Adnan project

Micro-ROS with ESP32: RC Car Control using a Nintendo Joystick

AUTHOR

Eng. Adnan Abdallah

2024-11-10

Contents

List of Figures

1 Introduction to Micro-ROS

Micro-ROS is an extension of the Robot Operating System 2 (ROS 2) designed specifically for microcontrollers and resource-constrained devices. It is similar to rosserial for ROS 1 but is not compatible with Arduin UNO, Nano or Mega. ROS 2 is a widely used framework in robotics that facilitates communication between different components of a robot, making it easier to create modular and scalable robotics applications. However, traditional ROS 2 was too resource-intensive for smaller embedded systems like microcontrollers. Micro-ROS was developed to bring the power of ROS 2 to microcontroller platforms, enabling seamless integration into larger robotics systems.

Micro-ROS supports various microcontrollers, including the popular ESP32, which is widely used in IoT and robotics applications due to its low cost, versatility, and built-in Wi-Fi and Bluetooth. With Micro-ROS on an ESP32, we can develop applications where this low-power device communicates over Wi-Fi with a more powerful ROS 2 system, such as a desktop or robot brain.

In short, Micro-Ros is the ROS 2 companion of microcontrollers for robotic applications. For example, we can create nodes, publishers and subscribers on the ESP32 using Micro-ROS and connecting the ESP32 to a local PC using USB port or even using a wireless connection.

2 Project Overview

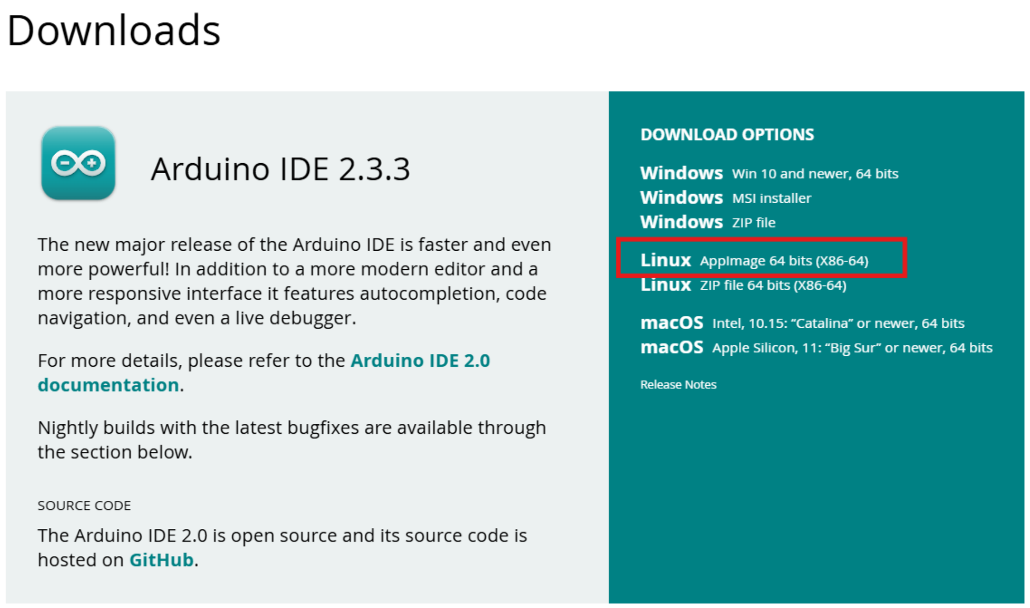
This project guides you through setting up Micro-ROS on an ESP32 using the Arduino IDE. As a practical example, we will control an RC car using an ESP32, which receives commands from a Nintendo Joy-Con controller over ROS 2 topics. We will start with the basics, going over the installation of Arduino IDE, connecting the ESP32 to the IDE, Installing Micro-ROS library, installing Mircro-ROS agent, and finally connecting everything through a small project.

3 Prerequisites

- ESP32 Board
- Nintendo Joy-Con Controller or any other controller (Xboc, Play Station) or you can just use your PC keyboard
- RC Car hardware with motors and an H-bridge motor driver
- Basic ROS 2 knowledge (publisher, subscriber, topics)

4 Download Arduino IDE

- Go to the official Arduino website and download the Linux 64-bit version of Arduino IDE 2.x.x. Select the AppImage option as shown in the figure.


Downloads

Arduino IDE 2.3.3

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the **Arduino IDE 2.0 documentation**.

Nightly builds with the latest bugfixes are available through the section below.

SOURCE CODE

The Arduino IDE 2.0 is open source and its source code is hosted on **GitHub**.

DOWNLOAD OPTIONS

Windows Win 10 and newer, 64 bits
Windows MSI installer
Windows ZIP file

Linux AppImage 64 bits (X86-64)
Linux ZIP file 64 bits (X86-64)

macOS Intel, 10.15: "Catalina" or newer, 64 bits
macOS Apple Silicon, 11: "Big Sur" or newer, 64 bits

Release Notes

Figure 1: Arduino IDE Download Option

- Create an installation directory in your home folder by typing the below commands in a new terminal:

```
cd ~
mkdir Arduino
```

- Copy the installed `AppImage` into the new directory (check Figure 2)
- Open new terminal and navigate to the Arduino directory previously created (that contains the installed AppImage) then type these commands to complete the installation

```
cd ~/Arduino
sudo add-apt-repository universe
sudo apt install libfuse2
chmod +x <name_of_the_AppImage> // e.g arduino-ide-2.3.3_Linux.AppImage
```

- Create a shortcut (or Desktop icon) for the IDE

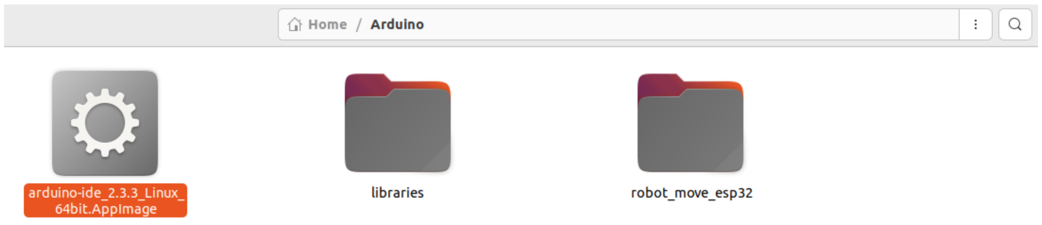


Figure 2: AppImage Copied to New Arduino Directory

```
    sudo apt install gedit
    cd ~/.local/share/applications
    gedit arduino.desktop
```

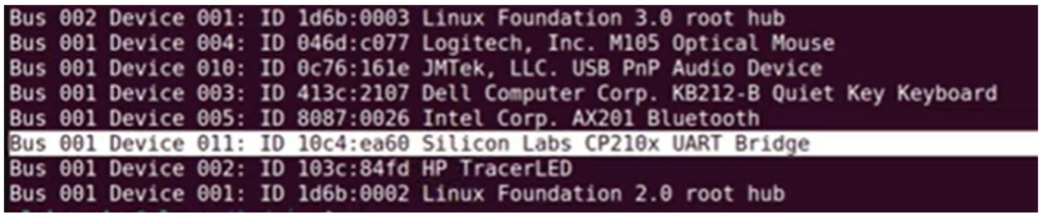
- A new file will pop up after executing the last command. You should type the below commands and save the file:

```
    [Desktop Entry]
    Type=Application
    Name=Arduino IDE 2.3.3
    Exec=/home/<username>/Arduino/<App_image_name.AppImage
```

- To know the `<username>` use the command `whoami`

Add Port Permissions

- Connect your ESP32 or any Arduino board to your computer and identify the port where your board is connected. To confirm that your ESP32 is detected use the command `lsusb` as shown in Figure 3, the highlighted line should appear.



```
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 004: ID 046d:c077 Logitech, Inc. M105 Optical Mouse
Bus 001 Device 010: ID 0c76:161e JMTek, LLC. USB PnP Audio Device
Bus 001 Device 003: ID 413c:2107 Dell Computer Corp. KB212-B Quiet Key Keyboard
Bus 001 Device 005: ID 8087:0026 Intel Corp. AX201 Bluetooth
Bus 001 Device 011: ID 10c4:ea60 Silicon Labs CP210x UART Bridge
Bus 001 Device 002: ID 103c:84fd HP TracerLED
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Figure 3: Confirm ESP32 Detection

- To know the name of the used usb port use the following command

```
ls -la /dev | grep ttyACM
```

- If nothing is printed on the terminal try this command:

```
ls -la /dev | grep ttyUSB
```

- After identifying the port name and number you should add read and write permissions to this port by using the below command:

```
sudo chmod a+rw /dev/<portname> (e.g ttyACM0 or ttyUSB0)
```

- To avoid requiring `sudo` each time, add yourself to the `dialout` group:

```
sudo usermod -aG dialout $USER
```

- If you have any IDE open close it and launch it again for changes to take effect.

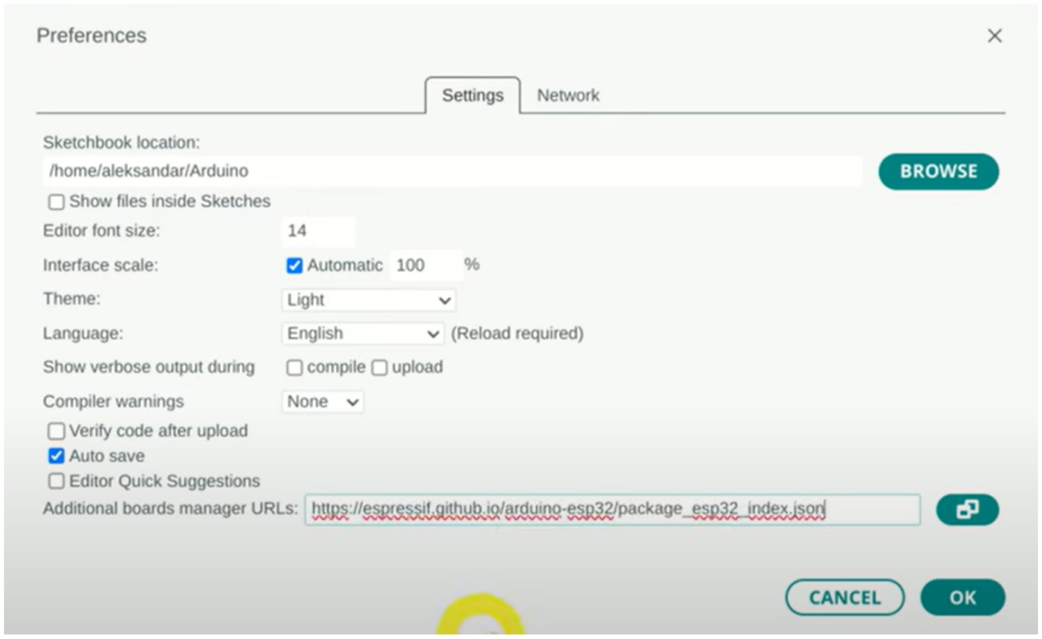
After completing these steps, the Arduino IDE 2.x.x should be set up with appropriate serial port permissions.

5 Install the ESP32 Board in Arduino IDE

- Open Arduino IDE, go to **File > Preferences**, and add the ESP32 board URL to **Additional Board Manager URLs** (chek Figure 4):

```
https://dl.espressif.com/dl/package_esp32_index.json
```

- Go to **Tools > Board > Board Manager**, search for **ESP32** and install it (Check Figure 5)
- Now select the bord `ESP32 Dev Module` along with the corresponding port and try to upload any example code. **Warning !!**. If you are using the ESP32 for the first time you might face an upload error. To resolve this problem hold down the `Booot` button then press and release the `RESET` button while still holding the `BOOT` button.


Preferences

Settings | Network

Sketchbook location:
/home/aleksandar/Arduino
BROWSE
☐ Show files inside Sketches
Editor font size: 14
Interface scale: ☑ Automatic 100 %
Theme: Light
Language: English (Reload required)
Show verbose output during ☐ compile ☐ upload
Compiler warnings None
☐ Verify code after upload
☑ Auto save
☐ Editor Quick Suggestions
Additional boards manager URLs: https://espressif.github.io/arduino-esp32/package_esp32_index.json

CANCEL OK

Figure 4: Adding ESP32 Link

6 Install the Micro-ROS Arduino Library

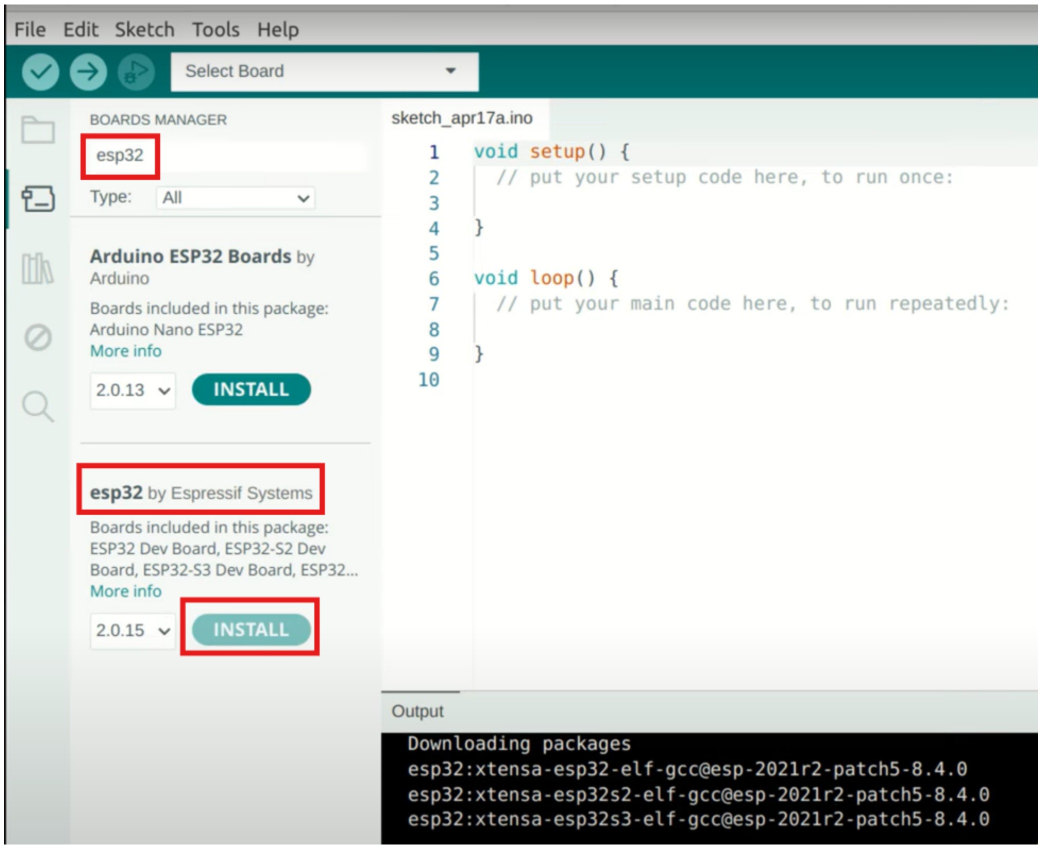
- This link will take you to the official Micro-ROS Arduino library GitHub repository. Once open, select the ROS 2 version installed on your PC and download the repository as `.zip` (check Figure 6)
- Install the **Micro-ROS Arduino** library by adding it to the Arduino IDE. Open your Arduino IDE and go to **Sketch > Include Library > Add .Zip Library** and select the downloaded library (from your download directory). Check Figure 7.
- Go to **File > Examples** and scroll down to `micrros arduino` and select `microros publisher` then upload it to your ESP32. Check Figure 9 for details.

7 Run Micro-ROS Agent

For the purpose of the tutorial, we will assume that the used ROS2 version is `humble`. However the process is the same for other versions just replace humble with your ROS version (distribution)

- Open a new terminal and source the environment

```
source /opt/ros/humble/setup.bash
```



File Edit Sketch Tools Help
Select Board
BOARDS MANAGER
esp32
Type: All
Arduino ESP32 Boards by Arduino
Boards included in this package:
Arduino Nano ESP32
More info
2.0.13
INSTALL
esp32 by Espressif Systems
Boards included in this package:
ESP32 Dev Board, ESP32-S2 Dev Board, ESP32-S3 Dev Board, ESP32...
More info
2.0.15
INSTALL
sketch_apr17a.ino
Output

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

```
Downloading packages
esp32:xtensa-esp32-elf-gcc@esp-2021r2-patch5-8.4.0
esp32:xtensa-esp32s2-elf-gcc@esp-2021r2-patch5-8.4.0
esp32:xtensa-esp32s3-elf-gcc@esp-2021r2-patch5-8.4.0
```

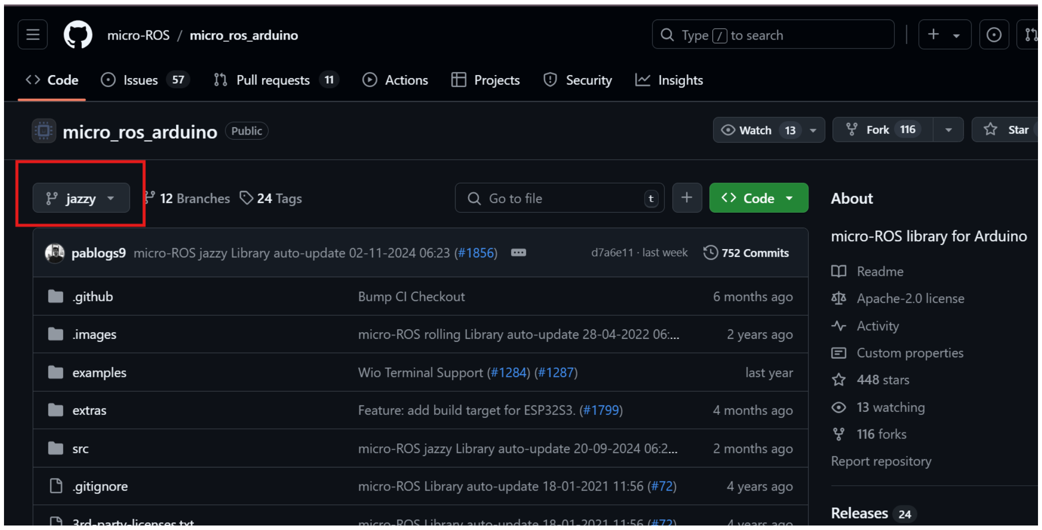
Figure 5: Adding ESP32 Boards

- Create the workspace folder

```
mkdir ws_test_microros
cd ws_test_microros
git clone -b $ROS_DISTRO https://github.com/micro-ROS/micro_ros_setup.git
    src/micro_ros_setup
sudo apt install python3-rosdep2
sudo apt update && rosdep update
rosdep install --from-paths src --ignore-src -y
sudo apt-get install python3-pip
colcon build
source ~ws_test_microros/install/local_setup.bash
```

- Create and build Micro-Ros2 agent:

```
   ros2 run micro_ros_setup create_agent_ws.sh
```


micro-ROS / micro_ros_arduino

Code · Issues 57 · Pull requests 11 · Actions · Projects · Security · Insights

micro_ros_arduino Public · Watch 13 · Fork 116 · Star

jazzy · 12 Branches · 24 Tags · Go to file · Code

pablogs9 micro-ROS jazzy Library auto-update 02-11-2024 06:23 (#1856) · d7a6e11 · last week · 752 Commits

.github	Bump CI Checkout	6 months ago
.images	micro-ROS rolling Library auto-update 28-04-2022 06:...	2 years ago
examples	Wio Terminal Support (#1284) (#1287)	last year
extras	Feature: add build target for ESP32S3. (#1799)	4 months ago
src	micro-ROS jazzy Library auto-update 20-09-2024 06:2...	2 months ago
.gitignore	micro-ROS Library auto-update 18-01-2021 11:56 (#72)	4 years ago

About

micro-ROS library for Arduino

Readme · Apache-2.0 license · Activity · Custom properties · 448 stars · 13 watching · 116 forks · Report repository

Releases 24

Figure 6: Github Page for Micro-ROS Library

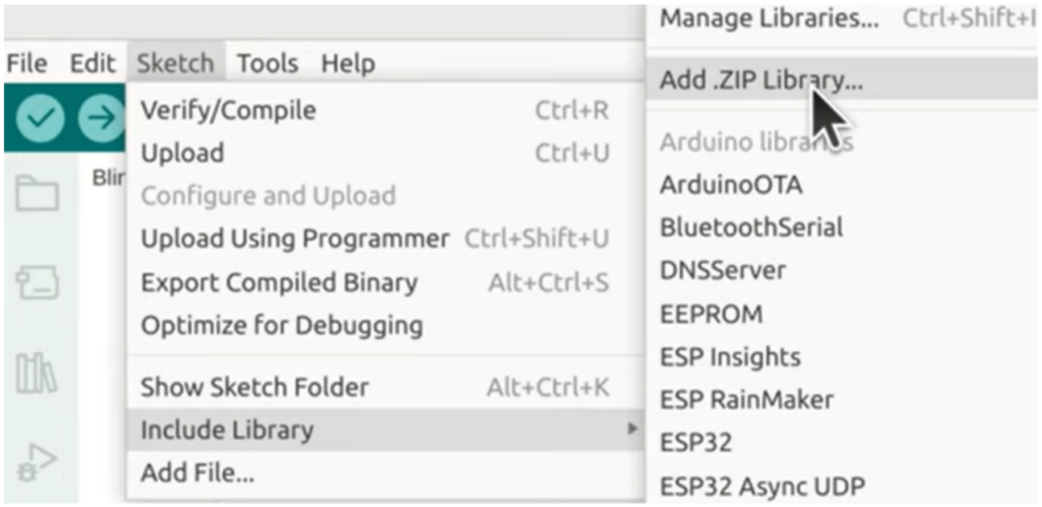


Figure 7: Adding Micro-ROS Library to Arduino IDE

```
    ros2 run micro_ros_setup build_agent.sh
    source ~ws_test_microros/install/local_setup.bash
```

- Run the agent program (for this step your ESP32 should be connected to your PC with the publisher code uploaded). Replace `ttyUSB0` by the name of your USB port

```
ros2 run micro_ros_agent micro_ros_agent serial ——dev/dev/ttyUSB0
```

- An output similar to Figure 9 should appear

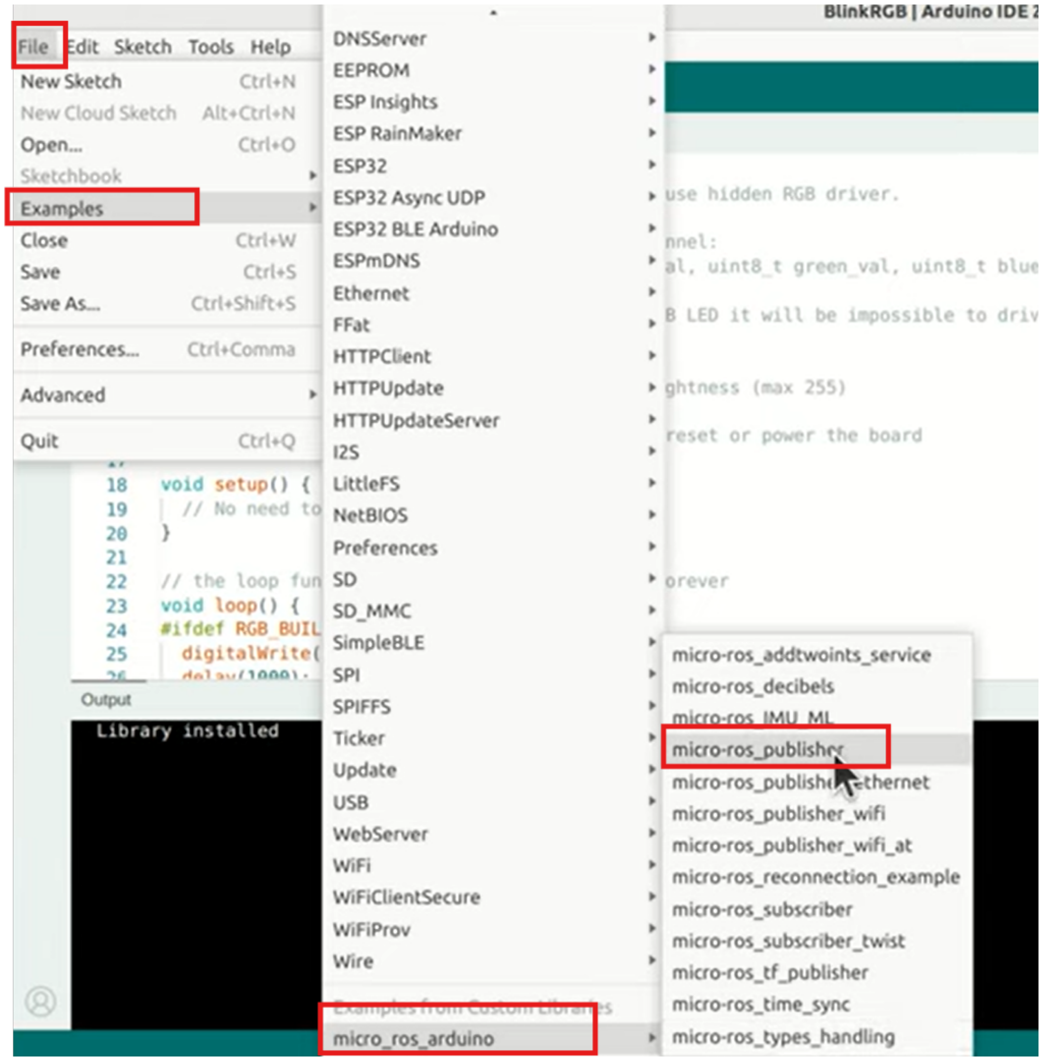
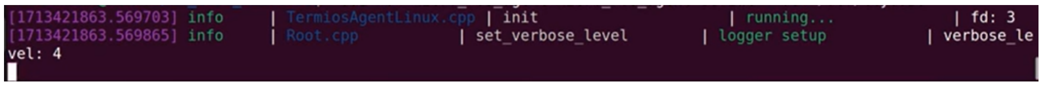


Figure 8: Publisher Example Upload



```
[1713421863.569703] info     | TermiosAgentLinux.cpp | init                     | running...             | fd: 3
[1713421863.569865] info     | Root.cpp              | set_verbose_level        | logger setup           | verbose_le
vel: 4
```

Figure 9: Agent Output

- To complete the setup press the `RESET` button on the ESP32 board.
- To visualize the messages sent from the ESP32 publisher open a new terminal and type the following commands (keep the Micro-ROS2 Agent running)

```
source /opt/ros/humble/setup.bash
```

```
ros2 topic list
```

- The below output should appear:

```
/micro_ros_arduino_node_publisher
/parameter_events
/rosout
```

- Type the following command in the same terminal

```
ros2 topic echo /micro_ros_arduino_node_publisher
```

8 Building the RC Car Controller

The following tutorials will include projects using Micro-ROS and ESP32. As an example check the below project. An RC car controlled using an external Joystick.

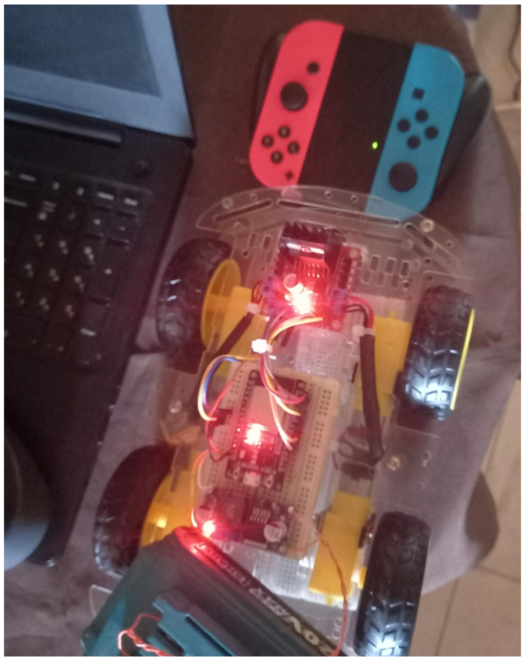


Figure 10: RC Car Project