Must read before use

After getting the car, you need to set the IP and ROS_DOMAIN_ID. The former is to connect to the proxy, and the latter is to achieve distributed multi-machine communication with the virtual machine.

1. Set IP

Take the matching virtual machine as an example, query the current IP of the virtual machine, input in the terminal,

```
ifconfig
```

```
yahboom@yahboom-VM:~$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:56:03:dd:d7  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.133  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::8757:4696:e812:c3e0  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:8d:3e:c2  txqueuelen 1000  (Ethernet)
        RX packets 714352  bytes 412836750 (412.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 772866  bytes 69435309 (69.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1016004  bytes 378626915 (378.6 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1016004  bytes 378626915 (378.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Here we find that our IP is 192.168.2.133, which is also the IP when we start the proxy, so we need to set the board to this IP to connect to the proxy.

Use a USB to Type-C cable to connect the board and the virtual machine, and make sure the virtual machine is connected to the board.

```
yahboom@yahboom-VM:~$ lsusb
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 004: ID 1a86:7522 QinHeng Electronics USB Serial
Bus 001 Device 003: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 001 Device 002: ID 0e0f:0003 VMware, Inc. Virtual Mouse
Bus 001 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
yahboom@yahboom-VM:~$
```

Then modify the config_Balance_Car.py code in the /home/yahboom directory and find the following part.

```
robot.set_udp_config([192, 168, 2, 108], 8899)
```

Change [192, 168, 2, 133] here to the IP you queried.
Also modify robot.set_ros_domain_id(20) and change the 20 inside to a custom number that cannot exceed 100.
After the modification is completed, save and exit, and run in the terminal,

```
python3 config_Balance_Car.py
```

```
yahboom@yahboom-VM:~$ python3 config_Balance_Car.py
Rebooting Device, Please wait.
version: 2.8.0
ssid: Yahboom2
passwd: yahboom890729
ip_addr: 192.168.2.108
ip_port: 8899
car_type: CAR_TYPE_COMPUTER
domain_id: 20
ros_serial_baudrate: 921600
ros_namespace:
Please reboot the device to take effect, if you change some device config.
```

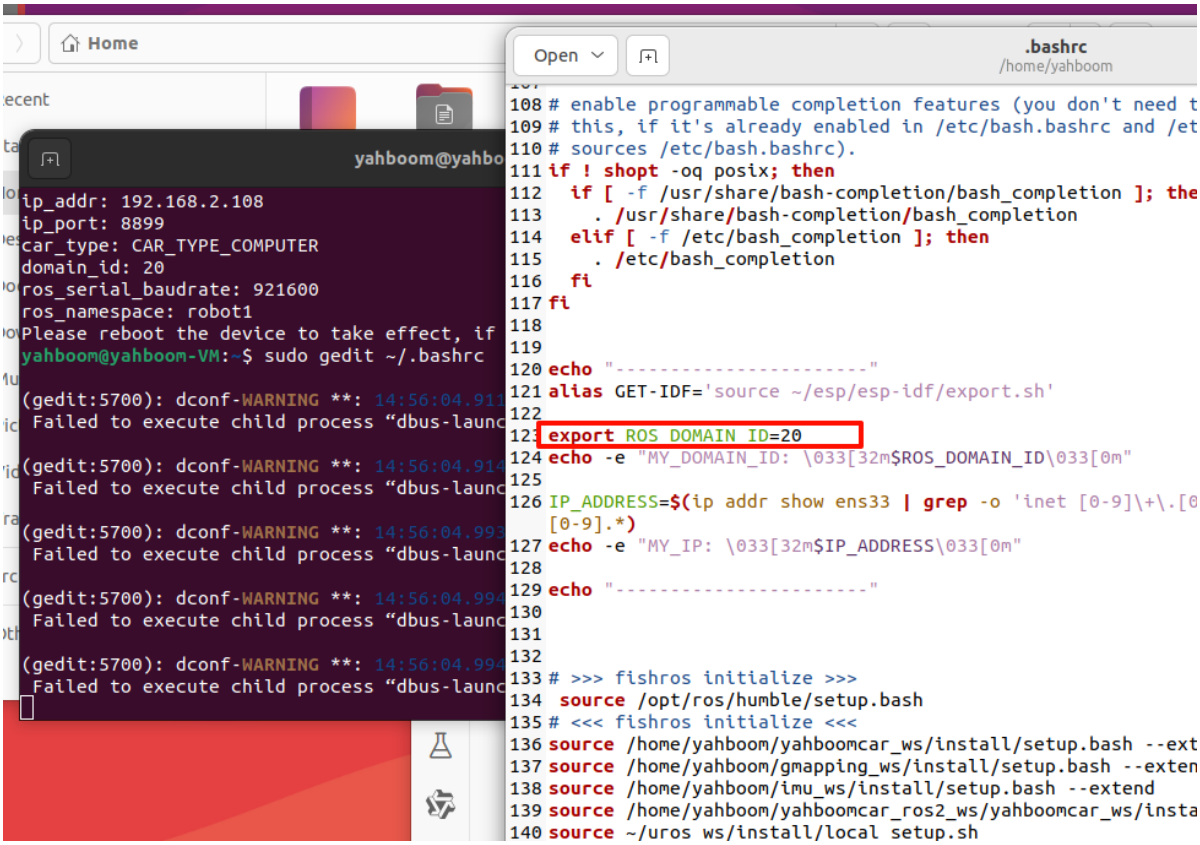
If the above screen appears, it means that the modification is successful. The content in the picture is subject to the actual modification. Then you need to restart the board to take effect.

2. Set ROS_DOMAIN_ID

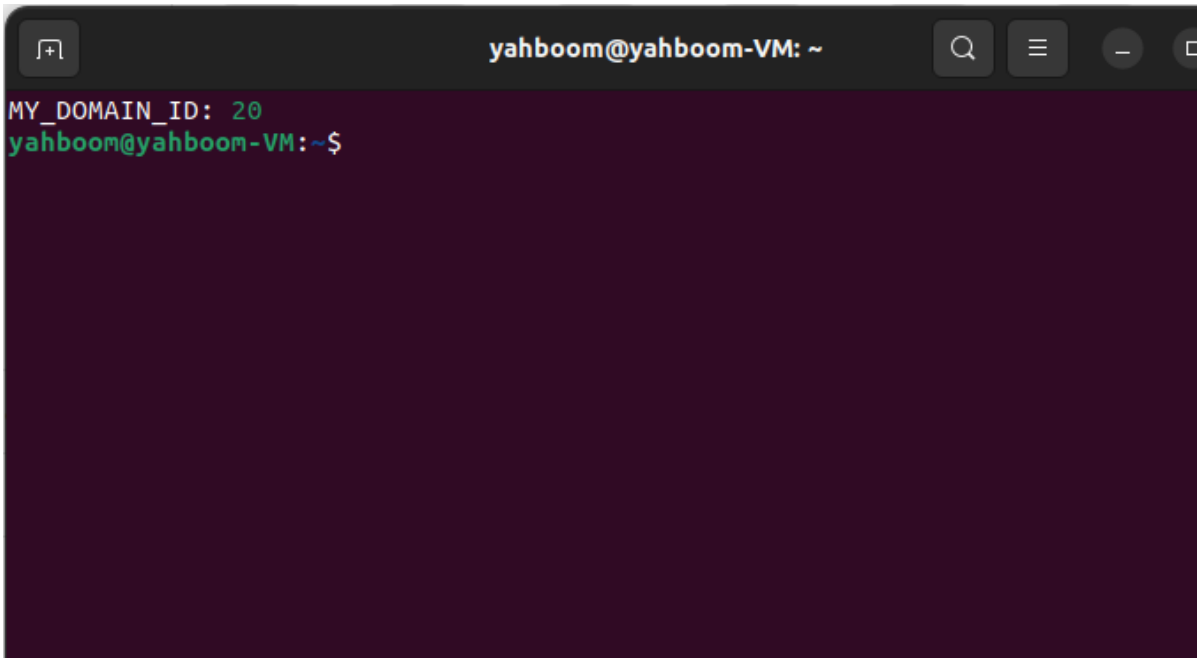
In 1, robot.set_ros_domain_id(20) is set to 20. Then you also need to set the same ROS_DOMAIN_ID in the ~/.bashrc file in the virtual machine to achieve distributed multi-machine communication between the board and the virtual machine.

Modify the ~/.bashrc file of the virtual machine and enter in the terminal,

```
sudo gedit ~/.bashrc
```



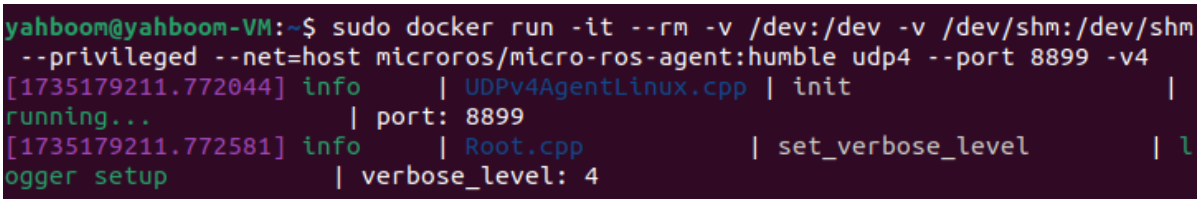
Find ROS_DOMAIN_ID and set it to be consistent with robot.set_ros_domain_id(20) set in 1. Assuming that it is set to 20, here
fill in 20. Then, save and exit, reopen the terminal, and the terminal will display the set ROS_DOMAIN_ID value.



3. Test whether the modification is complete

Enter the following command in the virtual machine terminal to start the agent,

```
sudo docker run -it --rm -v /dev:/dev -v /dev/shm:/dev/shm --privileged --
net=host microros/micro-ros-agent:humble udp4 --port 8090 -v4
```



Then, turn on the car switch and wait for the car to connect to the proxy. The connection is successful as shown in the figure below.

```
[1735179211.772044] info     | UDPv4AgentLinux.cpp | init                    | running...            | port: 8899
[1735179211.772581] info     | Root.cpp            | set_verbose_level       | logger setup          | verbose_level: 4
[1735179325.739277] info     | Root.cpp            | create_client           | create                | client_key: 0x0E5C3397, sess
ion_id: 0x81
[1735179325.739348] info     | SessionManager.hpp | establish_session        | session established   | client_key: 0x0E5C3397, addr
ess: 192.168.2.102:49954
[1735179325.971694] info     | ProxyClient.cpp     | create_participant      | participant created   | client_key: 0x0E5C3397, part
icipant_id: 0x000(1)
[1735179326.046043] info     | ProxyClient.cpp     | create_topic            | topic created         | client_key: 0x0E5C3397, topi
c_id: 0x000(2), participant_id: 0x000(1)
[1735179326.159287] info     | ProxyClient.cpp     | create_publisher        | publisher created     | client_key: 0x0E5C3397, publ
isher_id: 0x000(3), participant_id: 0x000(1)
[1735179326.176344] info     | ProxyClient.cpp     | create_datawriter       | datawriter created    | client_key: 0x0E5C3397, data
writer_id: 0x000(5), publisher_id: 0x000(3)
[1735179326.184566] info     | ProxyClient.cpp     | create_topic            | topic created         | client_key: 0x0E5C3397, topi
c_id: 0x001(2), participant_id: 0x000(1)
[1735179326.263761] info     | ProxyClient.cpp     | create_publisher        | publisher created     | client_key: 0x0E5C3397, publ
isher_id: 0x001(3), participant_id: 0x000(1)
[1735179326.276817] info     | ProxyClient.cpp     | create_datawriter       | datawriter created    | client_key: 0x0E5C3397, data
writer_id: 0x001(5), publisher_id: 0x001(3)
[1735179326.285996] info     | ProxyClient.cpp     | create_topic            | topic created         | client_key: 0x0E5C3397, topi
c_id: 0x002(2), participant_id: 0x000(1)
[1735179326.345401] info     | ProxyClient.cpp     | create_publisher        | publisher created     | client_key: 0x0E5C3397, publ
isher_id: 0x002(3), participant_id: 0x000(1)
[1735179326.365619] info     | ProxyClient.cpp     | create_datawriter       | datawriter created    | client_key: 0x0E5C3397, data
writer_id: 0x002(5), publisher_id: 0x002(3)
[1735179326.372863] info     | ProxyClient.cpp     | create_topic            | topic created         | client_key: 0x0E5C3397, topi
c_id: 0x003(2), participant_id: 0x000(1)
[1735179326.379913] info     | ProxyClient.cpp     | create_publisher        | publisher created     | client_key: 0x0E5C3397, publ
isher_id: 0x003(3), participant_id: 0x000(1)
[1735179326.448851] info     | ProxyClient.cpp     | create_datawriter       | datawriter created    | client_key: 0x0E5C3397, data
writer_id: 0x003(5), publisher_id: 0x003(3)
[1735179326.548363] info     | ProxyClient.cpp     | create_topic            | topic created         | client_key: 0x0E5C3397, topi
c_id: 0x004(2), participant_id: 0x000(1)
[1735179326.565153] info     | ProxyClient.cpp     | create_subscriber       | subscriber created    | client_key: 0x0E5C3397, subs
criber_id: 0x000(4), participant_id: 0x000(1)
[1735179326.574254] info     | ProxyClient.cpp     | create_datareader       | datareader created    | client_key: 0x0E5C3397, data
reader_id: 0x000(6), subscriber_id: 0x000(4)
```

Then reopen a terminal and enter the following command to query the currently running node.

```
yahboom@yahboom-VM: ~
yahboom@yahboom-VM:~$ ros2 node list
/YB_BalanceCar_Node
yahboom@yahboom-VM:~$
```

When the car connects to the proxy, a node program will run. If the node can be queried on the virtual machine, it means that the two have achieved distributed multi-machine communication.