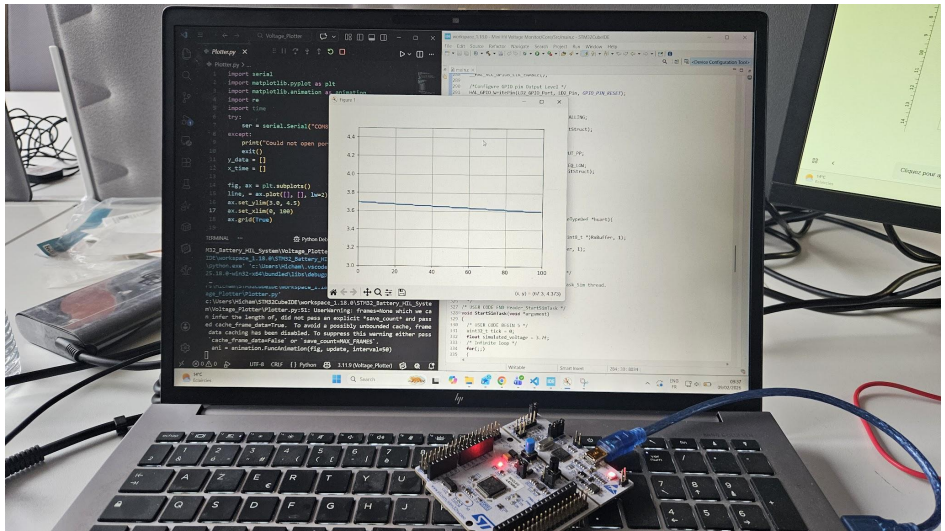
Hardware-in-the-Loop Battery Simulator

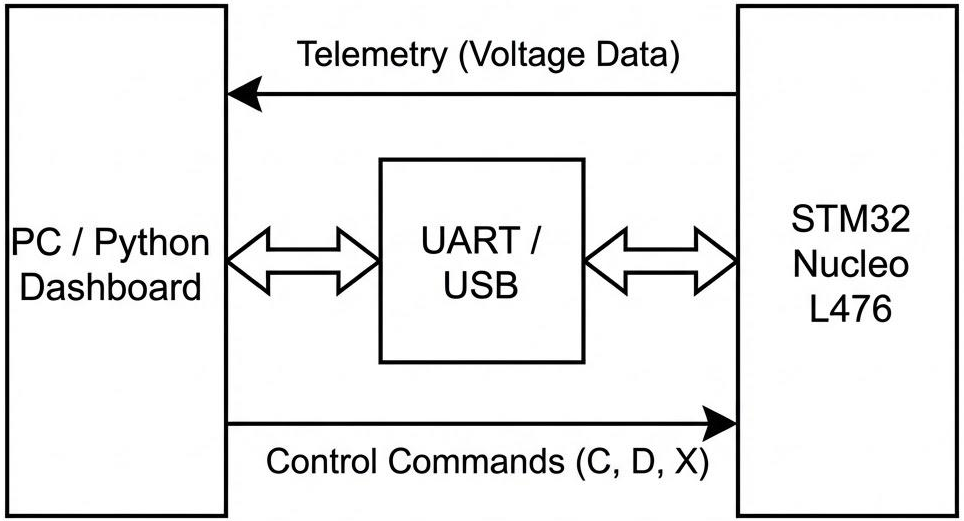
Real-time BMS Testing with STM32 & Python



Why Simulate a Battery?

- Testing Battery Management Systems (BMS) with real Li-Ion batteries is dangerous (fire risk) and slow (charging takes hours).
- **The Solution:** A HIL Simulator that mimics battery physics in real-time.
- **Benefit:** Allows for safe "Fault Injection" (e.g., short circuits) and instant charging/discharging scenarios.

High-Level Architecture



Real-Time Embedded Design (FreeRTOS)

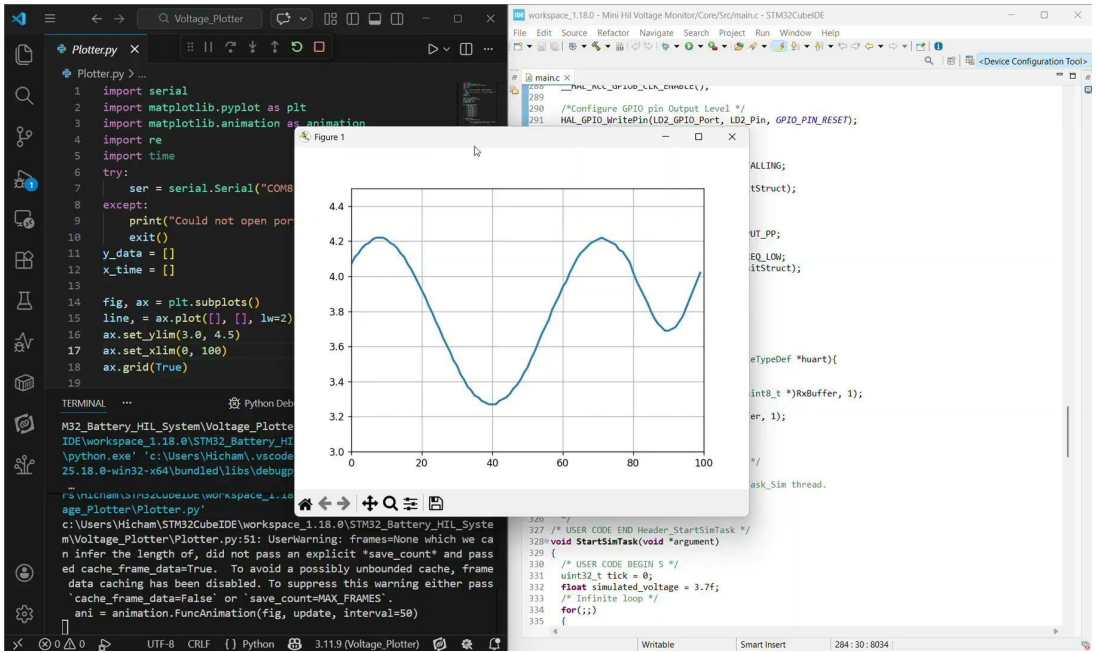
- **Task 1: Simulation (High Priority):** Calculates physics ($V = 3.7 + \sin(t)$ or $V_{new} = V_{old} - Load$).
- **Task 2: Filter (Normal Priority):** Implements a Moving Average Filter to remove sensor noise and checks safety limits.
- **Task 3: Communication (Low Priority):** Formats and transmits JSON/String data to the PC.

Dashboard & Control Station

- **Data Visualization:** Uses `matplotlib` for 10Hz real-time plotting.
- **Parsing:** Regex-based engine to extract reliable data from serial streams.
- **Event Handling:** Asynchronous keyboard listener for user control.

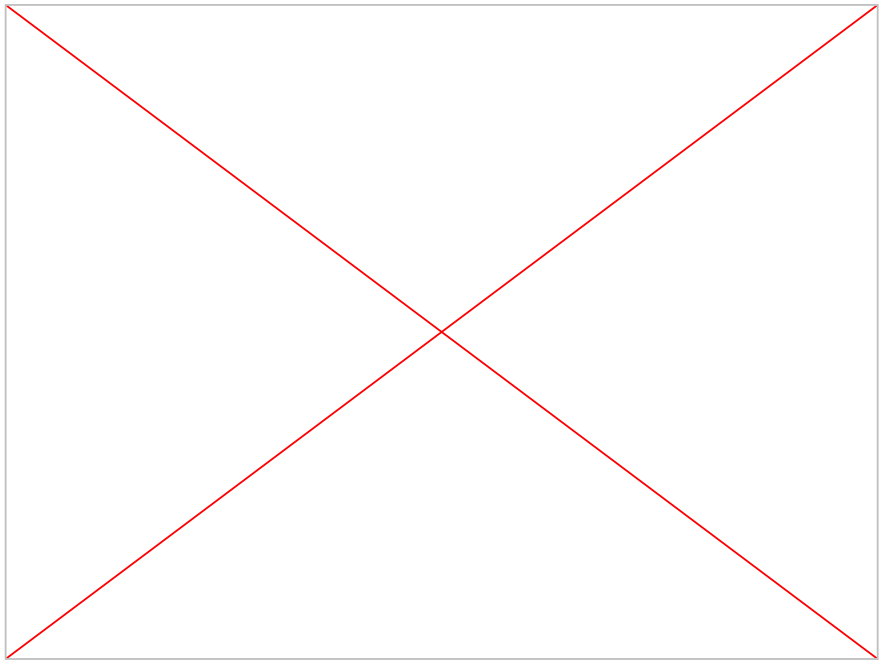
Interactive Control

when pressing ‘c’ the battery starts charging



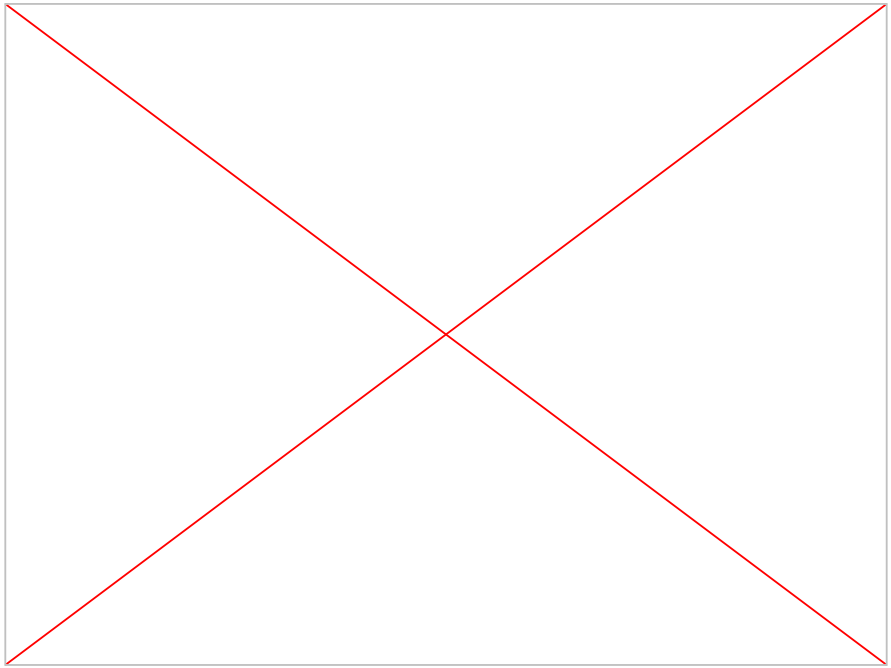
Safety Critical Testing

when pressing ‘x’ the system simulate a “cut wire” and the system alarm LED turns on in under 10 ms.



Realistic Discharge Curves

when pressing ‘e’ Simulating long-term battery drain to test the 'Low Battery' warning logic without waiting hours for a real battery to empty.



Engineering Challenges

- **Challenge:** The STM32 L476RG clock conflicted with FreeRTOS.
- **Solution:** Migrated the HAL Timebase to TIM6 to prevent SysTick collisions.
- **Challenge:** Serial data parsing was unreliable (partial strings).
- **Solution:** Implemented Regex filtering and a circular buffer.

Summary

- Successfully built a full-stack HIL test bench.
- Demonstrated RTOS, UART, and Python driver development.
- link:

  https://github.com/HichamBouzalmad/STM32_Battery_HIL_System/tree/master