HOASYS

Holographie Adaptative et Systèmes

HOA20K UI

User Manual — software version 1.10

November 2026

HOASYS SAS — www.hoasys.fr

Contents

1 Introduction 2
2 Installation 2
2.1 Windows 2
2.2 macOS 2
2.3 Linux 2
3 Quick start 2
4 User interface 3
4.1 Connection 3
4.2 Sinusoidal tab 4
4.3 Sinusoidal Sweep tab 5
4.4 Device Status 6
4.5 Emergency stop and log console 6
5 Python interface 6
5.1 A complete example 7
5.2 Automation and AI assistants 7
6 Serial protocol reference 8
7 Troubleshooting 8
8 Support 9

1 Introduction

HOA20K UI is the desktop control application for the HOASYS **HOA20K** high-voltage amplifier / signal generator. The application talks to the device over its USB serial port and provides:

- setting of the output **voltage** (0–28 V RMS, 0.01 V resolution) and **frequency** (0–20 000 Hz);
- automatic **amplitude sweeps** (voltage ramp up/down) and **frequency sweeps**, repeating with a programmable period;
- live device telemetry: temperature, battery level, output state;
- DC offset **calibration** of the output;
- a command **terminal** for direct access to the device protocol;
- a **Python interface** to connect and set the voltage and frequency from a script, a notebook, lab automation software or an AI assistant;
- a prominent **EMERGENCY STOP** that disables the output immediately.

HOA20K UI runs on Linux, Windows 10 or later, and macOS. The user interface is identical on the three platforms.

2 Installation

2.1 Windows

Run the installer `HOA20K_UI-<version>-setup.exe` and follow the wizard. It installs the application, creates a Start Menu entry and an optional desktop icon, and registers an uninstaller.

Alternatively, the portable `HOA20K_UI-<version>-win64.zip` needs no installation: unzip it anywhere and run `HOA20K_UI.exe`.

The device uses a standard FTDI USB-serial interface; Windows installs the driver automatically (a `COM` port appears in the Device Manager when the device is plugged in).

2.2 macOS

Open `HOA20K_UI-<version>.dmg` and drag **HOA20K_UI** to the **Applications** shortcut. The application is not notarized: on first launch, right-click the app and choose *Open*. No driver installation is needed; the device appears as `cu.usbserial-…`.

2.3 Linux

On Arch Linux, install the package (`sudo pacman -U hoa20k-ui-<version>-1-x86_64.pkg.tar.zst`). On any other distribution, use the AppImage: make it executable (`chmod +x HOA20K_UI-<version>-x86_64.AppImage`) and run it.

Your user must be allowed to open serial ports — on most distributions, add yourself to the `dialout` (Debian/Ubuntu) or `uucp` (Arch) group and log in again.

3 Quick start

1. Connect the HOA20K to the computer with the USB cable and switch it on.
2. Start HOA20K UI and select the device's port in the **Connection** list (`ttyUSB0`, `COM3`, `cu.usbserial-…`, depending on the platform), then press **CONNECT**.
3. The status LED turns **green** and the **output is enabled** automatically. Telemetry (temperature, battery) starts updating once per second.
4. Set the output with the **Voltage** and **Frequency** controls, or start a sweep from the **Sinusoidal Sweep** tab.
5. Switch the output off and on again with the **OUTPUT OFF / OUTPUT ON** button; press **EMERGENCY STOP** at any time to disable the output.

4 User interface

4.1 Connection

The topmost section selects the serial port and establishes the connection. The round status LED shows the link state:

LED	Meaning
red	not connected
orange	connected, output off
green	connected, output on

Connecting enables the device output. The application verifies the link by querying the device before declaring it connected; if the device does not answer, the log console shows the reason.

The **OUTPUT ON / OUTPUT OFF** button next to **CONNECT** switches the output without disconnecting. The button shows the action it will perform: it reads **OUTPUT OFF** while the output is on (LED green) and **OUTPUT ON** while it is off (LED orange). With the output off the device delivers 0 V; the set voltage and frequency are kept and are restored when the output is switched on again. The button is greyed out while the application is not connected.

4.2 Sinusoidal tab

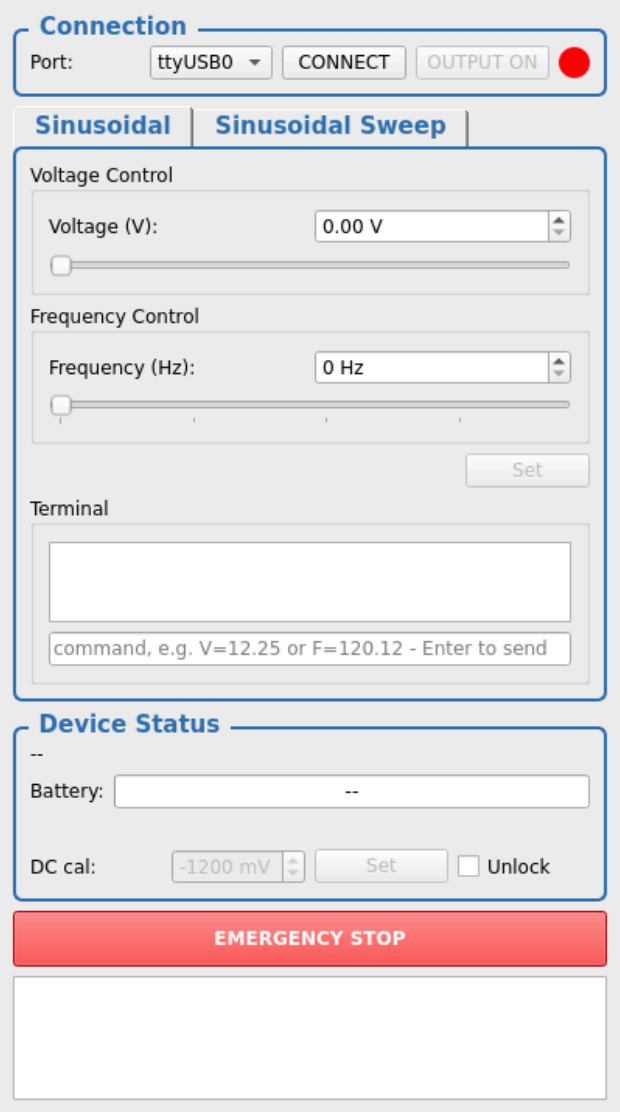


Figure 1: Sinusoidal tab

The **Sinusoidal** and **Sinusoidal Sweep** tabs are exclusive: the visible tab selects which controls are shown, and a running sweep keeps running if you switch tabs.

4.2.1 Voltage Control

Sets the output amplitude, from 0 to 28 V RMS with 0.01 V resolution, with the slider or the spinbox. The value is only staged: click **Set** to send it to the device (see below). The 28 V limit is enforced by the application on every path that can set a voltage, including the terminal.

4.2.2 Frequency Control

Sets the output frequency, from 0 to 20 000 Hz in whole hertz, with the slider or the spinbox. The slider is **logarithmic** and weighted towards the low frequencies: 1 Hz to 1 kHz occupies the left 80 % of its travel, with one tick mark per decade (1, 10, 100, 1 kHz), and 1 kHz to 20 kHz is compressed into the right 20 %. A few hertz can thus be dialled in as easily as a few hundred. The far left position is 0 Hz. The arrow keys move the slider by about 1 %, PageUp/PageDown by about 25 %; type in the spinbox for an exact value. As for the voltage, the value is staged until you click **Set**.

4.2.3 Set

Sends the edited voltage and/or frequency to the device. Moving a slider or typing in a spinbox does not change the output by itself: the button becomes active as soon as one of the two values differs from what the device holds, and a click sends only the value(s) you changed. Until then the edited value is kept on screen (the periodic device readback does not overwrite it). Each command applied by the device rebuilds its waveform, so staging the edit avoids a burst of commands while a slider is being dragged.

4.2.4 Terminal

A three-line terminal for sending protocol commands directly, e.g. `V=12.25`, `F=1200` or `CAL?` (see section 5). Type the command and press *Enter*; the command and the device's reply appear in the pane.

- input is converted to uppercase automatically (`v=12.25` works);
- `V=` values are clamped to 0–28 V and keep two decimals;
- `F=` values are truncated to whole hertz;
- the UI controls follow the values typed here;
- the terminal is disabled while a sweep is running.

4.3 Sinusoidal Sweep tab

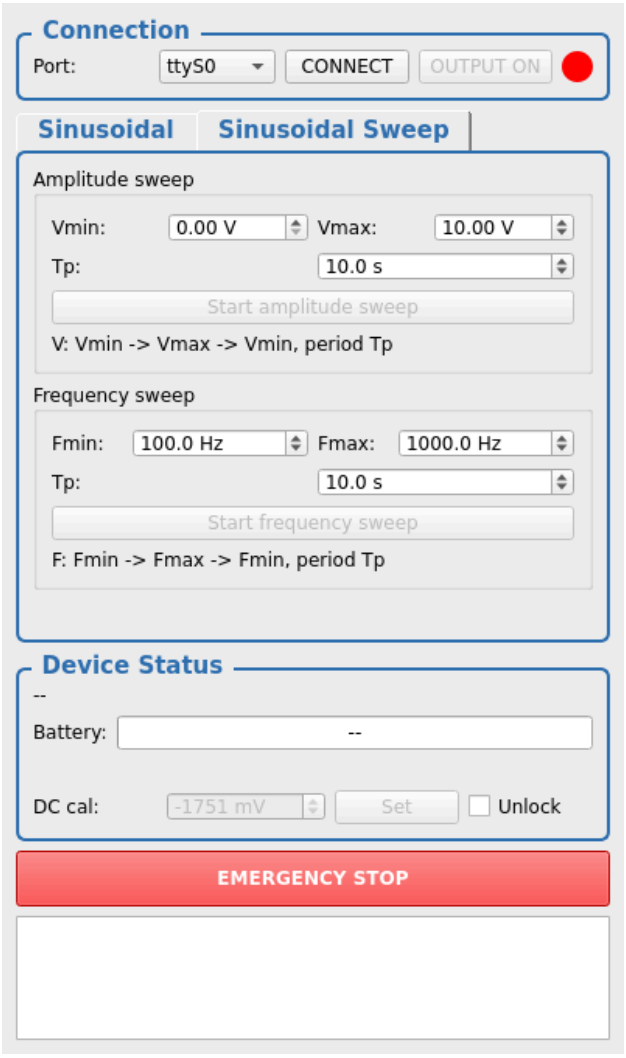


Figure 2: Sinusoidal Sweep tab

Both sweeps repeat a linear **triangle profile** (*min* → *max* → *min*) with period **Tp** (2 s to 3600 s). While a sweep runs, the manual voltage/frequency controls and the terminal are disabled; the other sweep cannot be started at the same time. The sweep's own **Stop** button, or the EMERGENCY STOP, ends it.

4.3.1 Amplitude sweep

Sweeps the output voltage between **Vmin** and **Vmax**. Steps are kept at 50 mV or less, so the ramp stays smooth regardless of the period.

4.3.2 Frequency sweep

Sweeps the output frequency between **Fmin** and **Fmax**, in steps of 10 Hz or less; the voltage is not modified.

Note. The sweep modes stream many settings per second to the device. They must only be used with device firmware that does not save each setting to permanent memory (firmware supplied by HOASYS for this purpose) — contact HOASYS if unsure.

4.4 Device Status

- **Temp** — internal temperature of the device, in °C.
- **Battery** — battery voltage shown as a colored gauge: 4.00 V or more reads as *charged*; the bar turns orange below half charge and red when nearly empty.
- **DC cal** — DC offset compensation of the output, in millivolts (±5000 mV). Verify with a multimeter that the DC voltage at the output is almost zero; if it is not, adjust the compensation value to minimize it (measure the DC component and set its opposite to null it). The controls are behind a safety lock: tick **Unlock**, set the value, press **Set**. The value is applied immediately and **saved permanently in the device**, so this is meant for occasional calibration only, not for repeated adjustments. On connection the application reads back the compensation stored in the device.

4.5 Emergency stop and log console

The red **EMERGENCY STOP** button disables the output immediately and stops any running sweep. The log console at the bottom of the window records every command sent, the device replies, and connection events; check it first when something behaves unexpectedly.

5 Python interface

A Python script can drive the running application: connect to the device, set the output voltage and frequency, switch the output on and off, and read the telemetry. The window follows what the script does, exactly as if the operator had used the controls.

The application listens on the local machine only (`127.0.0.1`, port **5108**); nothing is reachable from the network. The status bar at the bottom of the window shows the port and whether a script is attached.

The module `hoa20k.py` needs only a standard Python 3 (no extra package). It is installed with the application: in `/usr/share/hoa20k-ui/python` on Linux, and in the `python` folder of the Windows package and of the macOS disk image. Copy it next to your script, or add its folder to `PYTHONPATH`.

Start HOA20K UI, then:

```
from hoa20k import HOA20K

with HOA20K() as gen:
    gen.connect()                  # same as the CONNECT button
    gen.voltage = 12.5             # volts RMS, 0 to 28
    gen.frequency = 1000           # hertz, 0 to 20000
    gen.set(voltage=5, frequency=250)    # both in one call
    print(gen.voltage, gen.frequency)
    print(gen.temperature, gen.battery)
    gen.output = False             # output off, settings kept
```

Member	Meaning
`connect(port=None)`	connect to the device, like **CONNECT**
`ports()`	list of the serial ports the application sees
`voltage`	output voltage, V RMS (read / write, 0.01 V resolution)
`frequency`	output frequency, Hz (read / write, whole hertz)
`set(voltage=, frequency=)`	set one or both, returns the new state
`output`	output enabled, `True` / `False` (read / write)
`temperature`, `battery`	last telemetry reading (read only)
`connected`	the application is connected to the device (read only)
`modulation`	running sweep, `"none"` when idle (read only)
`state()`	all of the above in one dictionary

Each assignment returns once the device has applied the value (typically 10 to 100 ms), so no pause is needed between two settings. A value outside the limits, a device that is not connected, or a running sweep raises `HOA20KError` with an explanatory message, and nothing is sent. `example_basics.py`, next to the module, is a commented tour.

`connect()` does what the **CONNECT** button does, including switching the output **on** at the voltage and frequency the device holds. It uses the port selected in the window, or the one named (`gen.connect("ttyUSB0")`, `gen.connect("COM3")`), and does nothing when the application is already connected.

The DC calibration and the sweeps are deliberately not available from Python; they remain operator actions in the window.

> **Note.** With the standard firmware the device stores every new voltage or frequency in its permanent memory, which has a limited number of write cycles. Scripts that step the output many times (thousands of settings) require the no-save firmware; contact HOASYS.

To use another port, start the application with `--port N` and pass `port=N` to `HOA20K()`; `--port 0` disables the interface.

5.1 A complete example

A short measurement script: connect, step the voltage at a fixed frequency, and leave the output at 0 V.

```
import time
from hoa20k import HOA20K

with HOA20K() as gen:
    gen.connect()                    # output on, like CONNECT
    gen.frequency = 190              # hertz
    for volts in (0.5, 1.0, 1.5, 2.0):
        gen.voltage = volts          # returns when applied
        time.sleep(1.0)              # ... take the measurement ...
        print(volts, "V ->", gen.state())
    gen.voltage = 0
```

5.2 Automation and AI assistants

The interface is ready to be used by other software as it is: by scripts and Jupyter notebooks, by lab automation and acquisition programs, and by AI assistants and coding agents (Claude and similar), which can then operate the generator on request, e.g. *“connect and set 1 V at 190 Hz”*.

Nothing else has to be installed or configured for this:

- `hoa20k.py` is a single file using only the Python standard library, and it documents itself (`help(HOA20K)`);
- every refusal comes back as a plain-language message (value out of range, device not connected, sweep running), so a program or an assistant can understand it and correct itself;
- `state()` returns everything in one dictionary, to check the result of each action;
- the underlying protocol is plain text over a local TCP port (one command per line, one line of JSON back, see the README), so any language can use it without the Python module.

The safeguards of the application stay in force whoever drives it: the 0–28 V limit is enforced by the application, the DC calibration and the sweeps are not reachable, and the interface accepts connections from the local computer only. The window shows every action as it happens, and the **EMERGENCY STOP** button remains available to the operator at any time.

6 Serial protocol reference

The device communicates at **115200 baud, 8N1**, no flow control. Commands are ASCII lines terminated by CR LF. These commands can be typed in the terminal:

Command	Effect
`ON` / `OFF`	enable / disable the output
`V=<v>`	set voltage, volts (2 decimals)
`F=<f>`	set frequency, hertz
`V=?`	read voltage
`F=?`	read frequency
`B=?`	read battery voltage
`T=?`	read temperature
`CAL=<mv>`	set DC compensation, millivolts (signed, saved in the device)
`CAL?`	read DC compensation

7 Troubleshooting

The port does not appear in the list. Check the USB cable and that the device is powered. On Linux, verify you belong to the serial group (`dialout`/`uucp`). The list is read when the application starts: restart it after plugging the device.

CONNECT fails. Another program may hold the port open (close serial monitors, other instances of HOA20K UI). Verify you selected the right port.

The device stops answering. Power-cycle the device completely (switch it off, disconnect USB, reconnect). If the application was mid-operation, restart it as well.

macOS blocks the first launch. The application is not notarized: right-click **HOA20K_UI** in Applications and choose *Open* the first time.

Python: "Connection refused". HOA20K UI is not running, or it uses another port: the status bar shows the port in use. If it reads *port 5108 unavailable*, another instance of the application (or another program) already holds the port; close it or start with `--port N`.

Python: "device not connected". Call `gen.connect()` or click **CONNECT** in the window first; the script drives the application, which holds the serial port.

Which version am I running? The version is shown in the window title, e.g. *HOA20K UI v1.10*.

8 Support

For questions and support:

HOASYS SAS
120 route des macarons
06560 Valbonne, France

E-mail: **contact@hoasys.fr**
Web: **www.hoasys.fr**