

Introduction to the Automatic Polariser Guide

This guide provides detailed instructions on how to build and operate an open-source automatic polariser for acquiring spectra with incrementally changing polarisation angles. Designed for compatibility with Bruker Hyperion microscopes (and adaptable for other instruments), this low-cost, customizable device uses a stepper motor controlled by an ESP32 microcontroller.

What is needed:

To build the automatic polariser, you will need the following components:

1. ESP32 Microcontroller
2. Stepper Motor (BYJ28-64)
3. ULN2003 Driver Board
4. Endstop Switch
5. USB Power Supply (3.3V and 5V)
6. Wires for connections (if not using soldered connections)
7. 3D-printed components (for housing the polariser) –

How to Build the Hardware

1. **Wiring the Components:** Follow the wiring diagram included in this guide to connect the ESP32, stepper motor, ULN2003 driver board, endstop switch, and optional OLED display. Ensure that all connections are secure and correctly aligned with the respective pins on the microcontroller.
2. **3D Printing:** Print the necessary components for housing the stepper motor. You can find the 3D design files in the supplementary materials. Assemble the printed parts to create a sturdy and functional enclosure for the polariser.
3. **Final Assembly:** Integrate all components into the enclosure, ensuring that the motor can rotate freely and that the endstop switch is properly positioned to detect when the polariser reaches its limit.

List of Needed Components

1. ESP32 Development Board

- **Description:** Microcontroller with WiFi and Bluetooth capabilities.
- **Estimated Price:** €5 - €10

2. ULN2003 Stepper Motor Driver Module

- **Description:** A driver circuit for controlling stepper motors.
- **Estimated Price:** €1 - €3

3. BYJ28-64 Stepper Motor

- **Description:** A small 4-phase stepper motor commonly used in robotics and automation.
- **Estimated Price:** €3 - €5

4. Endstop Switch (Mechanical or Optical)

- **Description:** A switch used to detect the position of the moving part.
- **Estimated Price:** €0.50 - €2

5. 0.96-inch OLED Display (optional)

- **Description:** A small display module for showing information.
- **Estimated Price:** €2 - €5

6. USB Power Supply (3.3V and 5V)

- **Description:** Provides power to the ESP32 and connected components.
- **Estimated Price:** €5 - €10

7. Connecting Wires

- **Description:** Jumper wires for making connections.
- **Estimated Price:** €1 - €3

Total Estimated Cost: €16 - €33

Wiring Guide for ESP32, UNL2003, BYJ28-64, Endstop Switch, and Optional OLED Display

Components Required

- **ESP32 Development Board**
- **UNL2003 Stepper Motor Driver**
- **BYJ28-64 Stepper Motor**
- **Endstop Switch**
- **USB Power Supply (3.3V and 5V outputs)**

1. ESP32 Pin Connections

The ESP32 will control the UNL2003 and read input from the endstop switch. Below are the recommended pin connections:

- **GPIO 21** → IN1 on UNL2003
- **GPIO 19** → IN2 on UNL2003
- **GPIO 18** → IN3 on UNL2003
- **GPIO 5** → IN4 on UNL2003
- **GPIO 2** → Endstop Switch (StopPin)
- **GND** → GND on UNL2003 and the endstop switch
- **3.3V** → **3.3V Output** from the USB power supply

2. Stepper Motor Driver Connections (UNL2003)

Connect the control pins from the ESP32 to the UNL2003 driver as follows:

- **IN1** on UNL2003 → **GPIO 21** on ESP32
- **IN2** on UNL2003 → **GPIO 19** on ESP32
- **IN3** on UNL2003 → **GPIO 18** on ESP32
- **IN4** on UNL2003 → **GPIO 5** on ESP32

Connect the power:

- **VCC** on UNL2003 → **5V Output** from the USB power supply

- **GND** on UNL2003 → **GND** from the USB power supply and ESP32

3. Stepper Motor Connections (BYJ28-64)

The BYJ28-64 stepper motor typically connects directly to the UNL2003 driver:

- Connect the 4 wires of the BYJ28-64 to the output pins on the UNL2003 (usually labeled as OUT1, OUT2, OUT3, OUT4).

4. Endstop Switch Connection

The endstop switch detects when the motor reaches a certain position. Connect the switch as follows:

- **One terminal of the endstop switch** → **GPIO 2** on ESP32 (StopPin)
- **Other terminal of the endstop switch** → **GND** on ESP32

Final Notes

- Ensure that all connections are secure and correctly wired.
- This configuration does not require additional resistors for the endstop switch.

How to Upload Software Code to the ESP32

Requirements

1. **Arduino IDE** installed on your computer. If you don't have it, download it from the Arduino website.
2. **ESP32 Board Package** installed in the Arduino IDE.
3. **USB Cable** to connect the ESP32 to your computer.

Step 1: Install the ESP32 Board Package

1. Open the Arduino IDE.
2. Go to **File** → **Preferences**.
3. In the "Additional Board Manager URLs" field, add the following URL:
https://dl.espressif.com/dl/package_esp32_index.json
4. Click **OK** to save.

Step 2: Install the ESP32 Board

1. Go to **Tools** → **Board** → **Board Manager**.
2. In the search box, type "ESP32".
3. Find "ESP32 by Espressif Systems" and click **Install**.

Step 3: Select the ESP32 Board

1. Go to **Tools** → **Board** and select your specific ESP32 board (e.g., ESP32 Dev Module).

Step 4: Install Required Libraries

Step 5: Connect the ESP32 to Your Computer

1. Use a USB cable to connect the ESP32 to your computer.
2. Make sure the correct COM port is selected:
 - Go to **Tools** → **Port** and select the port corresponding to your ESP32.

Step 6: Upload the Code

1. Copy the code you want to upload (the complete code we discussed, including the setup for the stepper motor and OLED display).
2. Paste the code into the Arduino IDE.

3. Click the **Upload** button (right arrow icon) in the Arduino IDE toolbar.
4. Wait for the upload process to finish. You'll see messages in the bottom console area indicating the progress.

Step 7: Open the Serial Monitor (Optional)

1. After uploading, you can open the Serial Monitor to view debug messages.
2. Go to **Tools** → **Serial Monitor** or press **Ctrl + Shift + M**.
3. Set the baud rate to **115200** (or the rate you used in your code).

Step 8: Connect to the WiFi “ESP32_Access_Point” password = "123456789";

Step 9: Go to the page <http://192.168.4.1> or http://FTIR_Pol.local

Optional you can change the name of the WiFi and the password by editing the following lines:

```
const char *ssid = "ESP32_Access_Point";
```

```
const char *password = "123456789";
```

it is possible to change the address FTIR_Pol.local by editing the line: if
`(MDNS.begin("FTIR_Pol")) {...`