

Jetson Setup

This is a guide on how to enable all the functionality of the Jetson from a fresh flash. If you just received the drone, it is already set up likely! Only follow this guide if you need to re-flash the drone.

Flashing the Jetson

There are 3 options, in order of what is easiest to hardest.

Option 1: Flash Stereolabs image

Go here: <https://www.stereolabs.com/docs/get-started-with-zed-box-orin-nx/reset-update#flashing-the-zed-box-orin-nx-model>

Option 2: Follow Auvideo instructions

Follow instructions from "[Preliminary Orin nano and Orin NX flashing guide](#)" on Auvideo's website.

The Jetson files for Jetpack 6.0 can be downloaded here:

- https://developer.nvidia.com/downloads/embedded/l4t/r36_release_v3.0/release/jetson_linux_r36.3.0_aarch64.tbz2
- https://developer.nvidia.com/downloads/embedded/l4t/r36_release_v3.0/release/tegra_linux_sample-root-filesystem_r36.3.0_aarch64.tbz2

Option 3: Follow instructions in Auvideo firmware

Download most recent firmware from <https://auvideo.eu/firmware/>.

Follow instructions inside. However, note that BEFORE starting the final flashing script, you need to change the EEPROM size. Open `<working directory>/Linux_for_Tegra/bootloader/generic/BCT/tegra234-mb2-bct-misc-p3767-0000.dts`. Change `cvb_eeprom_read_size = <0x100>;` to `cvb_eeprom_read_size = <0x0>`

On first boot, it will ask you for your username and password. Put username `spade` and password `spade`.

Post-flash instructions

If you used either of the Auvideo methods, DO NOT RUN `sudo apt-get upgrade`! You may brick the device.

First, connect to the device. Either connect a screen and keyboard, or connect the device to ethernet and log in.

Next, clone this repository onto the device.

```
cd ~
mkdir workspaces
git clone --recurse-submodules git@gitlab.nibio.no:spade/forest-scanner-
```

```
ros1.git
cd forest-scanner-ros1
```

Next, run the post-setup tasks. These have been compiled into a script. Note, the script is untested, but if there are errors they should be fairly straightforward to fix by looking at the script itself.

```
./jetson-setup/post-install.sh
```

This will install some extra packages, install NoMachine for remote access, set up docker, and install the wifi drivers.

After running this script, you will need to reboot for the wifi drivers to come into effect.

Next, connect to wifi. Replace the line of code below with your own wifi SSID and password.

```
nmcli device wifi connect 'WIFI_SSID' password 'WIFI_PASSWORD' ifname wlan0
```

At this point, you should reconnect to your device over Wifi, if you're using ethernet. Subsequent steps will dedicate the port to the LiDAR.

Note! Your wifi network cannot use the subnet address 192.168.1.xx. This is used by the LiDAR!

SSH Setup

To ensure SSH is set up properly, type:

```
sudo nano /etc/ssh/sshd_config
```

Ensure that

```
PubkeyAuthentication yes
```

isn't commented out, and change the `AuthorizedKeysFile` line to:

```
AuthorizedKeysFile      .ssh/authorized_keys
```

Then `^X` to exit and save.

To log in without password, open the `authorized_keys` file and add your public key:

```
sudo nano ~/.ssh/authorized_keys
```

Setup ethernet connection for LiDAR

The script `scripts/setup-network.sh` is required to be run at startup to assign a fixed IP address to the ethernet port. Once you set this up, the port won't be available for internet connection, so do this last.

Assuming your working directory is the root of this repository:

```
# Install the service
cp ./jetson-setup/setup-network.service.txt /etc/systemd/system/setup-
network.service
# Reload the systemctl daemon
sudo systemctl daemon-reload
# Enable the service
sudo systemctl enable startup-network.service
```

Reboot the computer, then check that the `eth0` interface is set to the port `192.168.1.5`:

```
ip addr show
```

The result should show:

```
$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: can0: <NOARP,ECHO> mtu 16 qdisc noop state DOWN group default qlen 10
    link/can
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 2312 qdisc mq state UP
group default qlen 1000
    link/ether 34:c9:f0:96:81:b5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.217/24 brd 192.168.0.255 scope global dynamic
noprofixroute wlan0
        valid_lft 40984sec preferred_lft 40984sec
    inet6 fd67:8d5e:ecd4:0:d8ca:eabd:5aa9:c37a/64 scope global temporary
dynamic
        valid_lft 602585sec preferred_lft 83686sec
    inet6 fd67:8d5e:ecd4:0:83d:3e5b:e44f:9489/64 scope global mngtmpaddr
noprofixroute
        valid_lft forever preferred_lft forever
    inet6 2001:2020:337:ae26:d797:1093:8fbe:4043/64 scope global temporary
dynamic
        valid_lft 602585sec preferred_lft 83686sec
```

```
    inet6 2001:2020:337:ae26:b2a3:7d91:9e4b:f74b/64 scope global mngtmpaddr
noprofixroute
    valid_lft forever preferred_lft forever
    inet6 fe80::93f9:a86a:1b6d:58f8/64 scope link noprofixroute
    valid_lft forever preferred_lft forever
4: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP group default qlen 1000
    link/ether 48:b0:2d:ea:f4:fe brd ff:ff:ff:ff:ff:ff
    altname enP8p1s0
    inet 192.168.1.5/24 scope global eth0 <----- CHECK ADDRESS HERE
    valid_lft forever preferred_lft forever
    inet6 fe80::4ab0:2dff:feea:f4fe/64 scope link
    valid_lft forever preferred_lft forever
5: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue
state DOWN group default
    link/ether 02:42:d7:c0:41:28 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
    valid_lft forever preferred_lft forever
```

Your system is not set up and ready for use.

Build and run docker

This can be done via

```
./scripts/init.sh
```

This will build and run the docker image, then compile the source code.