1. Introduction

This system is designed to perform galvanostatic charge and discharge tests of supercapacitors, supporting target currents from –1 A to +1 A under a maximum operating voltage of 5 V.

Main components:

- STM32F446RE board with custom shield PCB installed (Fig. 1)
- ±12 V power suppl
- MAC/Windows PC

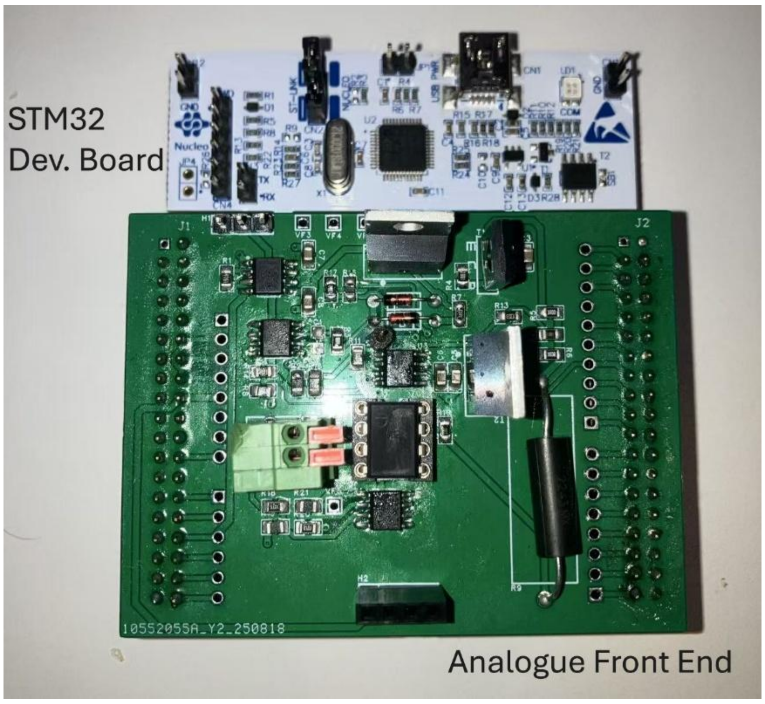


Fig. 1. STM32 development board with custom analogue front-end PCB installed

2. Firmware Installation

Requirements:

- STM32CubeIDE (compatible with macOS and Windows)
- SuperCapDAC project files

Flashing Procedure

1. Open STM32CubeIDE → File → Open Project from File System → Directory → select the extracted SuperCapDAC folder → Finish.
2. In the Project Explorer, expand SuperCapDAC, navigate to Core → Src → main.c, and open the file.
3. Click Build 🔨. The build should finish with 0 errors and 0 warnings.

4. Connect the STM32 board to the Mac/PC via USB (ST-LINK interface). Click Run . The LED on the development board should start blinking, and the STM32CubeIDE terminal should indicate successful flashing.

3. Software Installation (Host)

Requirements:

- Python 3.12 or later (Windows/macOS)
- Provided program: FinalHost.py
- Required packages: pyserial, matplotlib, numpy, pandas

Installation:

1. Install Python from the official website (https://www.python.org).
2. Open a terminal (Command Prompt or PowerShell on Windows, Terminal on macOS).
3. Install the required Python packages using the command:
4. pip install pyserial matplotlib numpy pandas
5. Place FinalHost.py in a working directory of your choice.

Port Identification:

1. Connect the STM32 board to the PC via USB.

2.a) On Windows:

1. Open *Device Manager*.
2. Under *Ports (COM)*, identify the entry labeled *STMicroelectronics STLink* Virtual COM Port.
3. Note the COM port number (e.g., *COM5*).
4. In the host software script (FinalHost.py), modify line 14: *PORT = "COM4"*. Replace *"COM4"* with the actual COM port number found in Device Manager (e.g., *"COM5"*).

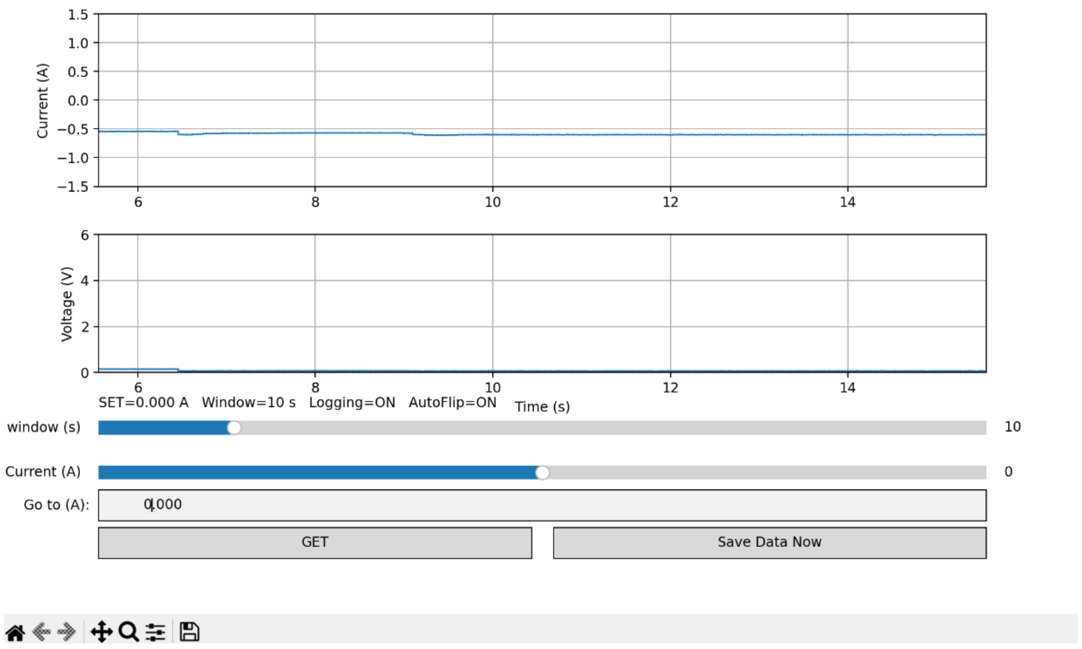
2.b) On macOS:

1. Open Terminal.
2. Run the command: *ls /dev/tty.**
3. Look for a device name similar to: */dev/tty.usbmodemXXXX*
4. In the host software script (FinalHost.py), modify line 14: *PORT = "COM4"*. Replace *"COM4"* with the full device path found in Terminal (e.g., *"/dev/tty.usbmodem1103"*).

Run the program with:

python FinalHost.py

The host program will connect to the STM32 board and display the user interface (Fig.2) for controlling the charge/discharge process and logging results.



SET=0.000 A Window=10 s Logging=ON AutoFlip=ON

Fig. 2. Host software interface showing real-time current and voltage plots with control panel

4. Operation Procedure

Important Safety Note:

Always ensure the ±12 V supply is switched **OFF** before making or changing any connections. Incorrect wiring or powering sequence may cause permanent hardware damage.

1. Connect the power pins to the board according to the pin definition in Fig 3.
2. Run **FinalHost.py** on the PC. Move the current setpoint slider fully to the left (−1 A). This ensures that the shield board does not unintentionally charge the capacitor when powered on.
3. Connect the device under test (DUT):

   - Capacitor positive terminal → **CAP+**
   - Capacitor negative terminal → **CAP−**

4. Double-check the power pin connections and capacitor polarity. Once verified, switch on the ±12 V supply.
5. In the FinalHost.py GUI, type in the desired current value in the text bar *"Go to (A): "*
    - Positive values → charging
    - Negative values → discharging

   Press *Enter* to apply the new current setting.
   After reaching 5 V, the system will automatically reverse the current to discharge it back to 0 V, completing one full charge/discharge cycle.
   Keyboard shortcuts:
    - ↑ / ↓ : increase or decrease by 0.1 A
    - → / ← : increase or decrease by 0.01 A
6. To perform multiple tests, repeat Step 5 with new current values.
7. Observe the capacitor voltage and current in the *Current (A)* and *Voltage (V)* panels. Use the *Window (s)* slider to zoom in/out on the time axis for a full view of the cycle.
8. Save the data:
   Click *Save Data Now* to immediately save the current dataset as a CSV file.
   Exiting the host program will also automatically save one dataset. The file path will be displayed in the Python terminal.

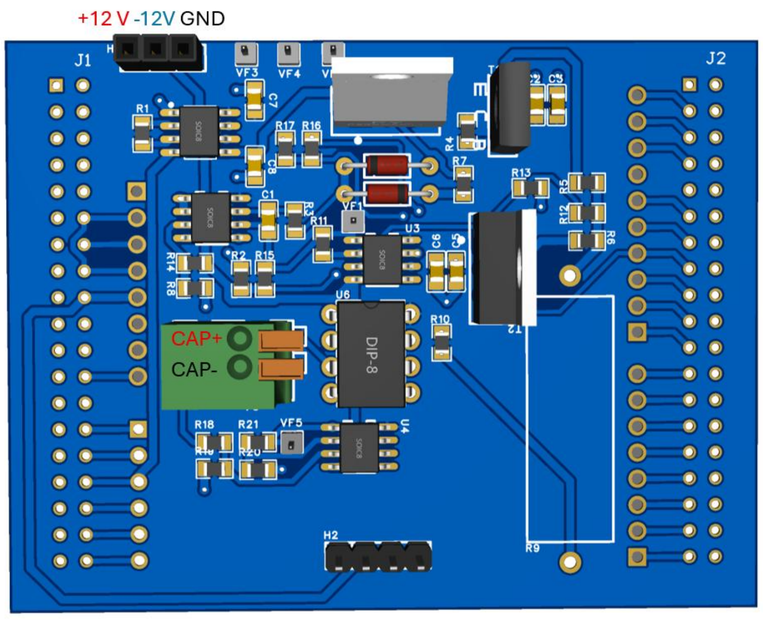


Fig. 3. PCB 3D view showing power supply and capacitor connection pins

Known Issues

1. **Maximum Current Limitation**

Although the system and host software are designed for a maximum current of ±1 A, the current hardware implementation can only operate reliably and stably up to ±0.5 A.

This limitation is caused by the base drive of the BJT power stage. Reducing the base resistor value or replacing the BJTs with Darlington transistors may solve the problem.

2. **Text Bar and Time Window Interaction**

   If the text bar (“Go to (A):”) contains a target current value different from the one currently applied, adjusting the time window slider will force this change to take effect.