**Initial Setup**

Flash a microSD card with [Ubuntu Rockchip](https://github.com/Joshua-Riek/ubuntu-rockchip) for the relevant OrangePi board. You want Ubuntu 22.04, and this guide will assume the full desktop edition, though the server version should work just the same. Any flash tool should work fine. If on Linux and using Balena etcher, first open the folder where you downloaded the .img.xz and unxz the file.

Once flashed, load the card into the OrangePi. If you have a variant of the OrangePi without wifi, connecting your phone via USB and using USB tethering works well. Complete setup, then

sudo apt upgrade && sudo apt update

After that runs, install ROS2 Humble according to [this](https://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debs.html) guide. Install the full ros-humble-desktop and ros-dev-tools. Once installed, rather than sourcing the environment every time, simply open the .bashrc file in the home directory (nano $HOME/.bashrc in the terminal) and add source /opt/ros/humble/setup.bash to the end of the file.

**Prepare for ODrive Use**

To use the ODrive motor controller, you will want a full python and pip environment. Additionally, you want to be able to use CAN. To ensure that both capabilities are present,

sudo apt install python3-pip can-utils

Once pip is installed, also run

pip3 install –upgrade python-can odrive

Next, CAN1 and PWM must be enabled on the OrangePi. Run

sudo nano /etc/default/u-boot

to edit the boot file and replace #U\_BOOT\_FDT\_OVERLAYS with

U\_BOOT\_FDT\_OVERLAYS="device-tree/rockchip/overlay/rk3588-can1-m1.dtbo

device-tree/rockchip/overlay/rk3588-i2c1-m4.dtbo

device-tree/rockchip/overlay/rk3588-i2c5-m3.dtbo

device-tree/rockchip/overlay/rk3588-pwm0-m1.dtbo device-tree/rockchip/overlay/rk3588-pwm13-m2 device-tree/rockchip/overlay/rk3588-pwm14-m1 device-tree/rockchip/overlay/rk3588-pwm15-m2 device-tree/rockchip/overlay/rk3588-pwm1-m2 device-tree/rockchip/overlay/rk3588-pwm3-m0"

then make it permanent with

sudo u-boot-update

Reboot to apply changes. The Orange Pi 5 documentation page 267 has a table showing the pins, names, and some other related info. At this point, follow the [ODrive S1 documentation](https://docs.odriverobotics.com/v/latest/guides/can-guide.html) to finish setup. Relevant packages to install are

sudo apt install can-utils net-tools && pip3 install python-can

and the important command is

sudo ip link set can0 up type can bitrate 250000

This turns the CAN bus on, and must be run on every boot. Alternatively, it can be set to auto run using a [custom systemd service](https://unix.stackexchange.com/questions/645914/running-a-sudo-command-automatically-on-startup).

It is easiest to configure each ODrive over USB with the [odrivetool GUI](https://gui.odriverobotics.com/dashboard). To do so, connect each ODrive to the OrangePi via USB isolator. Configure the Power Source section according to the battery specifications. For the Motor Parameters, we have the following:

**Motor Type: Gimbal**

**Phase Resistance: 0.27**

**Pole Pairs: 3**

**Kv: 300**

**Current Limit: 12**

Though the motors have a current limit of 9 A, there seems to be a scaling factor that was capping their draw to about 1.5 A with that setting. Raising it allowed them to pull more current, and this setting did the best at reducing heat load. The remaining settings can be set according to the power source. The incremental encoder should have 5000 counts, but not all are equivalent. If it fails calibration as a result of this, it will give you the correct number to use. Control mode is position, and a velocity limit of 10 gives plenty of speed to work with. The relevant interface parameters are as follows:

**CAN Bus: Enabled**

**Bitrate: 250000 (Default)**

**Node ID: Set each according to upper leg as the odd number, lower even**

**Heartbeat and Feedback: 100 ms**

Save and apply configuration, then calibrate. Under the dashboard PID gains can be set, and for best results I suggest reducing the D gain by an order of magnitude.

**Prepare Workspace**

First, make a directory for the workspace. This can be anywhere, these examples will create a folder to store multiple workspaces, then the necessary folders for a workspace, and then move into the source folder for that workspace.

mkdir ros2Workspaces

cd ros2Workspaces

mkdir odriveWS

cd odriveWS

mkdir src

cd src

From here, download and build the odrive CAN package.

git clone <https://github.com/odriverobotics/ros_odrive>

colcon build --packages-select odrive\_can

To ensure your terminal will recognize these packages, you can run

source ./install/setup.bash

from the src directory. At this point, you can control the ODrive from the CAN bus! Examples of how to do so from the terminal can be found in the [ODrive documentation](https://docs.odriverobotics.com/v/latest/guides/ros-package.html).

**Enabling I2C Adafruit Blinka Libraries**

In order to communicate with the IMU, we need to enable the I2C pins and Adafruit Blinka libraries. First, install the libraries with

pip install blinka libgpiod lsm6dox

Next, we need to add udev rules to enable the pins. The first one is created with

sudo nano /etc/udev/rules.d/60-gpiod.rules

In it, paste the following:

SUBSYSTEM=="gpio", KERNEL=="gpiochip\*", GROUP="gpiod", MODE="0660"

The second is created with

sudo nano /etc/udev/rules.d/99-gpio.rules

and contains the following:

SUBSYSTEM==”gpio”, KERNEL==”gpiochip\*”, ACTION==”add”, PROGRAM=”/bin/sh -c ‘chown root:gpio /sys/class/gpio/export /sys/class/gpio/unexport ; chmod 220 /sys/class/gpio/export /sys/class/gpio/unexport’” SUBSYSTEM==”gpio”, KERNEL==”gpio\*”, ACTION==”add” PROGRAM==”/bin/sh -c ‘chown root:gpio /sys%p/active\_low /sys%p/direction /sys%p/edge /sys%p/value ; chmod 660 /sys%p/active\_low /sys%p/direction /sys%p/edge /sys%p/value’”

Finally, create the groups to access these values with

addgroup --system i2c

addgroup --system gpio

sudo groupadd gpiod

and add yourself to them with

sudo usermod -aG i2c username

sudo usermod -aG gpio username

sudo usermod -aG gpiod username

where username is the username you set on installation.