2	1	UINT8	FW_major	Major firmware version
3	1	UINT8	FW_minor	Minor firmware version
4	1	UINT8	FW_patch	Patch of the firmware version
5	1	UINT8	hardware_version	Version of the hardware, currently only '1'
6	1	CHAR	HW_revision	Revision of the hardware, currently only 'B' is used
7	8	UINT64	MinFreq	Minimum supported frequency in Hz
15	8	UINT64	MaxFreq	Maximum supported frequency in Hz
23	4	UINT64	MinIFBW	Minimum supported IF bandwidth in Hz
27	4	UINT64	MaxIFBW	Maximum supported IF bandwidth in Hz
31	2	UINT16	MaxPoints	Maximum number of points per sweep
33	2	INT16	MincdBm	Minimum stimulus power in $\frac{1}{100}$ dBm
35	2	INT16	MaxcdBm	Maximum stimulus power in $\frac{1}{100}$ dBm
37	4	UINT32	MinRBW	Minimum supported resolution bandwidth in Hz
41	4	UINT32	MaxRBW	Maximum supported resolution bandwidth in Hz
45	1	UINT8	MaxAmplitudePoints	Maximum supported number of amplitude calibration points
46	8	UINT64	MaxHarmonicFrequency	Maximum supported frequency when using harmonic mixing
54	1	UINT8	NumPorts	The number of frontend VNA ports the device supports

#### 4.5 FirmwarePacket

This packet contains a part of the firmware. When updating the firmware, this packet must be transmitted multiple times until the whole firmware has been transferred.

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	4	UINT32	Address	Address at which the firmware data starts
4	256	UINT8	Data	Binary firmware data

#### 4.6 Ack

This packet is sent by the device whenever a valid packet has been received. It has no payload.

#### 4.7 ClearFlash

This packet must be sent before transferring the first piece of firmware data. It has no payload.

#### 4.8 PerformFirmwareUpdate

This packet must be sent after the complete firmware data has been transmitted. It triggers the actual update process. The device will reboot during the update process. It has no payload.

#### 4.9 Nack

This packet is sent by the device whenever an error occurred while processing a received packet. It has no payload.

#### 4.10 Reference

This packet is used to configure the external reference input and output.

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	4	UINT32	OutputFrequency	Frequency of the external reference output. Not every frequency can be reached by the PLL. Set to 0 to disable the reference output.
4	1	UINT8	ExternalInputConfig	Bit 0: Switch to external when signal detected Bit 1: Force usage of the external reference

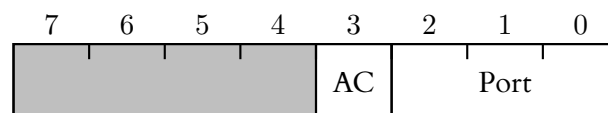
#### 4.11 Generator

This packet switches the LibreVNA into signal generator mode and configures the output signal.

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	8	UINT64	OutputFrequency	Output frequency of the generator in Hz
8	2	INT16	cdBmLevel	Output level in $\frac{1}{100}$ dBm
10	1	UINT8	Configuration	Configuration bitmap, see below

**Configuration:**



- **AC:** Amplitude correction enable. If set to 1, the source amplitude calibration is used to reach better amplitude accuracy.
- **Port:** Port selection:

Setting	Window
0	Disabled
1	Output on port 1
2	Output on port 2
3	Output on port 3
4	Output on port 4

#### 4.12 SpectrumAnalyzerSettings

Transmitting this packet will switch the LibreVNA into spectrum analyzer mode and start the sweep. During the sweep, SpectrumAnalyzerResult packets are generated for each completed point in the sweep.

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	8	UINT64	f_start	Start frequency in Hz
8	8	UINT64	f_stop	Stop frequency in Hz
16	4	UINT32	RBW	Resolution bandwidth in Hz
20	2	UINT16	pointNum	Number of reported points in the sweep. The internally used number of points can be higher (depending on the RBW)
22	2	UINT16	Configuration	Bitmap for configuration, see below
24	8	INT64	TrackingOffset	Offset of the tracking generator in Hz
32	2	INT16	TrackingPower	Power of the tracking generator in $\frac{1}{100}$ dBm

#### Configuration:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SM	syncMode		TGP	ASC	TGE	ARC	DFT		Detector		SID		Window	

- **SM:** Sync Master. Must be set to 1 at exactly one device when multiple devices are synchronized. Set to 0 when synchronization is disabled.
- **syncMode:** Synchronization mode when multiple devices are used together

Setting	Synchronization
00	Disabled
01	Protocol
10	Reserved
11	External trigger

- **TGP:** Tracking generator port. Port count starts at zero. E.g. set this to 1 for tracking generator active on port 2. Ignored if TGE is 0.
- **ASC:** Apply source amplitude corrections. If enabled, the amplitude calibration is used to reach better accuracy of the tracking generator output.
- **TGE:** Tracking generator enable.
- **ARC:** Apply receiver amplitude corrections. If enabled, the amplitude calibration is used to reach better measurement accuracy.

- **DFT:** Use DFT to speed up the acquisition. Can not be used when the tracking generator is enabled. Only useful for low resolution bandwidths.
- **Detector:**

Setting	Detector type
0	Positive peak
1	Negative peak
2	Sample
3	Normal
4	Average

- **SID:** Signal ID enable.
- **Window:**

Setting	Window
0	None
1	Kaiser
2	Hann
3	Flattop

#### 4.13 SpectrumAnalyzerResult

This packet is transmitted by the LibreVNA for every point in the sweep when in spectrum analyzer mode.

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	4	FLOAT	Port 1	Voltage signal level at port 1 (1.0 equals 1 mW into 50 $\Omega$ )
4	4	FLOAT	Port 2	Voltage signal level at port 2 (1.0 equals 1 mW into 50 $\Omega$ )
8	4	FLOAT	Port 3	Voltage signal level at port 3 (1.0 equals 1 mW into 50 $\Omega$ )
12	4	FLOAT	Port 4	Voltage signal level at port 4 (1.0 equals 1 mW into 50 $\Omega$ )
16	8	UINT64	Frequency or Time	Frequency of the point (or time since beginning of SA mode if in zerospan)
24	2	UINT16	PointNum	Number of the point in the sweep

#### 4.14 RequestDeviceInfo

This packet is used to make the device send the DeviceInfo packet. It has no payload.

#### 4.15 RequestSourceCal

This packet is used to make the device send the source amplitude calibration. It has no payload. For each source amplitude calibration point one SourceCalPoint packet will be returned.

#### 4.16 RequestReceiverCal

This packet is used to make the device send the receiver amplitude calibration. It has no payload. For each receiver amplitude calibration point one ReceiverCalPoint packet will be returned.

#### 4.17 SourceCalPoint

This packet contains one source calibration point. It can be transmitted in both directions. When reading the source calibration, it is transmitted from the device to the host. When writing the source

calibration multiple of these packets are transferred from the host to the device. In both cases the packet for the point with the highest point number must be transmitted last.

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	1	UINT8	TotalPoints	Amount of total points in the amplitude calibration
1	1	UINT8	PointNum	Number of the calibration point contained in this packet
2	4	UINT32	Frequency	Frequency of the calibration point in 10 Hz
6	2	INT16	Port 1	Correction value for port 1 in $\frac{1}{100}$ dB
8	2	INT16	Port 2	Correction value for port 2 in $\frac{1}{100}$ dB
10	2	INT16	Port 3	Correction value for port 3 in $\frac{1}{100}$ dB
12	2	INT16	Port 4	Correction value for port 4 in $\frac{1}{100}$ dB

#### 4.18 ReceiverCalPoint

This packet contains one receiver calibration point. It can be transmitted in both directions. When reading the receiver calibration, it is transmitted from the device to the host. When writing the receiver calibration multiple of these packets are transferred from the host to the device. In both cases the packet for the point with the highest point number must be transmitted last.

The packet payload is identical to the SourceCalPoint packet.

#### 4.19 SetIdle

This packet is used to stop any data acquisition from the LibreVNA. It has no payload.

#### 4.20 RequestFrequencyCorrection

This packet is used to make the device send the FrequencyCorrection packet. It has no payload.

#### 4.21 FrequencyCorrection

This packet contains the frequency correction factor for the internal reference. It can be transmitted in both directions. When reading the frequency correction, it is transmitted from the device to the host. When writing the frequency correction, it is transmitted from the host to the device.

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	4	FLOAT	PPM	Error of the internal TCXO in ppm

#### 4.22 RequestDeviceConfig

This packet is used to make the device send the AcquisitionFrequencySettings packet. It has no payload.

#### 4.23 DeviceConfig

This packet contains hardware specific configuration.

The content of this packet varies according to the hardware version reported in the DeviceInfo packet. Each hardware version sends a different DeviceConfig packet according to the available



hardware information. As the different content is implemented as a “union” in the protocol layer, the packet size always matches the largest content possible. For hardware versions whose content is smaller, the extra bytes can be ignored.

These settings are at default values after the device has booted. It is normally not required to send this packet but changing these settings might be useful in special use cases. It can be transmitted in both directions. When reading the acquisition settings, it is transmitted from the device to the host. When writing the acquisition settings, it is transmitted from the host to the device.

#### 4.23.1 Hardware Version 0x01

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	4	UINT32	1.IF frequency	1.IF frequency in Hz
4	1	UINT8	ADC prescaler	Prescaler used for the ADC sampling (refer to the FPGA protocol)
5	2	UINT16	DFT phase increment	Phase increment of the DFT between ADC samples (refer to the FPGA protocol). Together with the ADC prescaler it also sets the 2.IF frequency.

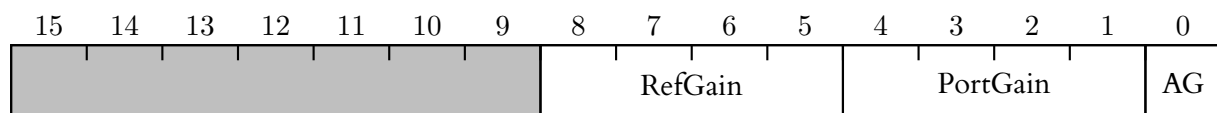
#### 4.23.2 Hardware Version 0xFF

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	4	UINT32	IP address <sup>s</sup>	IPv4 address in network byte order
4	4	UINT32	IP mask <sup>s</sup>	IPv4 mask in network byte order
8	4	UINT32	IP gateway <sup>s</sup>	IPv4 gateway address in network byte order
12	1	UINT8	DHCP <sup>s</sup>	1 if DHCP is enabled, 0 otherwise
12	2	UINT16	Gain Config	Additional gain configuration bits, see below

<sup>s</sup> This parameter is stored on the device and retains its value after a reboot.

#### Gain Config:



- **RefGain/PortGain:** Gain setting of the PGA in the frontend of the port or reference receiver.

Setting	Gain
0	1 V/V
1	10 V/V
2	20 V/V
3	30 V/V
4	40 V/V
5	60 V/V
6	80 V/V
7	120 V/V
8	157 V/V
9	0.25 V/V

- **AG:** Autogain. If set to 1, the gain values are ignored and the best PGA gain for each point is automatically determined while sweeping (reduces sweep speed).

#### 4.24 DeviceStatus

This packet contains the status of the device. It can be requested by sending a RequestDeviceStatus packet. The device also sends this packet on its own. The interval in which this packet is sent depends on the currently active mode.

The content of this packet varies according to the hardware version reported in the DeviceInfo packet. Each hardware version sends a different DeviceStatus packet according to the available hardware information. As the different content is implemented as a “union” in the protocol layer, the packet size always matches the largest content possible. For hardware versions whose content is smaller, the extra bytes can be ignored.

##### 4.24.1 Hardware Version 0x01

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	1	UINT8	StatusBits	Bitmap of various states. See below.
1	1	UINT8	temp_source	Temperature of the source PLL in °C
2	1	UINT8	temp_LO1	Temperature of the 1.LO PLL in °C
3	1	UINT8	temp_MCU	Temperature of the microcontroller in °C

##### StatusBits:

7	6	5	4	3	2	1	0
	ULV	OVL	LLO	SLO	FC	ERU	ERA

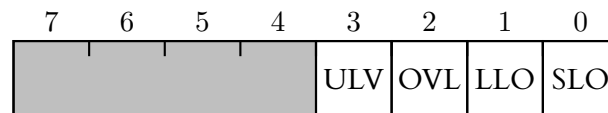
- **ULV:** Unlevel. The requested output signal amplitude can not be reached. This is not actually measured and based on calculations only.
- **OVL:** ADC overload. The amplitude of at least one of the ADCs reached the non-linear region and the signal level can not be trusted.
- **LLO:** 1.LO locked.
- **SLO:** Source locked.
- **FC:** FPGA successfully configured.
- **ERU:** External reference used. The external reference input is used for all PLLs.
- **ERA:** External reference available. A signal is detected at the external reference input.

#### 4.24.2 Hardware Version 0xFF

The packet contains the following fields:

Offset	Length	Type	Name	Description
0	1	UINT8	StatusBits	Bitmap of various states. See below.
1	1	UINT8	temp_MCU	Temperature of the microcontroller in °C

**StatusBits:**



- **ULV:** Unlevel. The requested output signal amplitude can not be reached. This is not actually measured and based on calculations only.
- **OVL:** ADC overload. The amplitude of at least one of the ADCs reached the non-linear region and the signal level can not be trusted.
- **LLO:** I.LO locked.
- **SLO:** Source locked.

#### 4.25 RequestDeviceStatus

This packet is used to make the device send the DeviceStatus packet. It has no payload.

#### 4.26 VNADatapoint

The VNADatapoint packet is generated by the device for every completed sweep point when in VNA mode.



This packet has the CRC set to 0x00000000 as the CRC calculation would take too long when using high IF bandwidths.

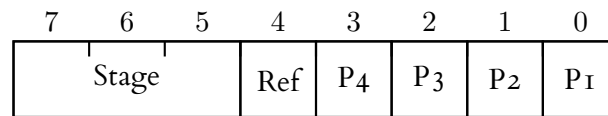
The packet contains the following fields:

Offset	Length	Type	Name	Description
0	8	UINT64	Frequency	Frequency of the sweep point in Hz
8	2	INT16	PowerLevel	Stimulus level of the sweep point in $\frac{1}{100}$ dBm
10	2	UINT16	PointNumber	Number of this point in the sweep
12	4*x	Array of FLOAT	Real values	The real parts of a variable amount of receiver data
12+4*x	4*x	Array of FLOAT	Imag values	The imaginary parts of a variable amount of receiver data
12+8*x	1*x	UINT8	Array of UINT8	Variable amount of data description bit-masks

The sampling data consists of a variable amount of values. The amount of values depend on the amount of configured stages and also on the hardware architecture (might change in the future). The VNADatapoint contains three arrays of equal length. Two of the arrays contain the real and imaginary parts of the acquired data. The third array contains a bitmask for every value, describing the content. The length of all arrays is not explicitly transmitted and must be inferred from the

overall packet length.

### Data description bitmask:



- **Stage:** The active stage when the value was acquired. The port on which the stimulus was active during this stage is known from the SweepSettings packet that was used to set up the currently active sweep.
- **Ref:** The value is from a reference receiver.
- **P4:** The value is from a port 4 receiver.
- **P3:** The value is from a port 3 receiver.
- **P2:** The value is from a port 2 receiver.
- **P1:** The value is from a port 1 receiver.

In case of a three receiver architecture (as the LibreVNA 1.0 has), multiple port bits can be set for reference receiver values. For a typical full two-port sweep, the LibreVNA 1.0 will generate six values for every sweep point:

#	Bitmask	Content
1	0x01	Port 1 receiver signal during stage 0
2	0x02	Port 2 receiver signal during stage 0
3	0x13	Reference receiver signal during stage 0
4	0x21	Port 1 receiver signal during stage 1
5	0x22	Port 2 receiver signal during stage 1
6	0x33	Reference receiver signal during stage 1

The host must assemble the S-parameter data from these receiver values. This calculation has to be offloaded to the host because the reference and port receiver measurements may be split across different devices when synchronizing multiple LibreVNAs.

### Example procedure to assemble S21

Some definitions and assumptions:

- S21 is the through measurement from port 1 to port 2, meaning we need the reference receiver data from port 1 and the port receiver data from port 2
- Port 1 had the stimulus signal at stage 0 and port 2 had the stimulus signal at stage 1. This is the default for a full two-port sweep. If configured differently in the SweepSettings packet, adjust the stage values accordingly

The host must perform the following operations:

1. Wait for reception of a VNADatapoint packet
2. Determine the array length of the received data:

$$array\_length = (packet\_size - 12) / 9$$

3. Find the port receiver data of port 2 for the correct stage (when port 1 had the stimulus, in this example stage 0)
  - (a) Iterate over all data description bitmasks in the VNADatapoint
  - (b) Find the one with bitmask 0b0000xx1x (stage 0, port 2, not a reference receiver measurement). Bits marked “x” are “don’t care”.

- (c) Note the index  $n$  of this data description bitmask in the data description bitmask array
- (d) Use the index  $n$  to get the real and imaginary values of the port receiver data from the real and imaginary arrays:

$$port\_receiver = Real\_values[n] + i * Imag\_values[n]$$

4. Find the reference receiver data of port I for the correct stage (when port I had the stimulus, in this example stage 0)
  - (a) Iterate over all data description bitmasks in the VNADatapoint
  - (b) Find the one with bitmask 0b0001xxx1 (stage 0, port I, reference receiver measurement). Bits marked “x” are “don’t care”.
  - (c) Note the index  $n$  of this data description bitmask in the data description bitmask array
  - (d) Use the index  $n$  to get the real and imaginary values of the reference receiver data from the real and imaginary arrays:

$$reference\_receiver = Real\_values[n] + i * Imag\_values[n]$$

5. Calculate  $S_{21}$  as the ratio between the port and reference receiver data:

$$S_{21} = \frac{port\_receiver}{reference\_receiver}$$

#### 4.27 SetTrigger

This packet is used when multiple devices are synchronized over the data protocol and can be transmitted in both directions. It has no payload. Synchronized devices must be logically organized in a closed loop. When a SetTrigger packet is received from any devices in the loop it must be passed on to the next device in the loop.

#### 4.28 ClearTrigger

This packet is used when multiple devices are synchronized over the data protocol and can be transmitted in both directions. It has no payload. Synchronized devices must be logically organized in a closed loop. When a ClearTrigger packet is received from any devices in the loop it must be passed on to the next device in the loop.

#### 4.29 StopStatusUpdates

This packet instructs the device to stop sending automatically scheduled DeviceStatus packets. Device status can still be requested explicitly via RequestDeviceStatus packets.

#### 4.30 StartStatusUpdates

This packet instructs the device to start sending automatically scheduled DeviceStatus packets. This restores default update behaviour if a StopStatusUpdates packet has previously been sent.

#### 4.31 InitiateSweep

This packet instructs the device to initiate a new single sweep when the VNA is configured for standby operation. This triggering method can be used for fast intermittent single sweeps with minimum latency. If the SweepSettings are not configured for standby operation, this packet will result in a Nack response.