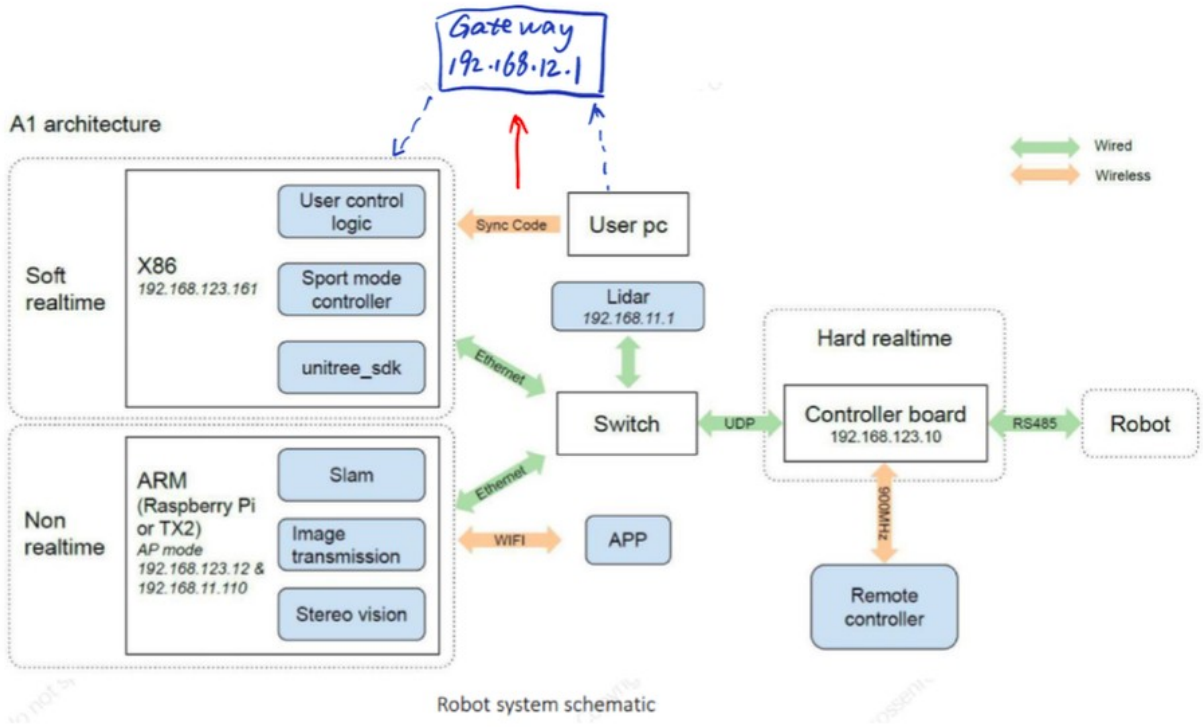
Unitree A1

Wireless/Remote PC Control

This is the architecture of the Unitree A1 modules. There're two master boards integrated in it. One is of X86 SoC for executing the real-time tasks like high level control and another is of ARM SoC for performing non-real time jobs like SLAM, Image Transmission, etc. Internally, they both communicate with low-level controller board through a media called Switch. All the IP addresses are static.



The document doesn't have an explicit illustration for connecting between A1 and PC but use a Sync Code for replacement. Exploring the robot dog system, I found that PC communicates with the dog via a default Gateway with IP address as 192.168.12.1 which is essential for interacting with the controller board. The reason as to why PC cannot send the command to the dog for execution through wireless mode is usually because there's a firewall which sets the rule for filtering the outer packets by default. Then all you need is to reset the firewall on dog system to make it work. Below is the high-level overview of the system.

Practically, you'll have to connect the hotspot of A1 after booting it. Then connect the system using command with password **123**:

$ ssh unitree@192.168.12.1

Then configure the system settings with command:

$ sudo vim /etc/sysctl.conf

Make sure the line **net.ipv4.ip_forward=1** is not commented. After that, you can reconfigure the firewall using tool iptables as follows:

$ sudo sysctl -p
$ sudo iptables -F
$ sudo iptables -t nat -F
$ sudo iptables -t nat -A POSTROUTING -o wlan1 -j MASQUERADE
$ sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
$ sudo iptables -A FORWARD -i wlan1 -o eth0 -j ACCEPT
$ sudo iptables -A FORWARD -i eth0 -o wlan1 -j ACCEPT

Finally, set the router to Gateway 192.168.12.1 on your own PC:

$ sudo route add default gw 192.168.12.1

Test whether you can get access to the low-level controller on your own PC:

$ ping 192.168.123.10

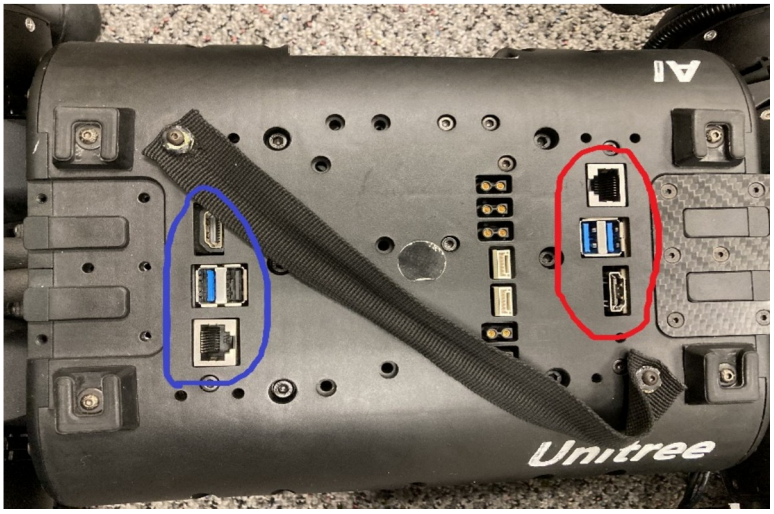
If success, then you should be able to run the code locally on your PC while in the meantime execute it on the robot dog via wireless mode. You can firstly test examples in legged_SDK to see whether it works.

In this time, your PC will keep connecting to the dog's hotspot in which case you may lose access to the Internet. If you want to both perform task and access to the internet, you can insert a network adaptor as a secondary wifi module. (But I don't recommend this because it may impact the dog connection during runtime…)

Permanent Configuration

After settings above, you're able to connect to A1 and control it remotely by running the code on your PC. But it will restore to its original settings once rebooting the quadruped robot. That means you should reconfigure the firewall rules every time you boot the dog, which is inconvenient. Here are the instructions to make a permanent configuration for it:

1). Plugin the display monitor and keyboard to the Nvidia ARM board (see below picture, the **blue region** is the **x86_64 UP-CHT01 board** and the **red region** belongs to the **Nvidia ARM NX board**)



2). Boot the robot dog, close its inherent hotspot (Router Mode), and turn on the network manager (AP Mode) by running the command:

$ sudo systemctl start network-manager.service

After that, a Wi-Fi icon should appear on the upper right corner of the display (If not, feel free to connect Wi-Fi using the tool **nmcli**). Then connect to the network with password. You can also choose to connect to the internet with Ethernet cable.

Specifically, if you still cannot get access to the internet while the Wi-Fi/Ethernet is connected, check the file **/etc/resolv.conf** to see whether there's any additional lines apart from **nameserver 127.0.0.1** in it. If it is, add available DNS servers below it (e.g. nameserver 8.8.8.8).

3). Obtaining access to the internet, install the tool **iptables-persistent** by:

$ sudo apt update

$ sudo apt install iptables-persistent

4). Follow the first part in this document to do all the iptables configurations. Save them after completion by:

$ sudo iptables-save > /etc/iptables/rules.v4

Also make sure **netfilter-persistent.service** gets started and enabled after booting:

$ sudo systemctl start netfilter-persistent.service

$ sudo systemctl enable netfilter-persistent.service

From now your iptables configurations should be kept permanently. Reboot the dog to see whether it takes an effect. If you still have questions regarding the iptables permanent configuration, please refer to this article for details.