Range finder (EY09A) connection to Orange Pi 5 Plus

1) Setting up of orange pi 5 plus with Armbian OS :

Setting up your Orange Pi 5 Plus with Armbian OS generally involves flashing the Armbian image to an SD card (or directly to the eMMC), booting from it, and then performing some initial configuration.

Here are the detailed steps:

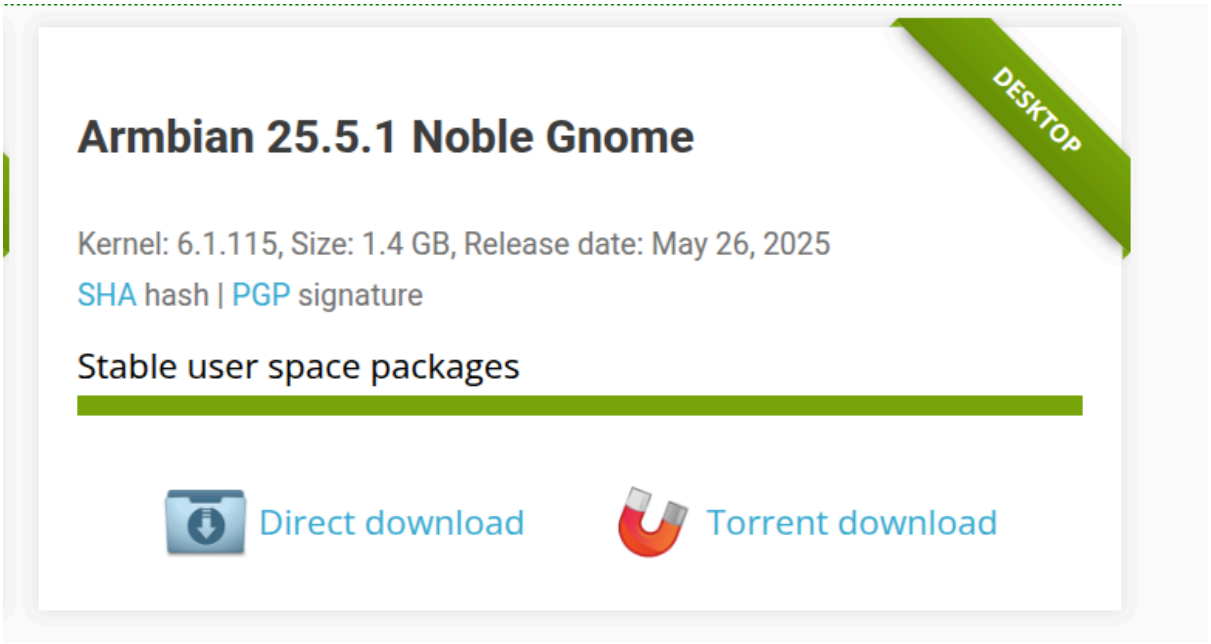
1. **Gather Your Materials** 🛠️
   - Orange Pi 5 Plus Board: Your single-board computer.
   - Reliable Power Supply: A high-quality USB-C power supply (5V/4A or as recommended by Orange Pi) is crucial for stability.
   - MicroSD Card (Class 10 / A1 rated or higher): At least 32GB is recommended for a smooth experience. A faster card will significantly improve performance.
   - Computer (Windows, macOS, or Linux): To download the image and flash the SD card.
   - USB MicroSD Card Reader: To connect your SD card to your computer.
   - HDMI Cable & Monitor
   - USB Keyboard & Mouse

- Ethernet Cable

---

2. Download the Armbian Image 📥

1. Visit the Official Armbian Website: Go to Armbian's download page.
2. Find Your Board: Navigate to the "Orange Pi" section and specifically look for "Orange Pi 5 Plus" images. It's important to select the *Plus* version, as images for the regular Orange Pi 5 may not work correctly. Armbian for orangepi5plus
3. Choose an Image:
    - Desktop (e.g., Gnome, XFCE): Includes a graphical desktop, suitable if you plan to use your Orange Pi like a mini-PC.
    - Stable: Most thoroughly tested, may have older kernels/packages.
    - Download the `.img.xz` file (or `.img.gz`).
4. Verify the Download (Optional but Recommended): Check the SHA256 checksum provided on the download page against your downloaded file to ensure integrity.



3. Flash the Armbian Image to MicroSD Card 💾

This process writes the downloaded Armbian operating system onto your microSD card, making it bootable.

1. Insert MicroSD Card: Insert your microSD card into your computer's card reader.
2. Use a Flashing Tool:
    - BalenaEtcher (Recommended for all OS):
        1. Download and install BalenaEtcher from balena.io/etcher.
        2. Open Etcher.
        3. Click "Flash from file" and select the downloaded Armbian `.img.xz` (or `.img.gz`) file.
        4. Click "Select target" and carefully choose your microSD card. Double-check this step to avoid accidentally formatting the wrong drive!
        5. Click "Flash!" and wait for the process to complete (it will verify the write).

4. Initial Boot and Setup 🚀

1. Insert SD Card: Gently insert the flashed microSD card into the Orange Pi 5 Plus's microSD card slot.
2. Connect Peripherals:
    - Connect the HDMI cable to your monitor.
    - Connect USB keyboard and mouse
    - Connect an Ethernet cable for network access.
3. Power On: Connect the power supply to the Orange Pi 5 Plus. It should automatically power on and begin the boot process.
4. First Boot Wizard:
    - Armbian will typically boot to a text-based wizard or a desktop environment.
    - If it's a CLI image, you'll be prompted to log in. The **default login** is usually:
        - **Username: root**
        - **Password: 1234**
        - You'll be immediately **prompted to change the root password and create a new regular user**. Do this for security!
    - If it's a desktop image, it might boot directly to the desktop or ask for initial setup on screen.
    - Follow the on-screen instructions to set up your keyboard layout, timezone, and create a non-root user.

5. System Updates & Cleanup 🧹

Once logged in, it's crucial to update your system to ensure you have the latest security patches and software.

1. Open a Terminal: If you're on the desktop, find the terminal application.
2. Update Package Lists: **sudo apt update**
3. Upgrade Installed Packages: This will download and install updates for all installed software : **sudo apt upgrade**
    - Follow any prompts (e.g., to confirm package installations).

4. Clean Up (Optional but Recommended):
5. **sudo apt autoremove**
6. **sudo apt clean**

    These commands remove unnecessary packages and clear the package cache, freeing up space.

---

6. Install to eMMC (Optional, but Recommended for Performance) ⚡

The Orange Pi 5 Plus has an onboard eMMC module, which offers significantly faster storage performance and better reliability than an SD card. It's highly recommended to install Armbian to the eMMC once you've confirmed it boots and runs well from the SD card.

1. Boot from SD Card: Ensure you are booted into Armbian from the microSD card.
2. Run `armbian-install`: This is a powerful Armbian utility for moving the OS to different storage- **`sudo armbian-install`**
3. Follow the Prompts:
    - Select the option to install to eMMC.
    - Confirm the target device (`/dev/mmcblkX`, typically `/dev/mmcblk0` for eMMC). Be extremely careful here not to select your SD card or any other connected storage device by mistake.
    - Confirm filesystem and partition options (default usually works).
    - The process will copy the entire system to the eMMC. This can take some time.
4. Shutdown and Remove SD Card: Once the installation to eMMC is complete and confirmed, power down your Orange Pi: **`sudo poweroff`**

    - Remove the microSD card.
5. Reboot: Power on your Orange Pi 5 Plus. It should now boot directly from the eMMC.

---

7. Additional Important Configuration ⚙️

armbian-config: This is another useful Armbian tool for various system configurations, including network settings, software installations, and system services (also enabling the **UART** ports)

sudo armbian-config

SSH: If you want to manage your board remotely, ensure SSH is enabled and working:

```
sudo systemctl enable ssh
sudo systemctl start ssh
```

To check ssh status :

```
sudo systemctl status ssh
```

You can then connect from another computer using

```
ssh your_username@your_orange_pi_ip.
```

Install Desktop Environment (if starting with CLI): If you initially chose a CLI (command-line-interface) image but later want a desktop:

```
sudo apt install task-gnome-desktop # For GNOME
# or
sudo apt install task-xfce-desktop # For XFCE
```

Then reboot.

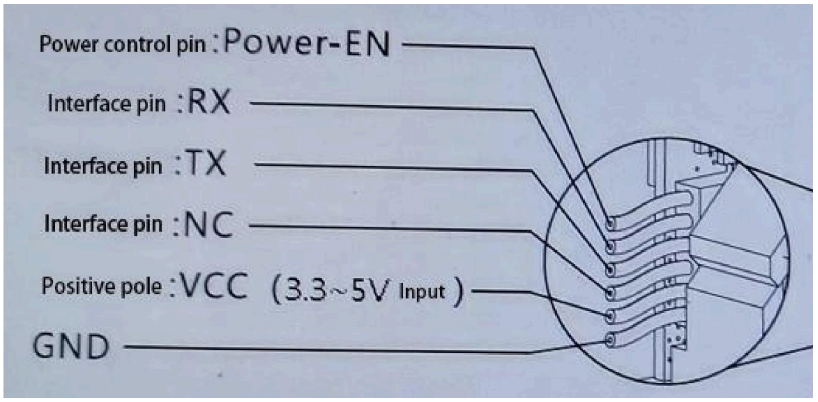
Till here all the basic setup is done Now to install important packages :

```
sudo apt install git
```

```
sudo apt install python3-pip
```

8.Enabling UART in orangepi 5 plus :

Wiring diagram for RF:



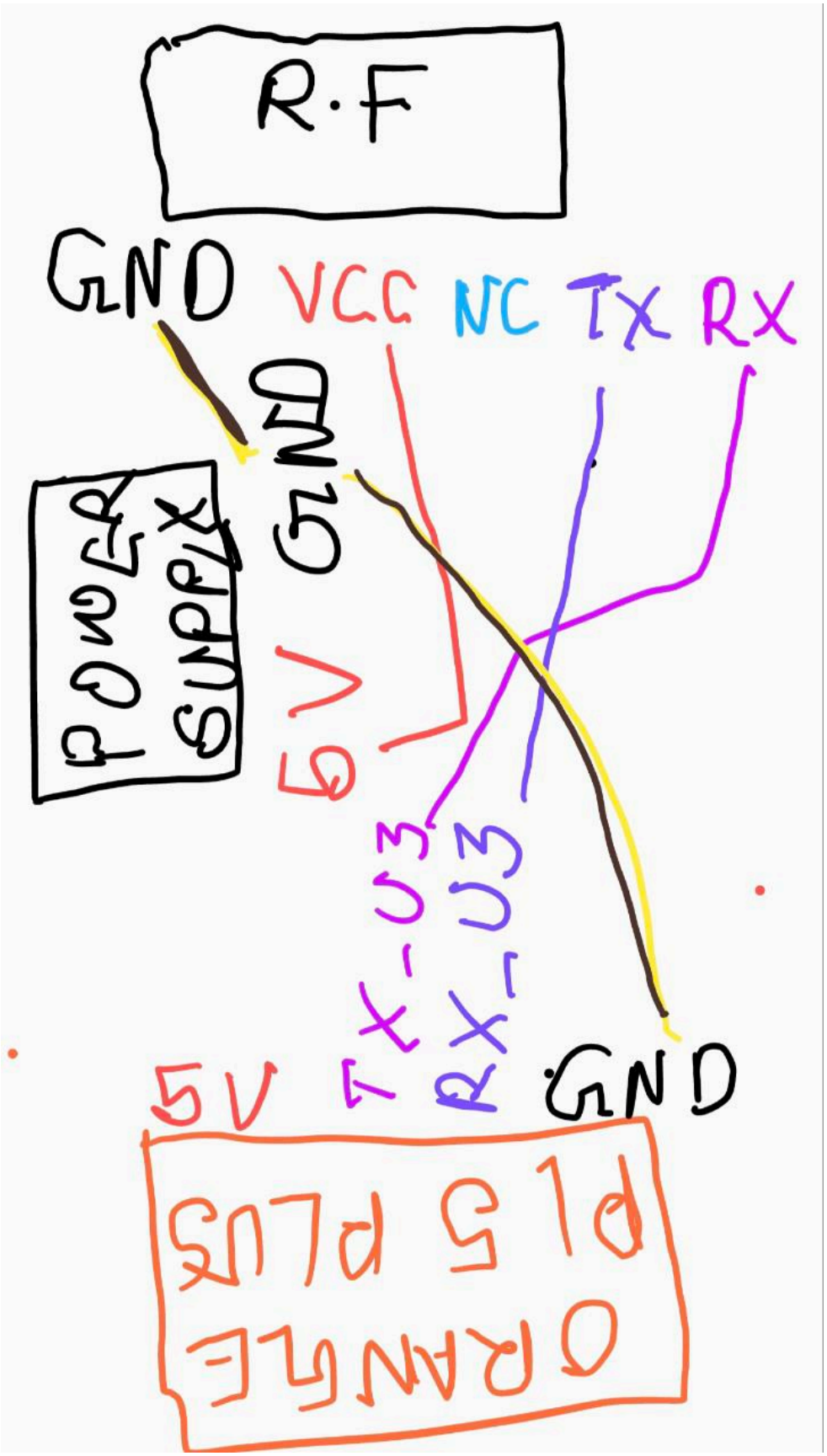
- **The RX pin shown in the figure above is connected to TX_U3**
- **The TX pin shown in the figure above is connected to RX_U3**
- **The VCC pin should be powered externally**

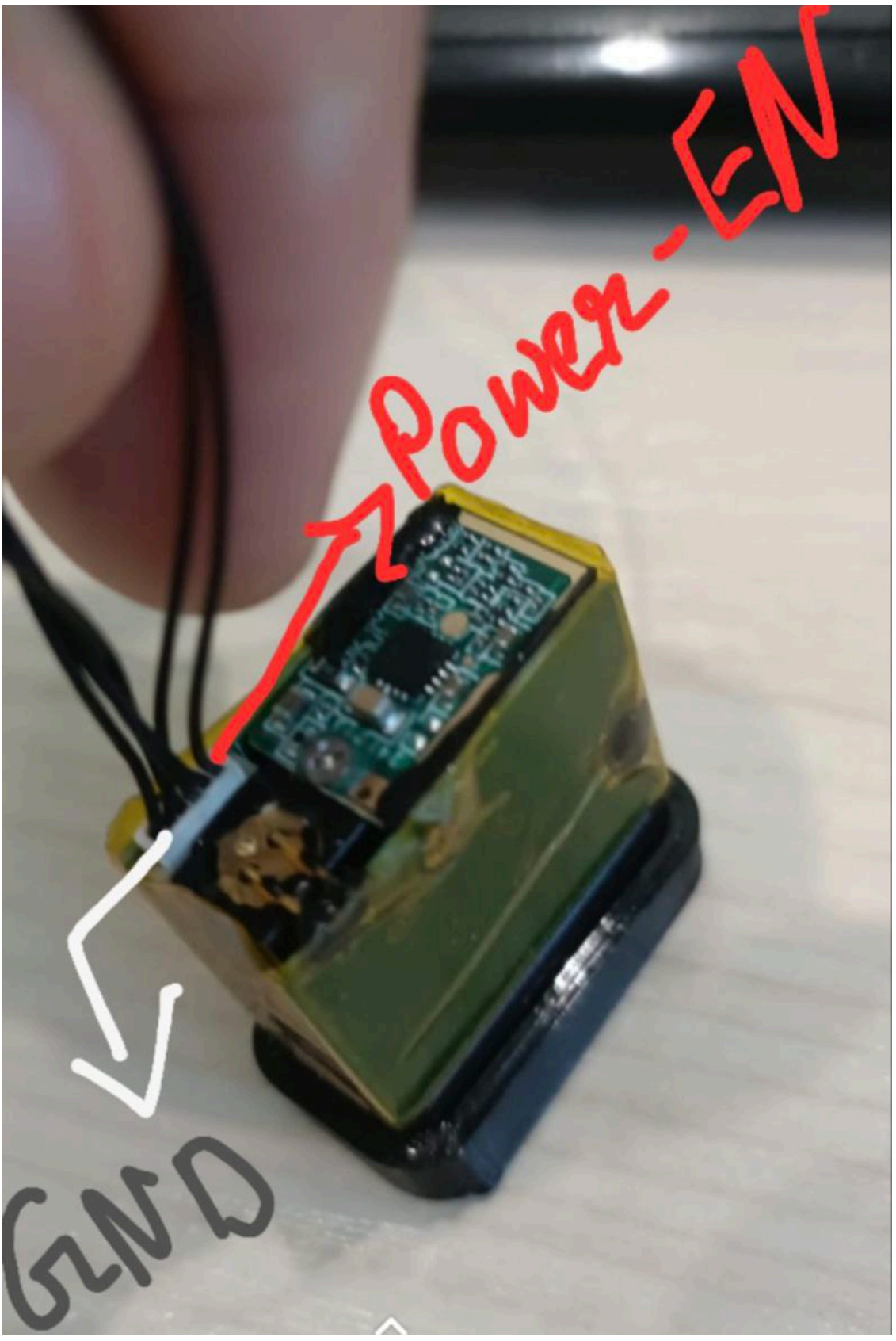
  (other than using Orange pi 5V pins)

  (preferably using UBEC (5V 3A))

- **The GND pin should be connected to both external power supply ‘s GND and orange pi’s GND pin**

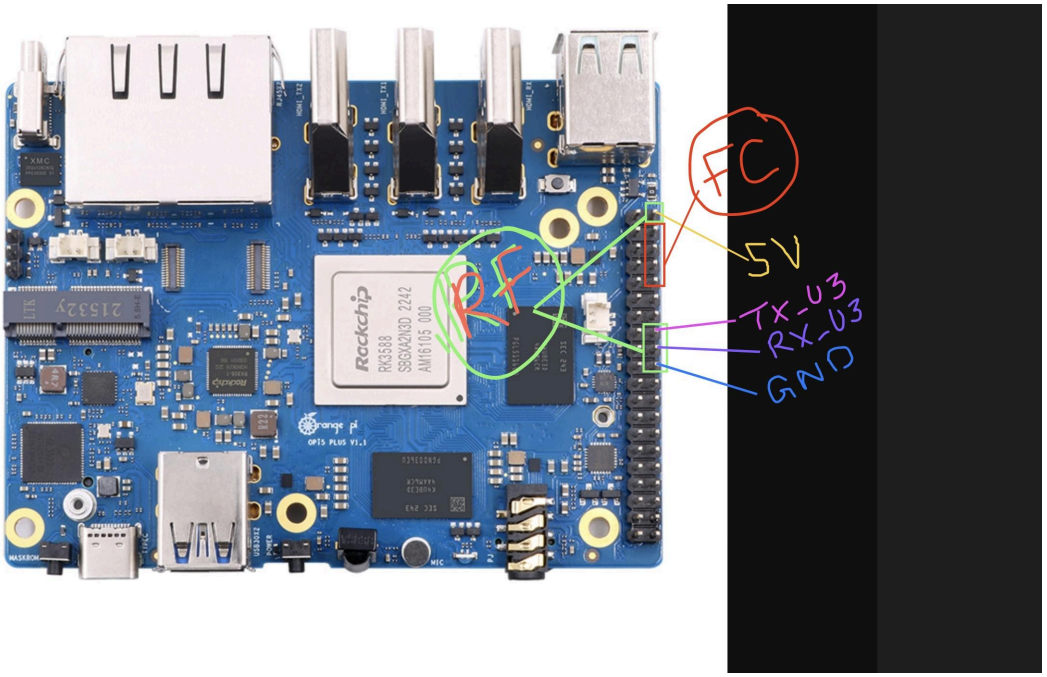
Full Circuit diagram for orange pi 5 plus + power supply (5V,3A) + Range finder (R.F)

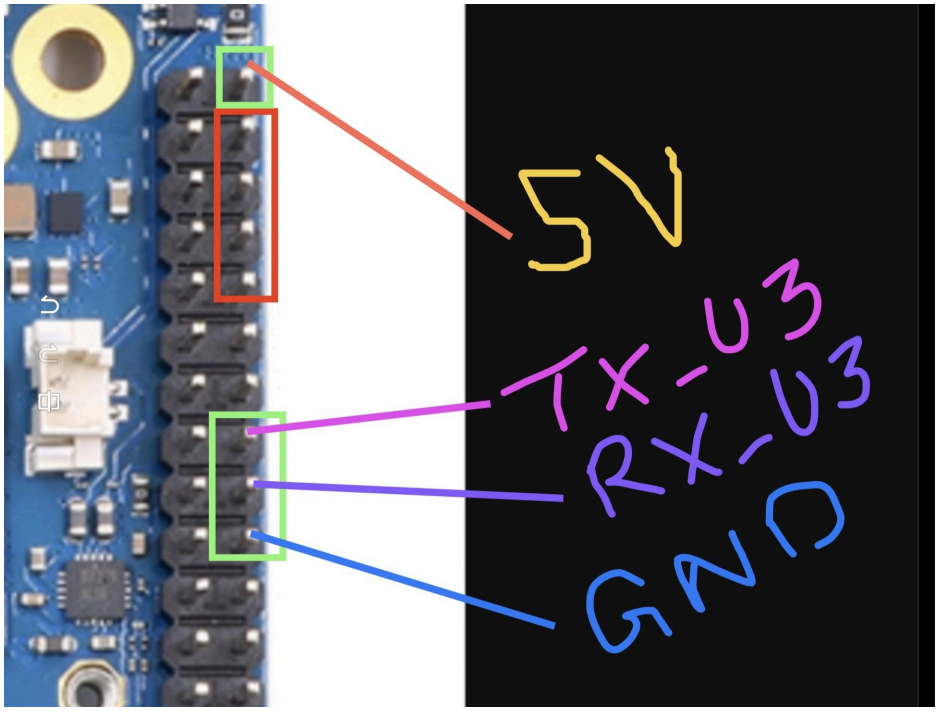

R.F
GND
VCC
NC
TX
RX
POWER SUPPLY
GND
5V
TX-U3
RX-U3
5V
GND
ORANGE PI 5 PLUS


Power-EN
GND

Power-EN
TX-U3
RX-U3
N.C
5V
GND

Orange pi 5 plus pinout





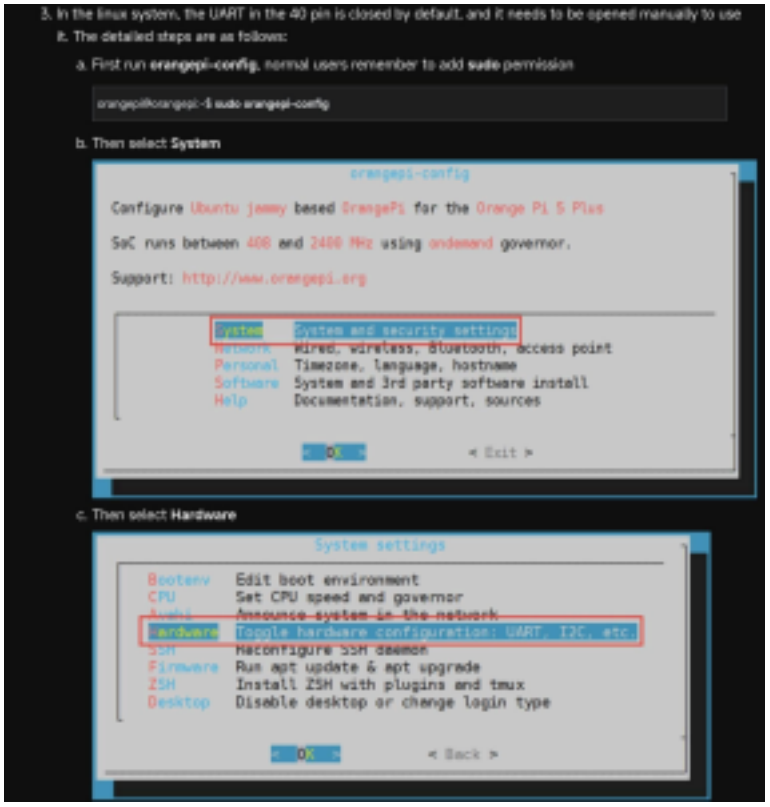
1. As can be seen from the table below, the uarts available for Orange Pi 5 Plus are uart1, uart3, uart4, uart6, uart7 and uart8, a total of 6 sets of uart buses

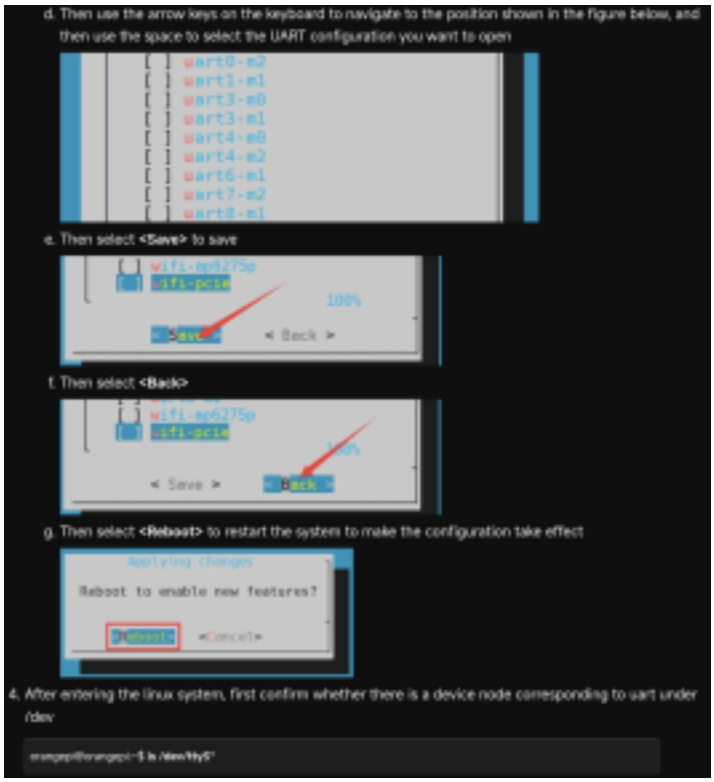
复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能	复用功能
			3.3V		1	2		5V				
CAN0_RX_M0	PWM1_M0 (fd8b0010)	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V				
CAN0_TX_M0	PWM0_M0 (fd8b0000)	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND				
UART1_RTSN_M1	I2C8_SCL_M2	PWM14_M2 (febf0020)	GPIO1_D6	62	7	8	109	GPIO1_A1	UART6_TX_M1	I2C2_SCL_M4	SPI4_MOSI_M2	
			GND		9	10	110	GPIO1_A0	UART6_RX_M1	I2C2_SDA_M4	SPI4_MISO_M2	
			GPIO1_A4	36	11	12	97	GPIO3_A1		SPI4_MOSI_M1	PWM11_IR_M0 (febe0030)	
			GPIO1_A7	39	13	14		GND				
			GPIO1_B0	40	15	16	32	GPIO3_B5	UART3_TX_M1	CAN1_RX_M0	PWM12_M0 (febf0000)	
			3.3V		17	18	33	GPIO3_B6	UART3_RX_M1	CAN1_TX_M0	PWM13_M0 (febf0010)	
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND				
		SPI0_MISO_M2	GPIO1_B1	41	21	22	34	GPIO1_A2	UART6_RTSN_M1	I2C4_SDA_M3	SPI4_CLK_M2	PWM0_M2 (fd8b0000)
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2	UART7_RX_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2	UART7_TX_M2		
PWM13_M2 (febf0010)	UART1_RX_M1	I2C5_SDA_M3	GPIO1_B7	47	27	28	46	GPIO1_B6	I2C5_SCL_M3	UART1_TX_M1		
	UART1_CTSN_M1	I2C8_SDA_M2	GPIO1_D7	63	29	30		GND				
PWM10_M0 (febe0020)	SPI4_MISO_M1		GPIO3_A0	96	31	32	35	GPIO1_A3	UART6_CTSN_M1	I2C4_SCL_M3	SPI4_CS0_M2	PWM1_M2 (fd8b0010)
		PWM14_M0 (febf0020)	GPIO3_C2	114	33	34		GND				
	SPI4_CLK_M1	UART8_TX_M1	GPIO3_A2	98	35	36	101	GPIO3_A5	UART8_CTSN_M1			
			GPIO3_C1	113	37	38	100	GPIO3_A4	UART8_RTSN_M1	SPI4_CS1_M1		
			GND		39	40	99	GPIO3_A3	UART8_RX_M1	SPI4_CS0_M1		

2. The corresponding pins of the 6 groups of UART buses in 40pin are shown in the following table:

UART bus	RX corresponds to 40pin	TX corresponds to 40pin	dtbo corresponding configuration
UART1_M1	Pin 27	Pin 28	uart1-m1
UART3_M1	Pin 18	Pin 16	uart3-m1
UART4_M2	Pin 19	Pin 23	uart4-m2
UART6_M1	Pin 10	Pin 8	uart6-m1
UART7_M2	Pin 24	Pin 26	uart7-m2
UART8_M1	Pin 40	Pin 35	uart8-m1

Software setup for enabling UART



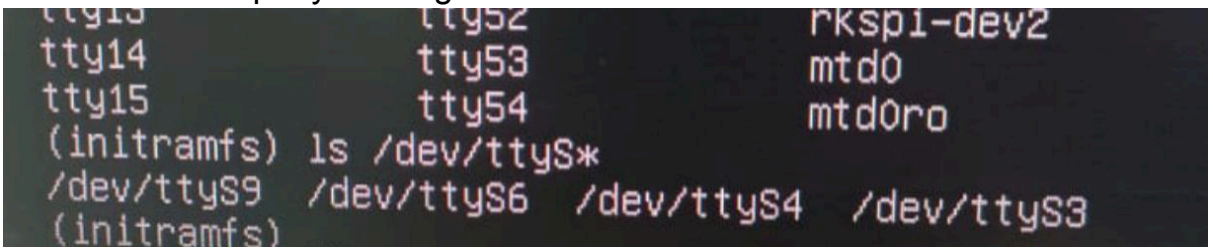


You have to enable both UART 3 and UART 6 in step(d) as shown above

UART 3 for range finder and UART 6 for the flight controller

In the last step you should most probably get **/dev/ttyS6 and /dev/ttyS3** as result if you are not getting anything other changes need to be made , if you are not getting value other than /dev/ttyS0 then CONTACT ME .

This is the output you will get if UART 3 and UART 6 are enabled



9.Installation of ROS2 jazzy for Armbian running ubuntu 24.04 Noble(if you have ubuntu 22.04 you should install ros2 humble)

ROS 2 Jazzy Installation Guide [reference]

This document provides a step-by-step guide to installing ROS 2 Jazzy on a Debian-based Linux system, such as Ubuntu 22.04 (Jammy Jellyfish) or Armbian distributions derived from Ubuntu. Following these steps will ensure a proper setup for development and usage of ROS 2.

---

1. System Setup

1.1. Set Locale 🌐

To ensure proper display and functionality, especially with UTF-8 characters, it is crucial to set your system locale correctly.

1. **Check current locale:** Open a terminal and run the following command to inspect your current locale settings.

   ```
   locale
   ```

   Verify that `UTF-8` is supported in the output. If it is not, proceed with the following steps.

2. **Install locales and generate UTF-8 locales:** This command ensures the necessary locale packages are installed and generates the `en_US.UTF-8` locale if it's not already available.

```
sudo apt update && sudo apt install locales
sudo locale-gen en_US en_US.UTF-8
```

3. **Update system locale settings:** This command configures the system to use `en_US.UTF-8` as the default locale for all language settings.

```
sudo update-locale LC_ALL=en_US.UTF-8
LANG=en_US.UTF-8
```

4. **Export the `LANG` environment variable for the current session:** This step applies the locale setting immediately to your current terminal session. For persistent changes, the `update-locale` command handles that for future sessions.

```
export LANG=en_US.UTF-8
```

5. **Verify updated settings:** After completing the above steps, run the `locale` command again to confirm that your settings now include `UTF-8` support.

locale

1.2. Enable Required Repositories 📦

You need to add the ROS 2 APT repository to your system's software sources to install ROS 2 packages.

1. **Enable the Ubuntu Universe repository:** This repository contains a vast collection of free and open-source software, including some dependencies required by ROS 2.

   ```
   sudo apt install software-properties-common
   sudo add-apt-repository universe
   ```

2. Install ros2-apt-source package: This package provides the necessary GPG keys and APT source configuration files for the official ROS 2 repositories. This automates the process of adding the correct repository.
   First, ensure curl is installed:

   ```
   sudo apt update && sudo apt install curl -y
   ```

3. Next, determine the correct `ROS_APT_SOURCE_VERSION` and download the `.deb` package. This command fetches the latest release version from GitHub.

```
export ROS_APT_SOURCE_VERSION=$(curl -s
https://api.github.com/repos/ros-infrastructure/ros-apt-so
urce/releases/latest | grep -F "tag_name" | awk -F\" '{print
$4}')
```

4. Then, download the ros2-apt-source Debian package. The $(. /etc/os-release && echo $UBUNTU_CODENAME) part automatically gets your Ubuntu codename (e.g., jammy for Ubuntu 22.04, or the derivative codename for Armbian).

   ```
   curl -L -o /tmp/ros2-apt-source.deb
   "https://github.com/ros-infrastructure/ros-apt-source/r
   eleases/download/${ROS_APT_SOURCE_VERSION}/r
   os2-apt-source_${ROS_APT_SOURCE_VERSION}.$(.
   /etc/os-release && echo
   $UBUNTU_CODENAME)_all.deb"
   ```

5. Finally, install the downloaded .deb package. This step adds the ROS 2 repositories and keys to your system.

   ```
   sudo dpkg -i /tmp/ros2-apt-source.deb
   ```

6. This sequence ensures the correct ros2-apt-source Debian package is downloaded and installed, automatically configuring the ROS 2 repositories for your system.

1.3. Install Development Tools 🛠️

If you plan to **build ROS packages from source** or engage in ROS 2 development beyond using pre-built binaries, it is highly recommended to install the development tools. This includes `rosdep`, `vcstools`, and other utilities.

```
sudo apt update && sudo apt install
ros-jazzy-ros-dev-tools
```

Note: The package name is `ros-jazzy-ros-dev-tools` for Jazzy, not just `ros-dev-tools`.

1.4. Install ROS 2 Jazzy 🚀

After successfully setting up the repositories, you can proceed with the installation of ROS 2.

1. **Update APT repository caches:** Before installing new packages, always refresh your local package list to ensure you're getting the latest available versions and information from the newly added ROS 2 repositories.

   ```
   sudo apt update
   ```

2. **Upgrade your system packages:** It is strongly recommended to ensure your entire system is up to date before installing new core software like ROS 2 to prevent potential dependency conflicts.

   ```
   sudo apt upgrade
   ```

3. **Install ROS 2 Jazzy Desktop (Recommended):** This is the most comprehensive installation option and is

recommended for most users. It includes the core ROS 2 packages, the RViz visualization tool, various demos, and tutorials.

```
sudo apt install ros-jazzy-desktop
```

---

2. Set Up Environment 💻

To use ROS 2 commands (like `ros2 run`, `ros2 topic`, etc.) in your terminal, you need to source the ROS 2 setup file. It is best practice to add this command to your `~/.bashrc` file. This ensures that the ROS 2 environment is automatically configured every time you open a new terminal session.

```
echo "source /opt/ros/jazzy/setup.bash" >> ~/.bashrc
```

After adding this line, you can either open a new terminal session or manually source your `~/.bashrc` file in the current session for the changes to take effect:

```
source ~/.bashrc
```

The range finder repo for ros2 control :

[Link to Github]

Here's the documentation for the ROS 2 driver for the EY09A Laser Rangefinder, formatted for a PDF document (suitable for pasting into Word).

ROS 2 Driver for EY09A Laser Rangefinder

This document provides instructions for setting up and running a ROS 2 driver for the EY09A Micro Laser Rangefinder. This driver facilitates communication with the sensor via USB/UART, enabling it to publish range data within the ROS 2 ecosystem.

Prerequisites

Before proceeding with the installation, ensure the following prerequisites are met on your Linux system:

- ROS 2 (Humble or equivalent): A working installation of ROS 2, with its environment sourced in your terminal. This guide assumes a `humble` (or newer) distribution.
- Python 3.8+: ROS 2 typically comes bundled with a compatible Python version.
- `pyserial`: This crucial Python library is used for serial communication with the laser rangefinder. It will be installed during the installation steps.

  sudo pip3 install pyserial –break-system-packages

- Serial Port Access: Your user account needs the necessary permissions to access serial ports (e.g., `/dev/ttyUSB0`,

/dev/ttyACM0,/dev/ttyS3). To grant these permissions, execute the following command in your terminal:

```
sudo usermod -a -G dialout $USER
```

- Important: You must log out and log back in (or reboot your system) for this change to the user group to take effect. Failure to do so will result in permission errors when attempting to access the serial port.

Installation Steps

Follow these steps carefully to set up and build the ROS 2 driver package in your workspace:

1. Navigate to your home directory: Begin by ensuring you are in your home directory, which is a common starting point for creating development workspaces.

```
cd ~
```

2. Create a ROS 2 workspace: A ROS 2 workspace is a directory where you will store your ROS 2 packages. The src subdirectory is where your package source code will reside.

```
mkdir -p lidar_ws/src
```

3. Change into the source directory of your new workspace: All subsequent package cloning will happen within this directory.

```
cd lidar_ws/src
```

4. Clone the rangefinder ROS 2 package: This command downloads the source code for the EY09A laser rangefinder driver from its GitHub repository into your workspace's src directory.

git clone https://github.com/a11m234/rangefinder.git

5. Change into the cloned package directory: This places you inside the newly cloned rangefinder package, which contains its requirements.txt file.

cd rangefinder

6. Install Python dependencies: This step installs pyserial and any other Python libraries listed in the requirements.txt file that are necessary for the driver to function. It's installed into your system's Python environment (or the active virtual environment if you are using one).

pip3 install -r requirements.txt

    - Note on Permissions: If you encounter permission errors (e.g., Permission denied), it usually means your user does not have sufficient privileges to install packages directly to the system Python directories. While sudo pip3 install -r requirements.txt might resolve this, it is generally not recommended as it can lead to system-wide Python environment issues. A better practice is to ensure your user has appropriate permissions, or ideally, use a Python virtual environment for project-specific dependencies.
7. Navigate back to the root of your ROS 2 workspace: After installing dependencies, return to the lidar_ws directory to build the entire workspace.

cd ~/lidar_ws/

8. Build the ROS 2 package: This command uses `colcon` (the ROS 2 build tool) to compile your `rangefinder` package and any of its dependencies. If the build completes without errors, it means the package is successfully compiled and ready for use.

```
colcon build
```

9. Source the ROS 2 workspace: To make your newly built package, its executables, and its launch files discoverable by ROS 2, you need to source the workspace setup file.
    - For the current terminal session: This command will enable ROS 2 to find your package only in the terminal window where it is executed.

```
source install/setup.bash
```

    - For permanent setup (recommended): To avoid sourcing the workspace every time you open a new terminal, add the sourcing command to your `~/.bashrc` file. This ensures the workspace is automatically configured upon starting a new shell session.

```
echo "source ~/lidar_ws/install/setup.bash" >> ~/.bashrc
source ~/.bashrc # Apply changes to the current terminal
```

Running the Driver

After completing the installation steps and sourcing your workspace, you can now launch the EY09A laser rangefinder driver.

1. Launch the ROS 2 node: This command starts the ROS 2 node that communicates with the laser rangefinder. By default, it attempts to connect to `/dev/ttyS3`.

```
ros2 launch rangefinder rangefinder_launch.py
```

   - Specify Serial Port (if different from default `/dev/ttyUSB0`): If your EY09A rangefinder is connected to a different serial port (e.g., `/dev/ttyACM0`, `/dev/ttyS3`, `/dev/ttyAMA3`), you must specify it when launching the node. Replace `/dev/ttyS3` with your actual port.

ros2 launch rangefinder rangefinder_launch.py serial_port:=/dev/ttyUSB0

   - **To identify available serial ports on your system, you can use the command:**

```
ls /dev/tty*
```

   - This command will list all detected TTY (teletypewriter) devices, which include serial and USB-to-serial adapters. Look for names like `ttyUSBX`, `ttyACMX`, `ttySX`, or `ttyAMAX`.

2. Verify Data (in a new terminal): To confirm that the driver is successfully communicating with the rangefinder and publishing data, open a new terminal window. Ensure that your ROS 2 workspace is sourced in this new terminal (as per step 9 of the Installation Steps). Then, use the `ros2 topic echo` command to view the messages being published on the `/rangefinder/range` topic:

```
ros2 topic echo /rangefinder/range
```

3. You should now see `sensor_msgs/msg/Range` messages continuously appearing in your terminal. This indicates that the driver is successfully acquiring and publishing range data from your EY09A laser rangefinder.