EnSpectr autonomous power controller – telemetry information monitoring

Table of contents

1. Introduction

EnSpectr spectrometers are widely used for research and identification of materials, quality control in chemical and pharmaceutical industries, and mixture composition analysis. They operate based on observing luminescence and Raman scattering spectra of the material under laser irradiation. To ensure prolonged autonomous operation from batteries, EnSpectr developed a universal autonomous power controller. This controller not only powers the spectrometer from rechargeable batteries but also charges them when an external power source is available. Additionally, it provides wireless communication with the spectrometer, transmits telemetry data to the spectrometer and external applications via Bluetooth, and includes an optical sensor that informs the spectrometer about the presence of a sample.

2. Architecture of the autonomous power controller

The simplified structural diagram of the autonomous power controller shows that the charge controller (1) is responsible for charging the battery (2) when external power is available. The power switch (3) supplies power to the controller elements and the connected load (spectrometer) from either the battery or external power when available. A circuit comprising a DC/DC converter (4) and a linear

stabilizer (5) provides low-voltage power to the microcontroller (6), BT adapter (7), and sample sensor (9). The output converter (8) generates the required output voltage for the load as commanded by the microcontroller (6). The optical sample sensor (9) detects the sample via reflected light and sends a presence signal to the spectrometer. The Bluetooth adapter (7) facilitates two-way data transmission between the spectrometer and the control software on a computer, and also provides a one-way telemetry data channel using Bluetooth Low Energy (BLE) protocol, which can be received on a computer or mobile device via a special web application. This application is described in the following sections.

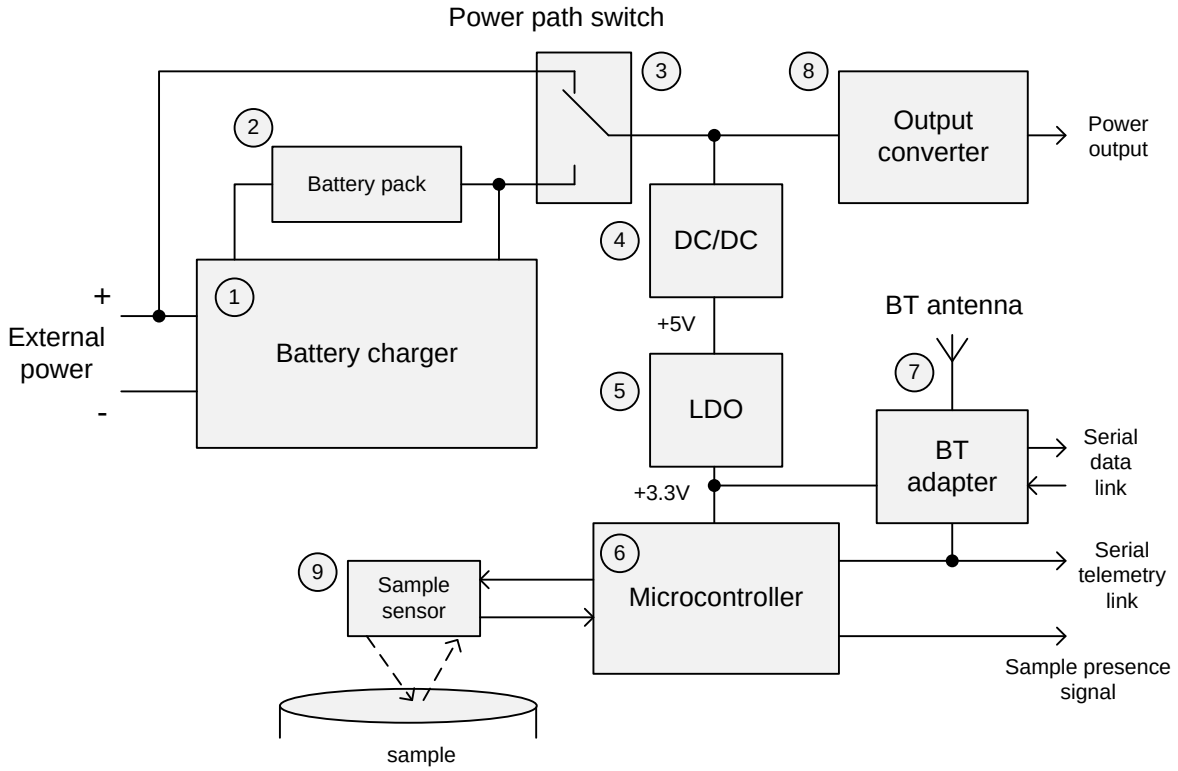

Figure 1. Device diagram.

3. Application for receiving telemetry information

3.1. Launching the application

To open the application, go to the link below.

https://enspectr.github.io/espw/

If the browser does not support Bluetooth, an error message will appear.

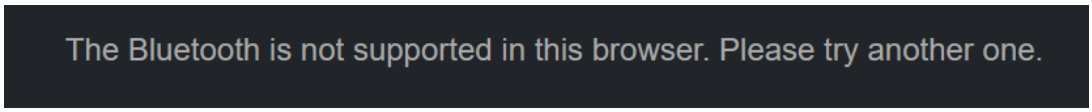

Figure 2. Error message.

For solutions, refer to the section "Application Specifics on Different Platforms".

If the browser supports Bluetooth, the application page will load.

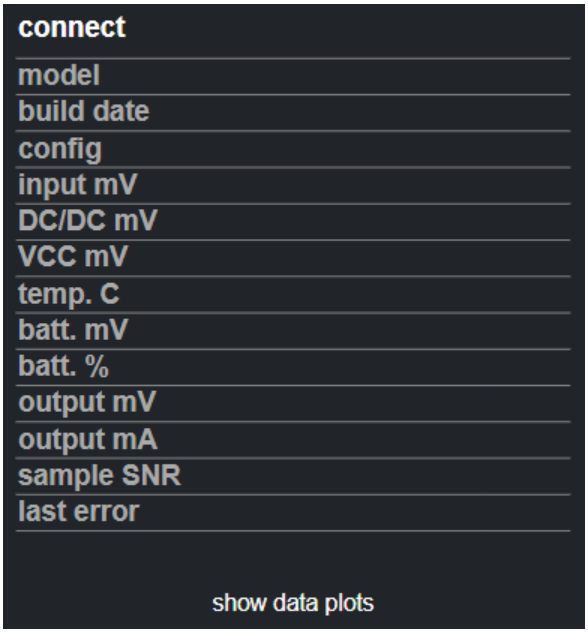

Figure 3. Application page.

To start, click the “connect” button, which will open a menu with a list of Bluetooth devices.

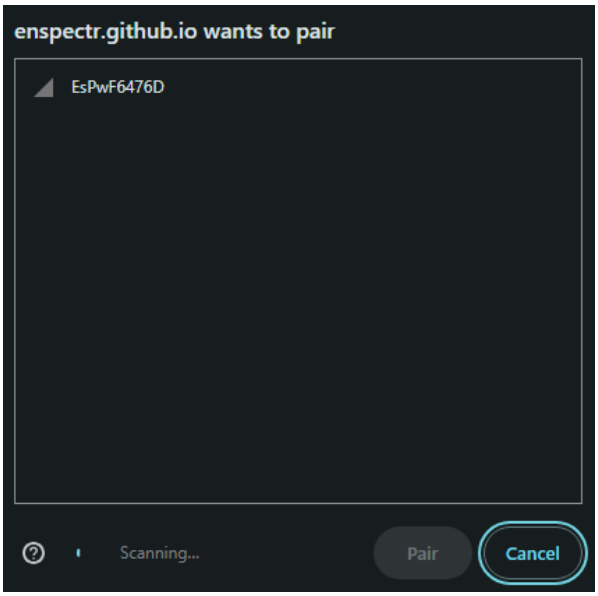

Figure 4. List of available Bluetooth devices.

After selecting the desired device, the connection will be established, and the telemetry information will be displayed on the page.

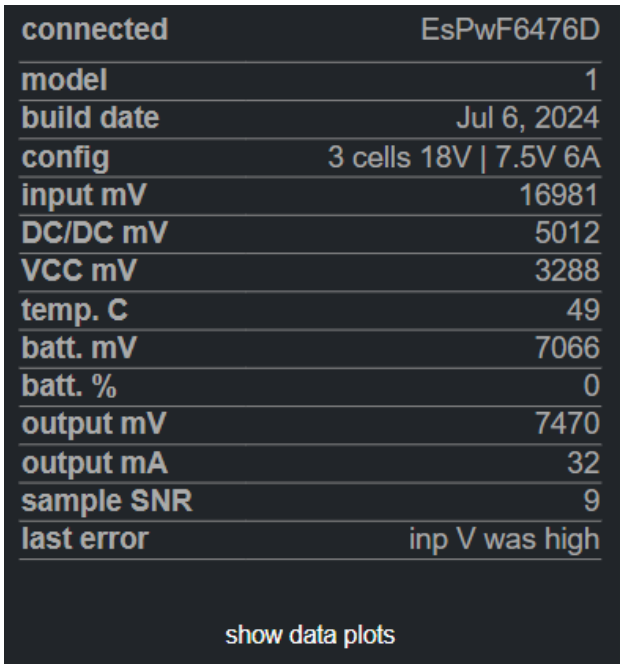

Figure 5. Parameters display.

3.2. Monitoring main parameters

At the top of the application page, the device name and its current connection status are displayed. Below that, the device parameters, which update in real-time, are shown. The list of parameters is provided in the table:

Table 1. List of parameters

model	Device model
build date	Device firmware date
config	Device configuration
input mV	Input voltage in millivolts
DC/DC mV	DC/DC converter voltage in millivolts
VCC mV	Power line voltage (VCC) in millivolts
temp. C	Device temperature in degrees Celsius
batt. mV	Battery voltage in millivolts
batt. %	Battery charge in percentage
output mV	Output voltage in millivolts
output mA	Output current in milliamps
sample SNR	Sample presence sensor signal
last error	Last encountered error

At the very bottom, there is a "show data plots" button to display data graphs.

3.3. Displaying parameters graphs

The application allows visualizing changes in device parameters over time.

Clicking the “show data plots” button opens the display of parameter graphs over time.

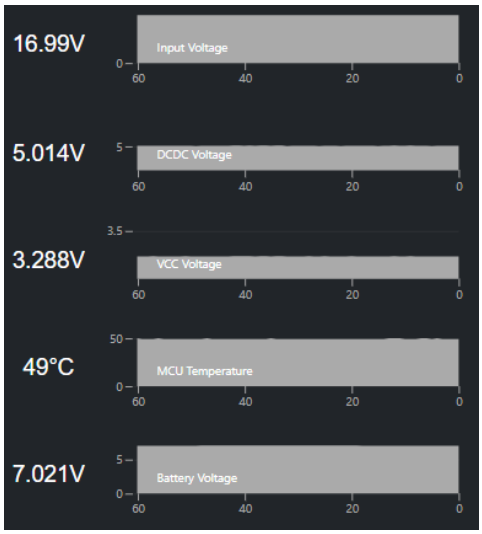

Figure 6. Displaying graphs.

At the bottom of the page, there are buttons for adjusting the data display on the graphs.

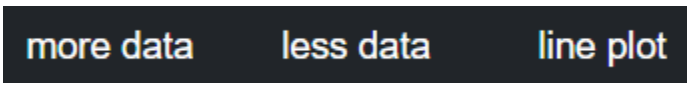

Figure 7. Graph adjustment buttons.

The "more data" and "less data" buttons increase and decrease the time scale by a factor of 2. The third button changes the graph display type. In line plot mode, the vertical scale is always chosen automatically to reflect the range of data changes. This mode provides more detailed information about data changes over time.

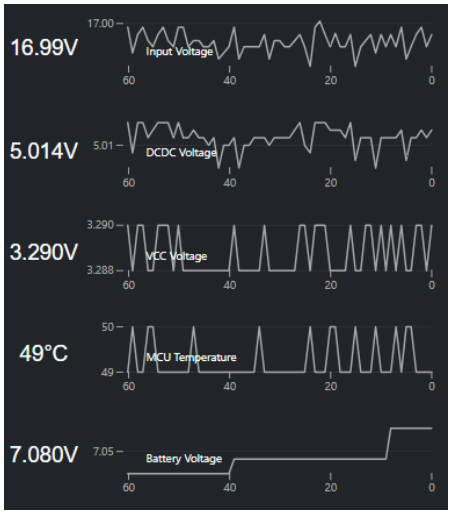

Figure 8. Line plot graphs.

3.4. Saving parameter dependencies over time to disk

The application supports saving data as a file. Clicking on a parameter value to the left of the graph saves its data to a file, where the first column is time, and the second column is the parameter values.

3.5. Error handling

The web application processes device errors by decoding their codes and displaying user-friendly messages. When an error occurs, the application shows it as a short name:

Table 2. Known errors

Error name	Description
vcc	Power line voltage
temp	Device temperature
batt V	Battery voltage
inp V	Input voltage
out V	Output voltage
out I	Output current
out failure	Output converter failure
dcdc V	DC/DC converter voltage

If the error is unknown, the application displays its code received from the device. The application also indicates whether the error occurred due to a parameter exceeding its upper or lower limit (“high” or “low”). If the error was active within the last 60 seconds, it is marked as “is high” or “is low”, and if it was active earlier, it is marked as “was high” or “was low”.

When an error was active within the last 60 seconds, all values of the variable that caused it are displayed in red on the page.

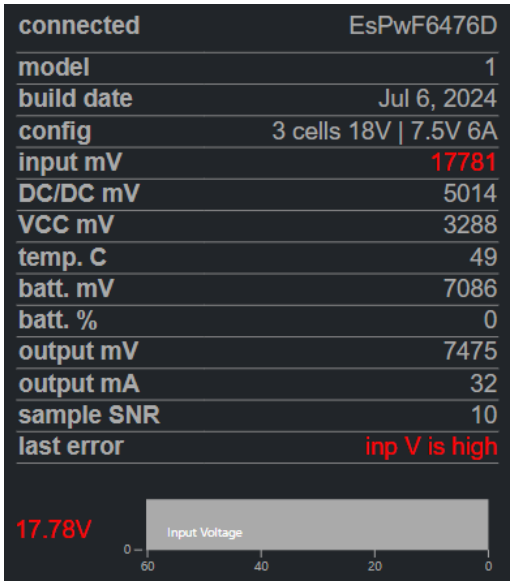

Figure 9. Example of error display in the application.

4. Application specifics on different platforms

In this section we will consider general guidelines for using the application on various platforms. More detailed information about application's compatibility with various browsers and platforms can be found at the following link [Web Bluetooth Implementation Status] .

4.1. Application specifics on the Windows platform

For optimal performance on Windows, it is recommended to use the Google Chrome browser. It provides the best compatibility with the application. Ensure that Bluetooth is enabled on the device.

4.2. Application specifics on the Linux platform

For Linux OS, it is recommended to use the Google Chrome or Chromium browser. In the browser, a setting must be enabled to allow the web application to connect to the device. To do this, navigate to the following address and set flag to Enabled:

`chrome://flags/#enable-experimental-web-platform-features`

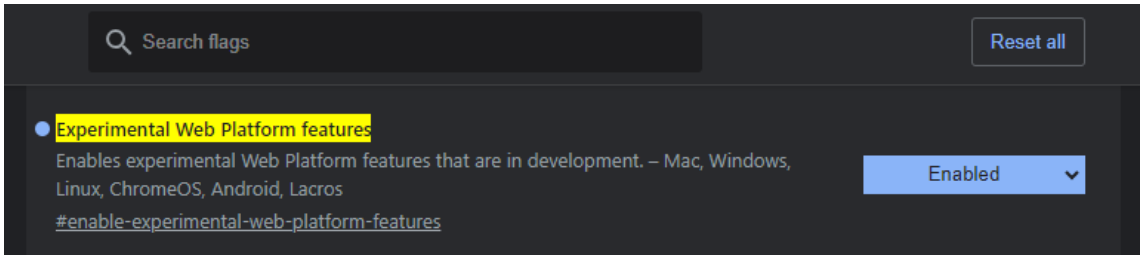

Figure 10. Experimental web platform features setting

To apply the change, restart the browser.

After this, in modern Linux distributions, the web application should work. However, if it does not, ensure that system has the Linux kernel version 3.19 or higher and the Bluez component version 5.41 or higher.

4.3. Application specifics on the Android platform

For Android versions 11 and below, the application should be run in the Google Chrome browser with location permission enabled. For Android 12 and above, the "Nearby Devices" permission needs to be enabled.

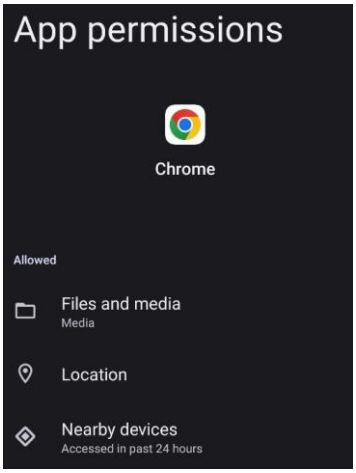

Figure 11. Chrome browser permission settings.

4.4. Application specifics on the iOS platform

For iOS devices, standard browsers like Google Chrome and Safari cannot be used because they do not support Bluetooth functionality. Therefore, it is necessary to use the Bluefy web browser, which supports the Bluetooth API. This browser can be found and downloaded from the AppStore.

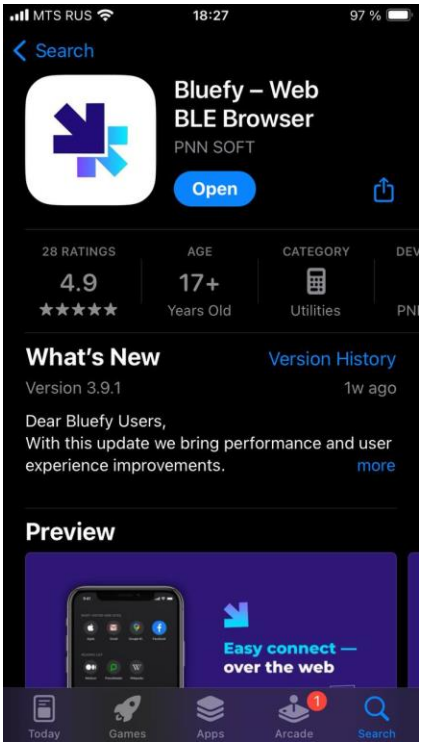

Figure 12. Bluefy page in the AppStore.

Ensure that the Bluetooth permission is enabled.

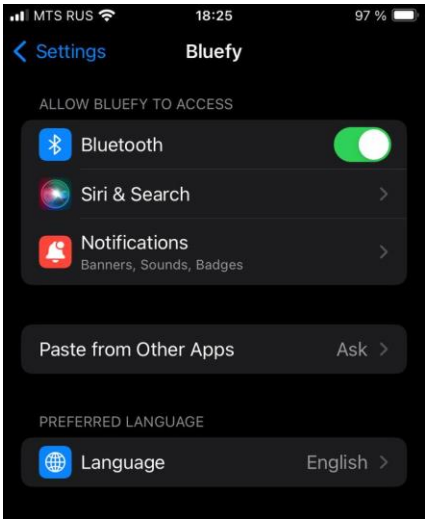

Figure 13. Bluefy settings.