

Host-Device Interaction via Wi-Fi (HTTP Control on Linux/macOS)

This guide shows you how to use **HTTP commands** to wirelessly control your robot driver board from a **Linux or macOS** computer.

The driver board includes a built-in lightweight HTTP server, allowing you to send JSON commands directly over Wi-Fi — no special drivers or tools required.

1. Download the Project

Clone the example project from GitHub:

```
git clone https://github.com/EffectsMachine/robot_driver_with_esp32s3_lite.git  
  
cd robot_driver_with_esp32s3_lite
```

💡 Tip:

If Git is not installed, you can install it with:

```
sudo apt install git    # Ubuntu / Debian  
brew install git        # macOS (Homebrew)
```

The repository includes several host-side control examples.

In this tutorial, we'll focus on the **HTTP example**.

2. About HTTP Control

Your robot driver board runs a tiny built-in HTTP server (default port **80**).

You can control it directly through your browser or by sending HTTP requests from a Python script.

Why HTTP?

Advantage	Description
Cross-platform	Works seamlessly on Linux, macOS, Windows, Android, and iOS.
Developer-friendly	Every modern language (Python, C++, JavaScript, etc.) supports HTTP libraries.
Readable & structured	Commands use standard JSON, making them easy to debug and expand.
Scalable	Can evolve into HTTPS-secured or RESTful API systems for cloud or web control.

⚠ Note:

HTTP is not designed for real-time control.

It's ideal for **~40 commands/sec** operation such as movement control, parameter updates, or data queries —

not for millisecond-level trajectory or servo synchronization.

3. How It Works

When powered on, the driver board launches an **HTTP server on port 80** and connects to a Wi-Fi network in one of two modes:

1. Access Point (AP Mode)

- The board creates its own Wi-Fi hotspot named **Robot**.
- Connect your PC to this hotspot and access:

```
http://192.168.4.1:80
```

2. Station (STA Mode)

- The board connects to your existing router.

- The assigned IP address is shown on the OLED display, e.g.:

```
http://192.168.0.105:80
```

Once connected, you can send a JSON command via HTTP **POST** :

```
{"T":202, "line":1, "text":"http cmd test", "update":1}
```

No drivers or special software are needed — any networked device (laptop, Raspberry Pi, or server) can communicate instantly.

4. Setting Up the Python Environment

4.1 Check Python Installation

Most Linux and macOS systems already include Python. Verify the version:

```
python3 --version
```

If it's below **3.10**, install or update it:

- **Ubuntu / Debian**

```
sudo apt update  
sudo apt install python3 python3-pip python3-venv
```

- **macOS (Homebrew)**

```
brew install python3
```

4.2 Create a Virtual Environment

Navigate to the example folder:

```
cd "Example for Robot Driver Lite/python_example/http"
```

Then create and activate a Python virtual environment:

```
python3 -m venv venv  
source venv/bin/activate
```

Install dependencies:

```
pip install -r requirements.txt
```

This keeps your workspace clean and isolated from other projects.

5. Edit and Run the Example

Open the example file:

```
nano http_example.py
```

Locate this line and set your board's IP address:

```
ESP32_IP = "192.168.0.105"
```

If you're connected directly to the board's hotspot, use:

```
ESP32_IP = "192.168.4.1"
```

Save and exit.

Run the Script

Execute:

```
python3 http_example.py
```

If communication succeeds, you'll see:

```
status: 200
feedback: OK
```

This confirms the board received and executed your JSON command.

6. Inside the Example

Here's the simplified version of `http_example.py`:

```
import requests, json

ESP32_IP = "192.168.0.105"
PORT = 80
URL = f"http://{ESP32_IP}:{PORT}/api/cmd"

def send_json_command(payload: dict):
    headers = {"Content-Type": "application/json"}
    try:
        response = requests.post(URL, headers=headers, data=json.dumps(payload), timeout=2)
        print(f"status: {response.status_code}")
        print(f"feedback: {response.text}")
    except requests.exceptions.RequestException as e:
        print(f"exception: {e}")

if __name__ == "__main__":
    data = {"T":202, "line":1, "text":"http cmd test", "update":1}
    send_json_command(data)
```

You can easily modify `data` to send other JSON commands supported by your robot firmware.

7. Troubleshooting

Issue	Possible Cause	Solution
ConnectionError	Board not connected or incorrect IP	Check Wi-Fi connection and IP shown on OLED
Timeout	Weak signal or blocked port	Reboot board or disable firewall temporarily
404 Not Found	Invalid endpoint path	Ensure firmware version supports <code>/api/cmd</code>

Quick Recap

Controlling your robot board from Linux or macOS takes only **three steps**:

1. Connect the board to Wi-Fi and note its IP address
2. Edit `http_example.py` with that IP
3. Run the script to send JSON commands

Your robot driver board now acts as a **mini RESTful robot server** — ready for automation scripts, cloud control, or web-based interfaces.