National University of Sciences & Technology
Course: MTS - 417 Intro to Robotics
Lab Manual
Prepared By: Dr. Tayyab Zafar & LE Hamza Sohail

Lab Number 11
Interfacing ROS with Arduino

Introduction:

Robot Operating system (ROS) can be integrated with Arduino. Before doing this lab, make sure that ROS is installed in your operating system ubuntu.

Ubuntu and ROS Version:

Ubuntu: 18.04
ROS : Melodic

Arduino Board:

Arduino Nano

Objectives of the Lab:

- To interface Arduino with ROS

➔ Open Ubuntu and install Arduino IDE directly from browser.

➔ Open terminal

➔ Write down the following lines on terminal to install the desired libraries to connect ROS with Arduino.

sudo apt install ros-melodic-rosserial

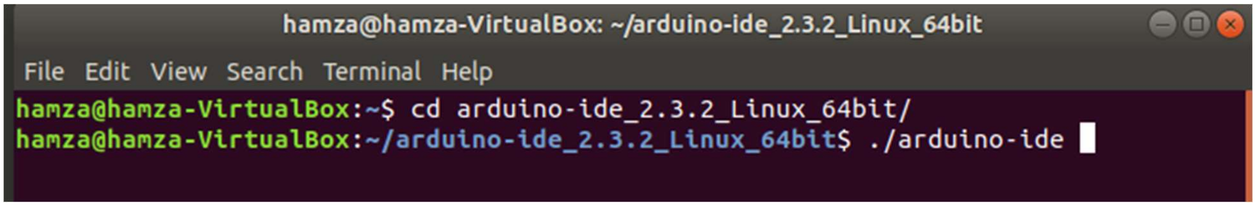
sudo apt install ros-melodic-rosserial-arduino

sudo apt install ros-melodic-rosserial-python

➔ Open Arduino IDE from terminal.

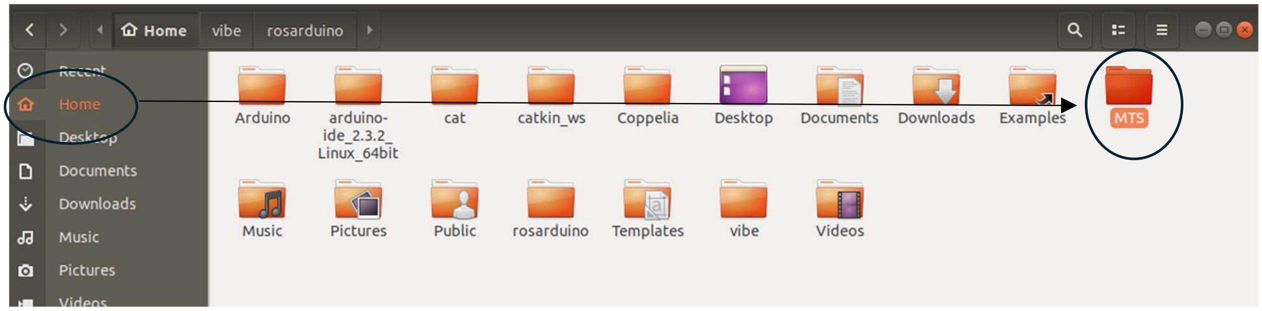
cd <Arduino folder>

./arduino-ide

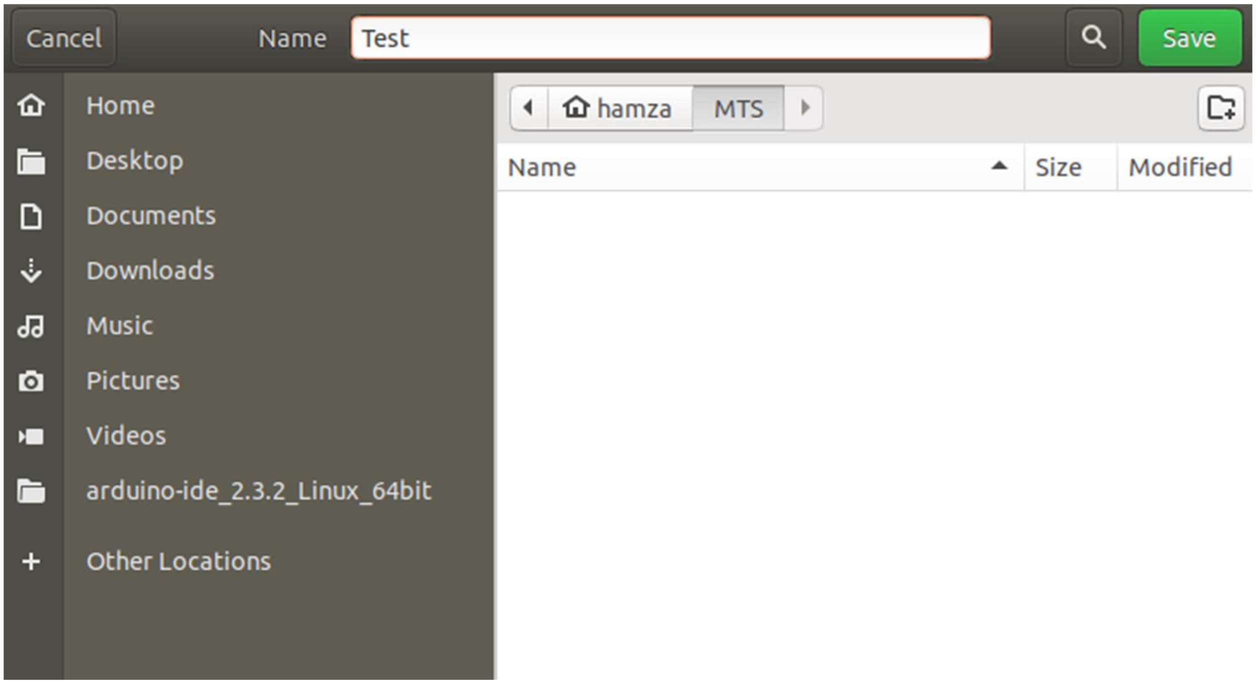

hamza@hamza-VirtualBox: ~/arduino-ide_2.3.2_Linux_64bit
File Edit View Search Terminal Help

```
hamza@hamza-VirtualBox:~$ cd arduino-ide_2.3.2_Linux_64bit/
hamza@hamza-VirtualBox:~/arduino-ide_2.3.2_Linux_64bit$ ./arduino-ide
```

➔ Make a new folder in home.

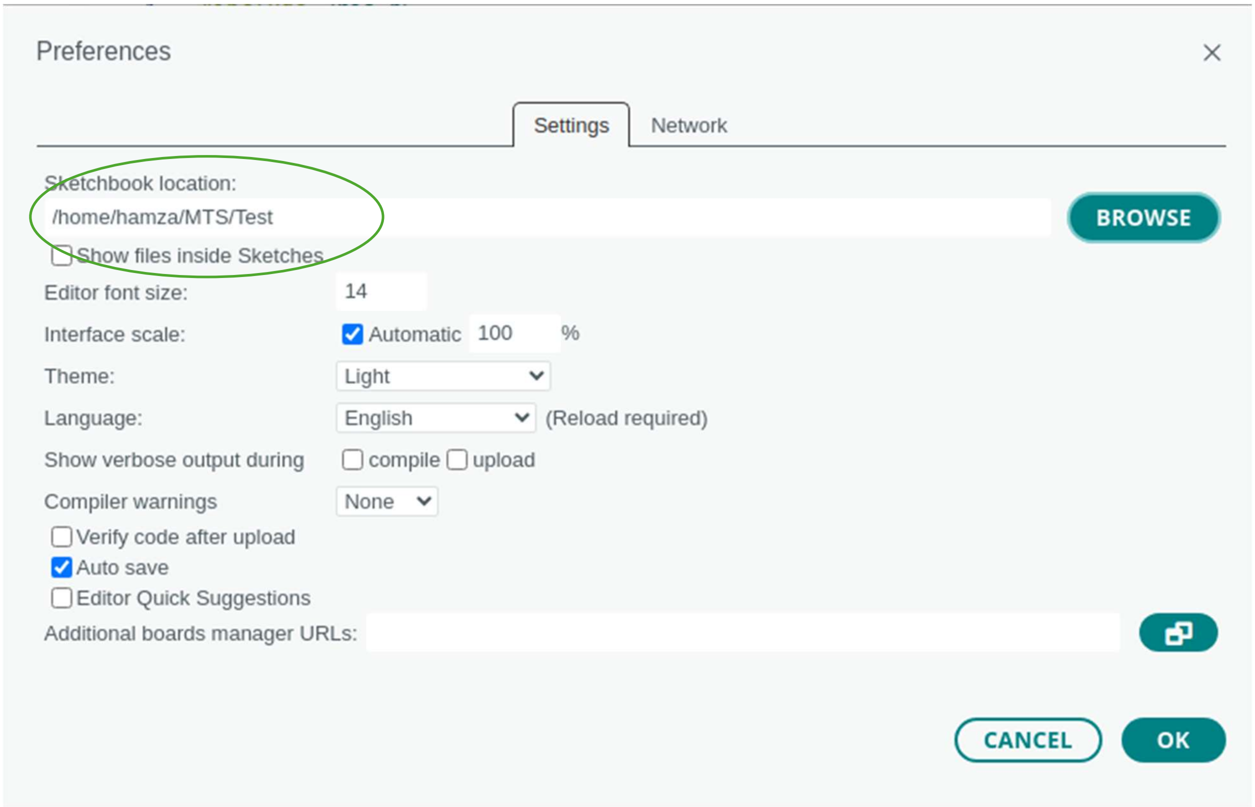


➔ Now save the Arduino sketch in this folder “MTS” and name the sketch as “TEST”



Name: Test

➔ Go to File → preferences and set the folder MTS.



Preferences

Sketchbook location: /home/hamza/MTS/Test

□ 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

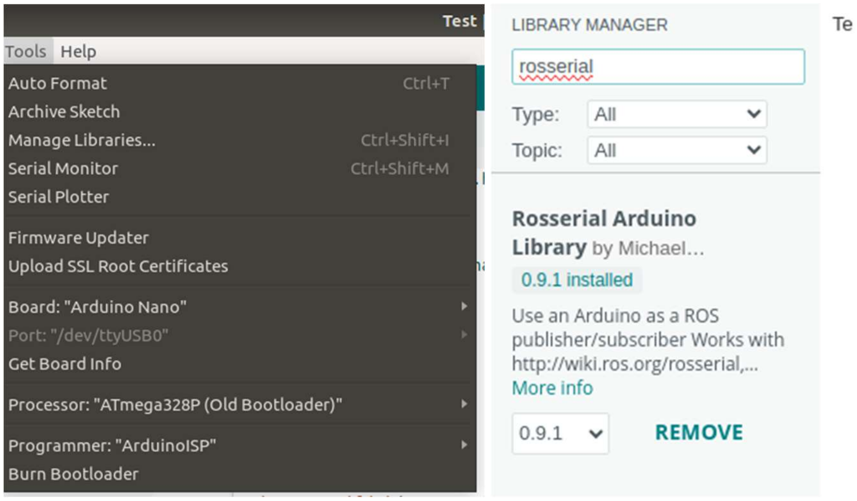
□ Verify code after upload

☑ Auto save

□ Editor Quick Suggestions

Additional boards manager URLs:

➔ Go to tools and manage libraries in Arduino IDE and install rosserial Arduino library.



rosserial

Type: All

Topic: All

Rosserial Arduino Library by Michael...

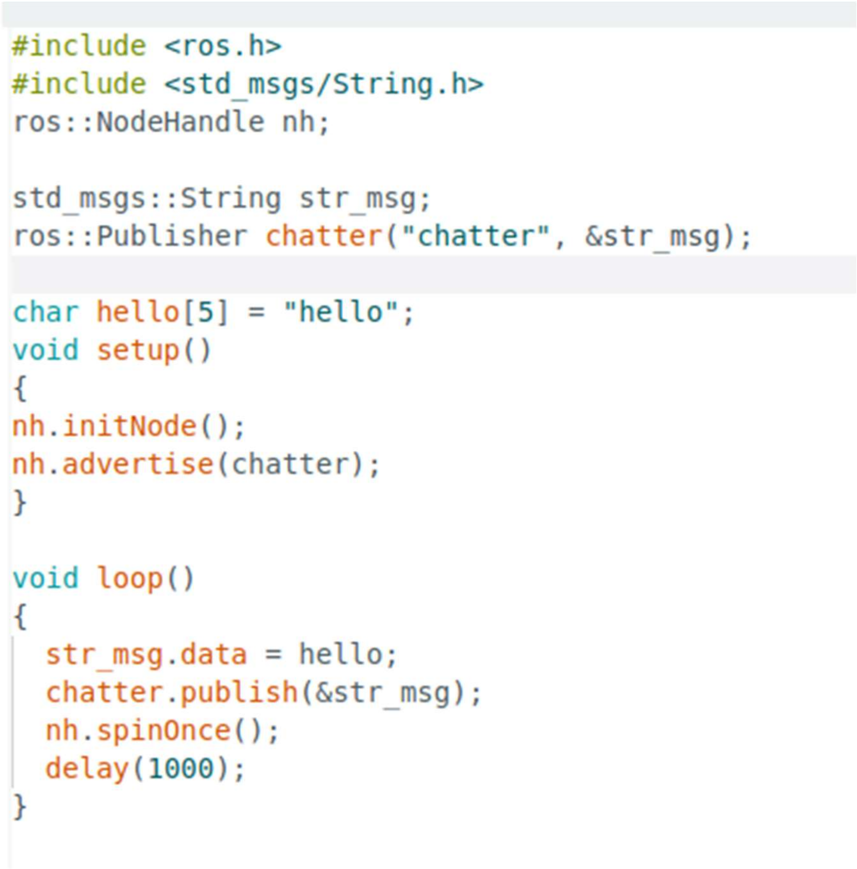
0.9.1 installed

Use an Arduino as a ROS publisher/subscriber Works with http://wiki.ros.org/rosserial,...

More info

0.9.1 REMOVE

➔ Write down the code.



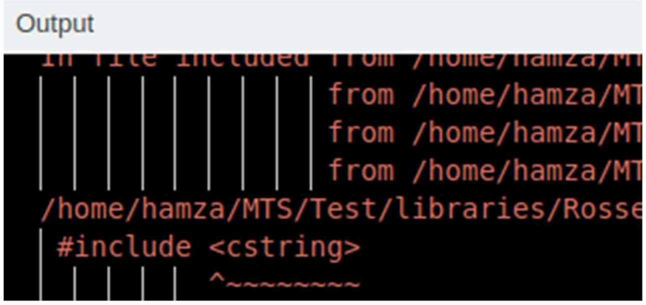
```
#include <ros.h>
#include <std_msgs/String.h>
ros::NodeHandle nh;

std_msgs::String str_msg;
ros::Publisher chatter("chatter", &str_msg);

char hello[5] = "hello";
void setup()
{
nh.initNode();
nh.advertise(chatter);
}

void loop()
{
  str_msg.data = hello;
  chatter.publish(&str_msg);
  nh.spinOnce();
  delay(1000);
}
```

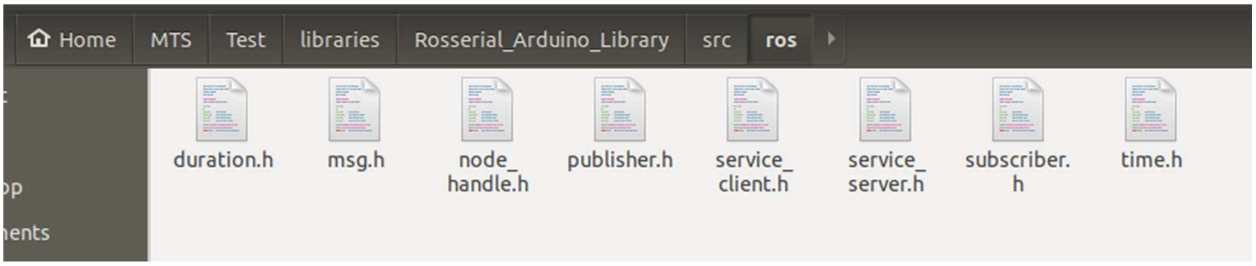
- ➔ Now upload it. You might see an error. "#include <cstring>



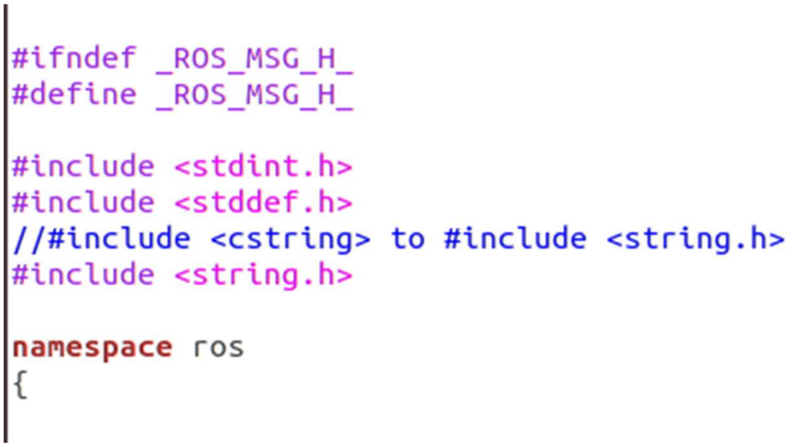
Output

```
  In file included from /home/hamza/MT
  | | | | | | | | |  from /home/hamza/MT
  | | | | | | | | |  from /home/hamza/MT
  | | | | | | | | |  from /home/hamza/MT
  /home/hamza/MTS/Test/libraries/Rosse
  | #include <cstring>
  | | | | |  ^~~~~~~~~
```

- ➔ To solve this do the following:
- ➔ Go to the folder in which you have set the preference and saved your sketch. In this case **MTS→libraries→Rosserial_arduino_library→src→ros→msg.h**



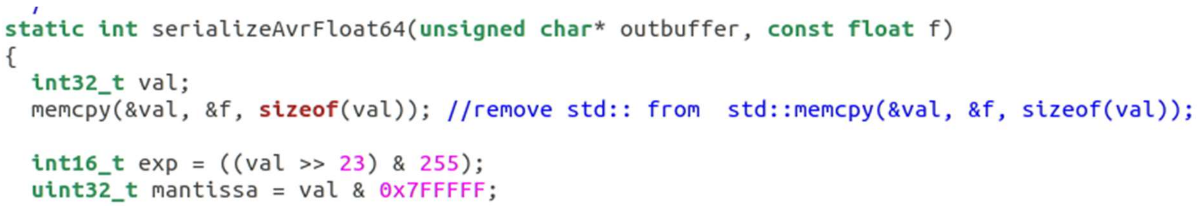
➔ In msg.h change the following:



```
#ifndef _ROS_MSG_H_
#define _ROS_MSG_H_

#include <stdint.h>
#include <stddef.h>
//#include <cstring> to #include <string.h>
#include <string.h>

namespace ros
{
```

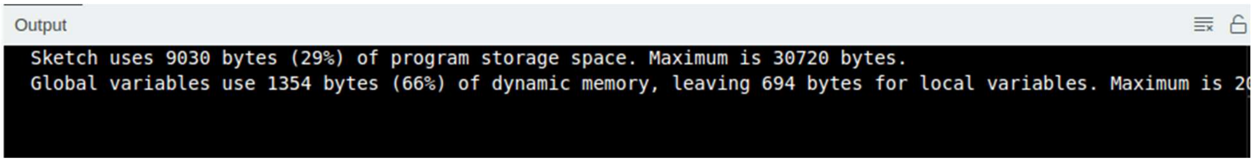


```
static int serializeAvrFloat64(unsigned char* outbuffer, const float f)
{
  int32_t val;
  memcpy(&val, &f, sizeof(val)); //remove std:: from  std::memcpy(&val, &f, sizeof(val));

  int16_t exp = ((val >> 23) & 255);
  uint32_t mantissa = val & 0x7FFFFF;
```

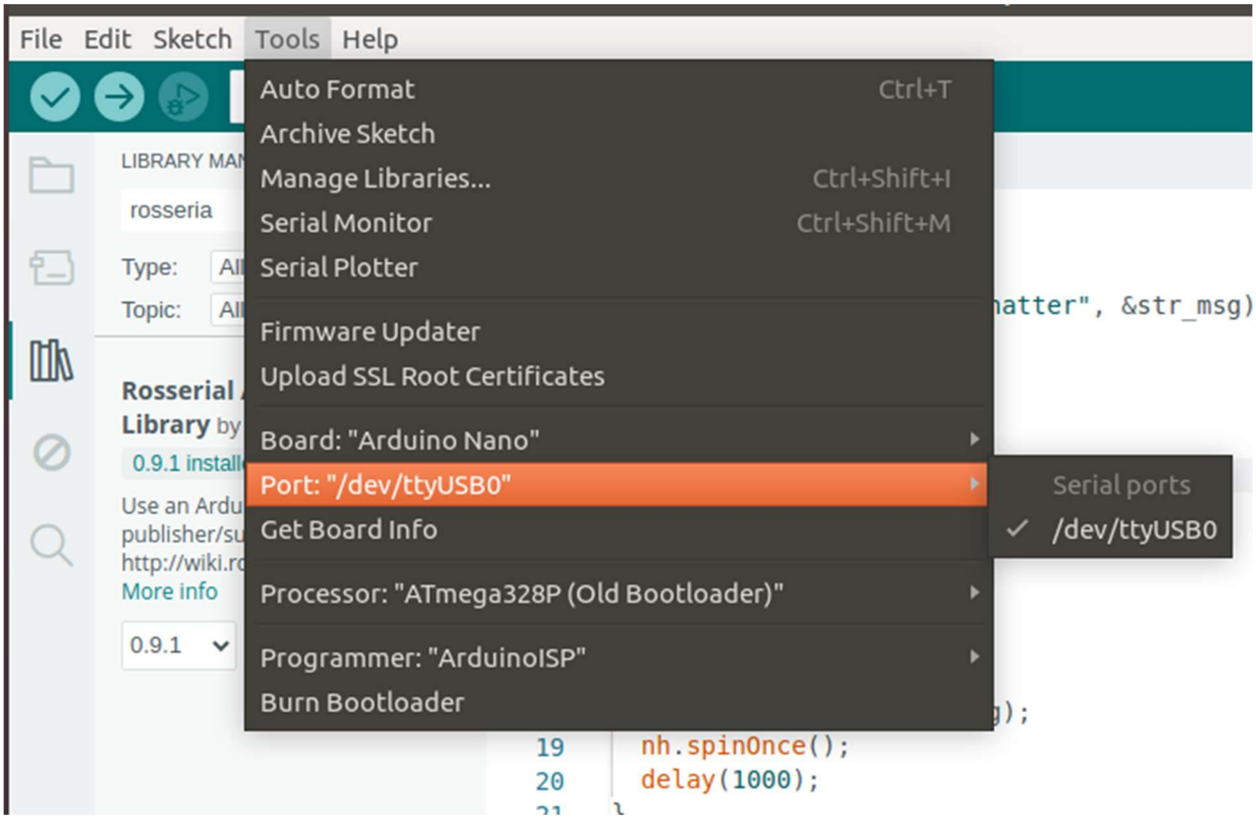
Make sure to remove std:: from two memcpy(---) and save the file

➔ Now try to compile the sketch.



```
Sketch uses 9030 bytes (29%) of program storage space. Maximum is 30720 bytes.
Global variables use 1354 bytes (66%) of dynamic memory, leaving 694 bytes for local variables. Maximum is 2
```

➔ Plug in the Arduino and check the port. It must be like /ttyUSB0 OR /ttyACM0



Board: "Arduino Nano"

Port: "/dev/ttyUSB0"

Processor: "ATmega328P (Old Bootloader)"

Programmer: "ArduinoISP"

```
atter", &str_msg)
g);
nh.spinOnce();
delay(1000);
```

- ➔ Open a new terminal and run the following command

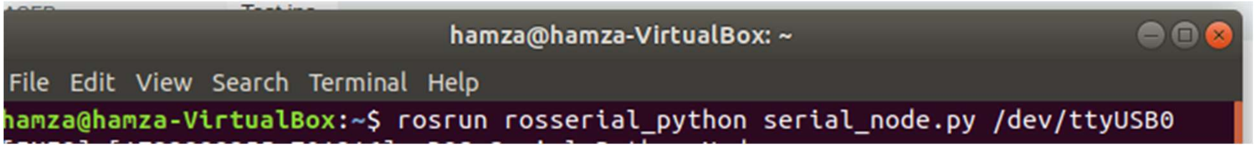
  roscore

- ➔ Unlock your port. It should only be done one time if you have already done it. Open a new terminal and write the following:

  sudo chmod 666 /dev/ttyUSB0

- ➔ Upload the code.

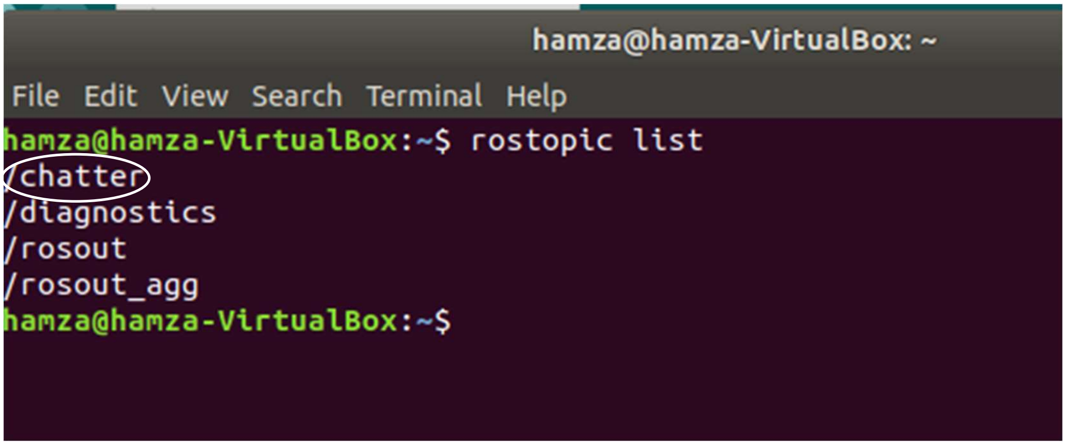
- ➔ Open a new terminal and run the following node.


hamza@hamza-VirtualBox: ~
File Edit View Search Terminal Help

```
hamza@hamza-VirtualBox:~$ rosrun rosserial_python serial_node.py /dev/ttyUSB0
```

- ➔ Open a new terminal and check if the node is in the list by the following command.

  rostopic list

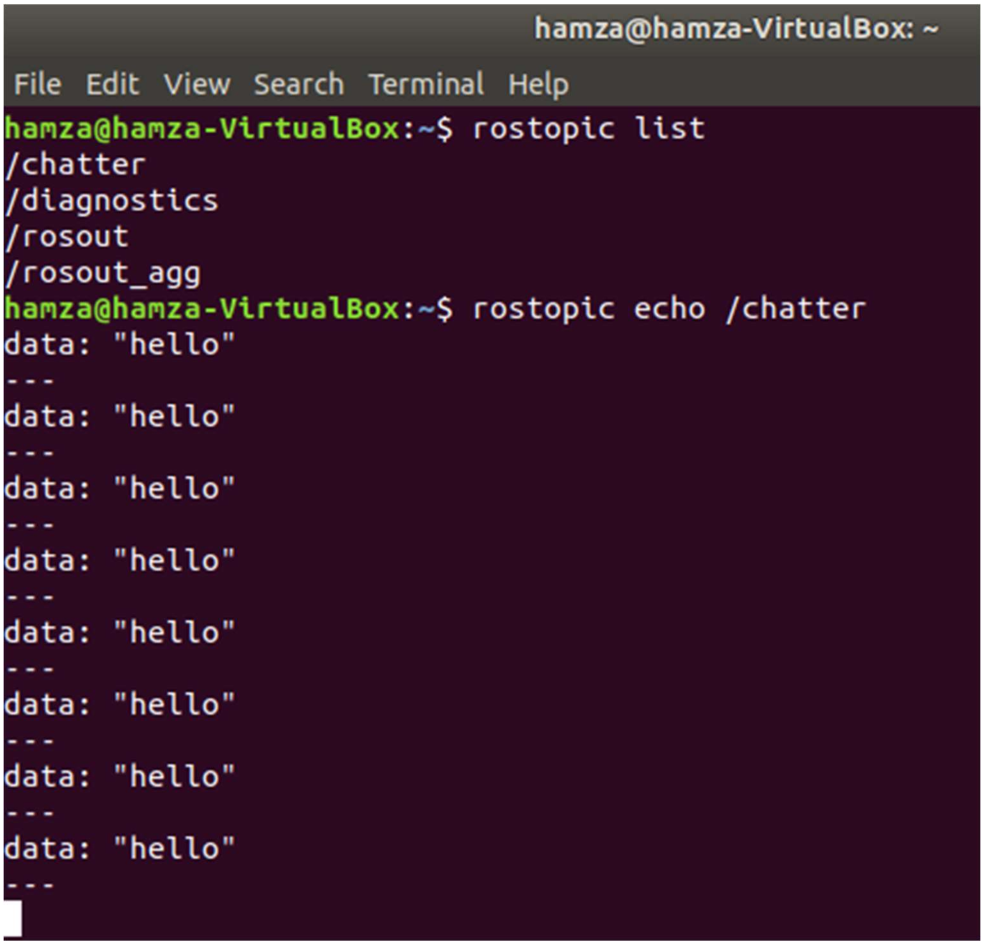


hamza@hamza-VirtualBox: ~
File Edit View Search Terminal Help

```
hamza@hamza-VirtualBox:~$ rostopic list
/chatter
/diagnostics
/rosout
/rosout_agg
hamza@hamza-VirtualBox:~$
```

- ➔ Now write in the same terminal.

  rostopic echo /chatter



hamza@hamza-VirtualBox: ~
File Edit View Search Terminal Help

```
hamza@hamza-VirtualBox:~$ rostopic list
/chatter
/diagnostics
/rosout
/rosout_agg
hamza@hamza-VirtualBox:~$ rostopic echo /chatter
data: "hello"
---
data: "hello"
---
data: "hello"
---
data: "hello"
---
data: "hello"
---
data: "hello"
---
data: "hello"
---
data: "hello"
---
```

The data is being published. Hence the connection between Arduino and ROS is successful.