Road Map

ROS2 Hardware Interfacing Learning Guide

Using rosserial with ROS2 using ros1_bridge

Some processes are better done using stand-alone micro-controllers like driving motors, however, this need communication with ROS2 nodes. ROS2 provides micro-ros framework for communicating with micro-controllers, but it needs a bit powerful boards and isn't suitable for AVR-based boards like Arduino UNO or Mega. One easy way to overcome this is using ROS1 with its serial communication package rosserial through ros1_bridge.

Preparing the ROS1 Enviroment

1. Remove ROS2 sourcing from .bashrc file
   From now on, we will install and use both ROS1 and ROS2 distributions, thus having any of them sourced in all the terminals we use will lead to conflicting dependencies
2. Follow this link to install ROS1 (and don't put its sourcing in .bashrc).
3. Install rosserial and rosserial-arduino:
   ```
   sudo apt-get install ros-noetic-rosserial
   sudo apt-get install ros-noetic-rosserial-arduino
   ```

Preparing the Arduino Enviroment

1. Install the Arduino IDE from the Software Store. When opening for the first time it will prompt you to enter a command through the terminal to enable serial port access. Follow their instructions then reboot your device
2. To install the ros_lib library, don't use the library manager like in windows as it will cause compilation error due to missing dependencies. Instead, install it manually:
   1. Navigate to the Arduino's libraries folder. To know its location: open the IDE the browse to file>preferences . The sketchbook location will be written there. Browse to this location and you'll find a folder called libraries. Open this folder in the terminal.
   2. Source the ROS1 distribution (`. /opt/ros/noetic/setup.bash`) then enter this command:
      ```
      rosrun rosserial_arduino make_libraries.py .
      ```

Installing and using ros1_bridge

1. Enter this command:
   ```
   sudo apt install ros-foxy-ros1-bridge
   ```
2. Try the examples from this link. Note that before the example they say that ros1_bridge source code must be built from source in the workspace. I tried skipping this step and replacing it with the previous command and everything worked fine. This step, however, may be needed later on if we wish to use a custom interface.
   To sum up, for now you can just start from the "Example 1" subtitle.

Using rosserial with Arduino and communicating with ROS2

Now you have everything set and ready. Go through the tutorials in this link. The second tutorial creates a simple string publisher on Arduino and subscribe to it using ROS1. Follow all the steps but instead of subscribing using ROS1 node, open a ros1_bridge like in the previous section's example and subscribe using a ROS2 node.

The following tutorial is a simple subscriber on the Arduino that toggles the built-in LED with each received message. Similarly, follow it but use a ROS2 publisher instead of ROS1 publisher.