1. Introduction of voice module

1.1. Introduction to the voice module

1.1.1. CSK4002 chip

The voice control module on ROSMASTER is developed based on the **CSK4002** chip. **CSK4002** is an AISoC with higher performance, better computing power, lower power consumption and designed for the AIoT field. It can be widely used in smart home, smart home appliances, and emerging consumer electronics industries.

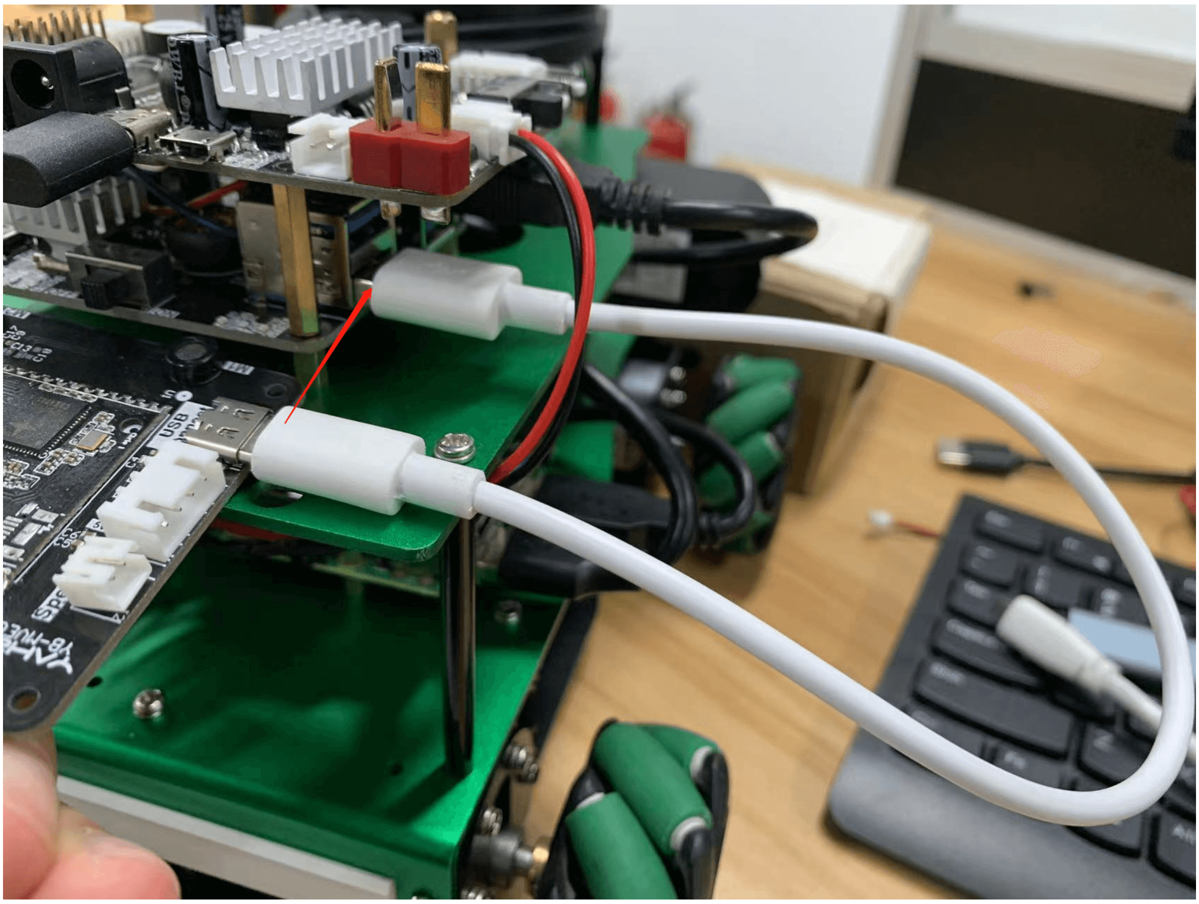
- CSK4002 adopts Andes D1088 core, and its AI/DSP acceleration module MVA supports a variety of Neural Network operators and vector operations, and is deeply adapted to iFLYTEK AI algorithms, with a computing power of up to 128GOPS.
- Comes with 8M Flash, 8M PSRAM, 1M SRAM.
- Supports 8 channels of PDM audio input and 16 channels of I2S Audio Input data processing.
- Integrated rich mainstream peripheral interfaces: GPIO/UART/I2C/SPI/QSPI/SDIO/USB1.1/SDIO, etc.
- Equipped with low-latency embedded operating system Free RTOS, complete BSP driver, and complete development tool resources.

1.2. Using the voice module

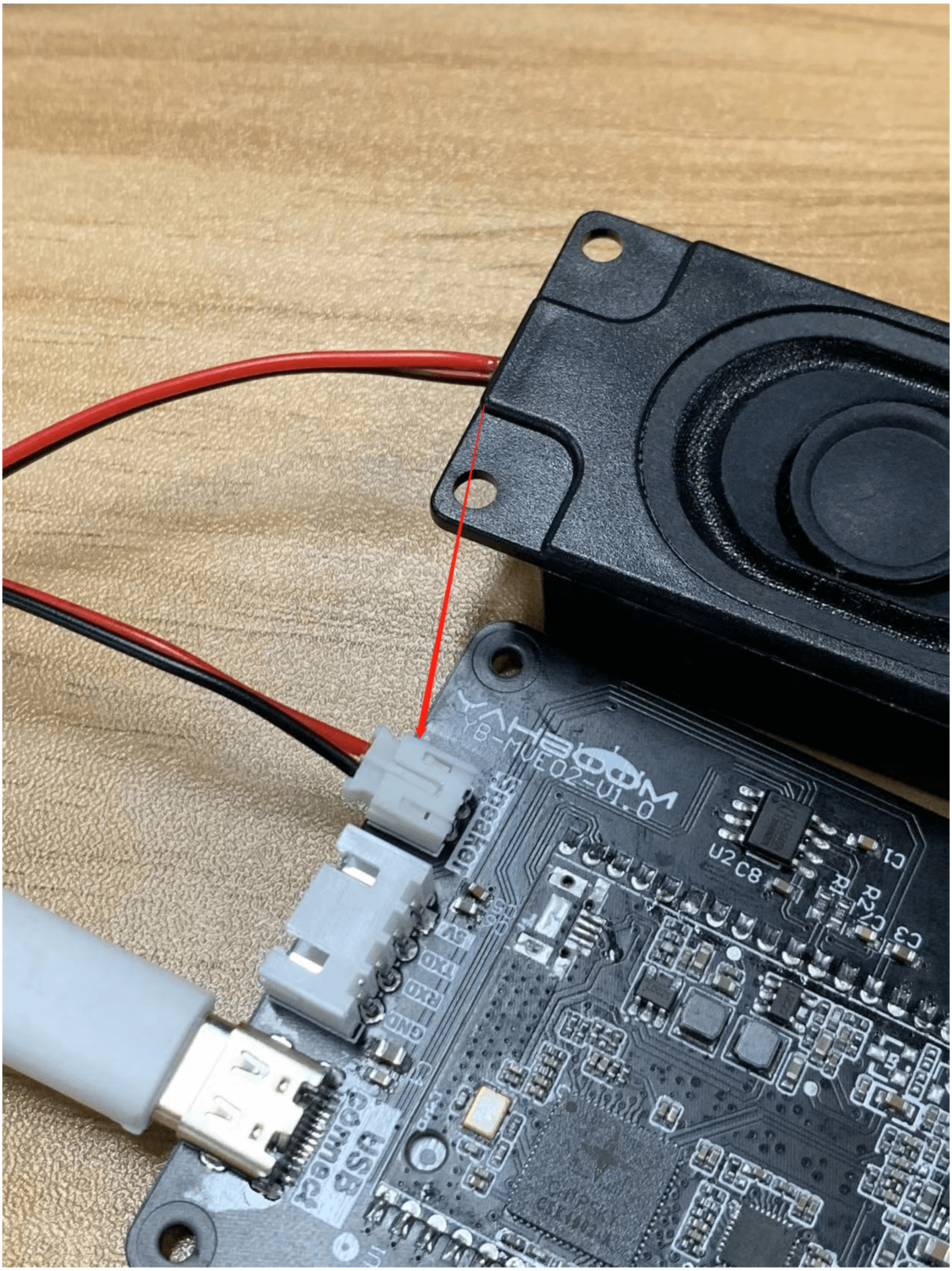
1.2.1. Wiring

The module is connected to the ROSMASTER robot (or HUB board) through a Type-c data cable, and the speaker and the module are connected through a PH2.0 data cable.

- As shown below, one end of the Micro-USB data cable is connected to the "Program" port of the module, and the other end is connected to the port of the ROSMASTER main control board (**after this is plugged in, it cannot be modified after the rear port is bound).** About this, please check the next course).



- As shown below, the PH2.0 data cable port is connected to the Speaker port on the module.



1.2.2. Wake-up word

The wake-up word is "Hi, Yahboom". When you wake up, you need to slow down your speech rate. If it is too fast, the module will not recognize it. After the module is woken up, other command words can be recognized later.

Within 20 seconds of waking up, you don't need to wake up again, just say the command word directly.

1.2.3. Command words

1.Voice control car movement

speech recognition content	The voice module is sent to the host	The host sends to the voice module	Voice broadcast content
Robot stop	$B002#	$A002#	OK , I'm stop.
Go ahead	$B004#	$A004#	OK , let's go.
Back	$B005#	$A005#	OK , I‘m back.
Turn left	$B006#	$A006#	OK , I‘m turning left.
Turn right	$B007#	$A007#	OK , I‘m turning right.
Enter A mode	$B008#	$A008#	OK, I‘m working on A mode.
Enter B mode	$B009#	$A009#	OK, I‘m working on B mode.

2.Voice control RGB light strip effect

speech recognition content	The voice module is sent to the host	The host sends to the voice module	Voice broadcast content
Close light	$B010#	$A010#	OK, light is closed.
Red light up	$B011#	$A011#	OK, red light is on.
Green light up	$B012#	$A012#	OK, green light is on.
Blue light up	$B013#	$A013#	OK, blue light is on.
Yellow light up	$B014#	$A014#	OK, yellow light is on.
light A	$B015#	$A015#	OK, light A is on.
light B	$B016#	$A016#	OK, light B is on.
light C	$B017#	$A017#	OK, light C is on.
Display battery value	$B018#	$A018#	OK, battery value has been display.

3.Voice control color recognition

speech recognition content	The voice module is sent to the host	The host sends to the voice module	Voice broadcast content
What color is this?	$B060#	$A061#	This is red
What color is this?	$B060#	$A062#	This is blue
What color is this?	$B060#	$A063#	This is green
What color is this?	$B060#	$A064#	This is yellow

4.Voice control color tracking

speech recognition content	The voice module is sent to the host	The host sends to the voice module	Voice broadcast content
yellow following	$B072#	$A072#	OK, I found the yellow
red following	$B073#	$A073#	OK, I found the red
green following	$B074#	$A074#	OK, I found the green
follow this color	$B075#	$A075#	OK, I found this color
stop following	$B076#	$A076#	OK, it has been stoped

5.Voice control automatic driving (line patrol)

speech recognition content	The voice module is sent to the host	The host sends to the voice module	Voice broadcast content
Close tracking mode	$B022#	$A022#	OK, tracking mode is closed
Tracking the red line	$B023#	$A023#	OK, I will track the red line
Tracking the green line	$B024#	$A024#	OK, I will track the green line
Tracking the blue line	$B025#	$A025#	OK, I will track the blue line
Tracking the yellow line	$B026#	$A026#	OK, I will track the yellow line

6.Voice control multi-point navigation

speech recognition content	The voice module is sent to the host	The host sends to the voice module	Voice broadcast content
Go to the point A	$B019#	$A019#	OK, I'm going to the point A.
Go to the point B	$B020#	$A020#	OK, I'm going to the point B.
Go to the point C	$B021#	$A021#	OK, I'm going to the point C.
Go to the point D	$B022#	$A022#	OK, I'm going to the point D.
Return to the original place	$B023#	$A023#	OK, I'm return back.

7.Basic voice control of robotic arm

speech recognition content	he voice module is sent to the host	The host sends to the voice module	Voice broadcast content
Warning	$B038#	$A038#	OK
Lift the arm up	$B039#	$A039#	OK , the arm has been raised
Put the arm down	$B040#	$A040#	OK, the arm is down
Arm left	$B041#	$A041#	OK, the arm has turned left
Arm right	$B042#	$A042#	OK, the arm has turned right
Clamp the clip	$B043#	$A043#	OK, the clip is clamped
Open the clip	$B044#	$A044#	OK, the clip has been opened.

8. control of robotic arm

speech recognition content	he voice module is sent to the host	The host sends to the voice module	Voice broadcast content
Action A	$B045#	$A045#	OK

speech recognition content	he voice module is sent to the host	The host sends to the voice module	Voice broadcast content
Action B	$B046#	$A045#	OK
Action C	$B047#	$A045#	OK
Action D	$B048#	$A045#	OK
Reset	$B049#	$A045#	OK
Action E	$B050#	$A045#	OK

9. action of the robotic arm

speech recognition content	he voice module is sent to the host	The host sends to the voice module	Voice broadcast content
Dancing	$B052#	$A052#	OK, Let's dance
Clip the block	$B053#	$A053#	OK,let me clip them

10. robotic arm memory learning

speech recognition content	he voice module is sent to the host	The host sends to the voice module	Voice broadcast content
record it	$B055#	$A055# $A056#	OK, please set the next action. Too many actions, I can't record them
Now is over	$B056#	$A057#	OK,Noted it,I've recorded them.
Display actions	$B057#	$A058#	OK, Let's do it
Clear actions	$B058#	$A059#	OK, actions have been cleared

11. Sort color block

speech recognition content	he voice module is sent to the host	The host sends to the voice module	Voice broadcast content
Sort color block	$B061#	$A065#	It has been in its place
		$A066#	This is yellow

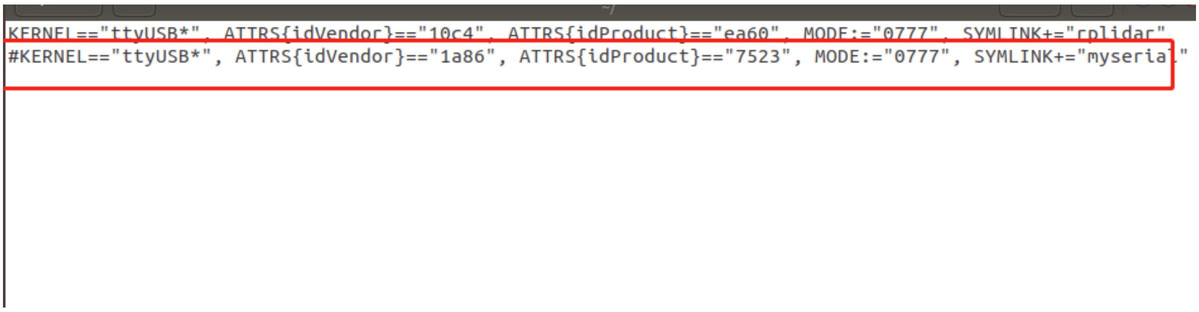
speech recognition content	he voice module is sent to the host	The host sends to the voice module	Voice broadcast content
		$A067#	This is green
		$A068#	This is blue
		$A069#	This is red

2. Voice control module port binding

Since the ID device number of the ROS expansion board and the voice control module are the same, it is not possible to bind the ID device number according to the method in the previous tutorial. **Unbound ports may cause port conflicts or device identification errors. Do not change the positions of bound ports at will. Otherwise, the bound ports become invalid**.

This course is completed on the host, no docke binding is required, and this section uses the Raspberry PI as the host to demonstrate. Before binding, it is necessary to comment out the part involving myserial in the host rule file, otherwise the voice board and ROS extension board will be identified as the same device.

```
sudo gedit /etc/udev/rules.d/usb.rules
```



```
KERNEL=="ttyUSB*", ATTRS{idVendor}=="10c4", ATTRS{idProduct}=="ea60", MODE:="0777", SYMLINK+="rplidar"
#KERNEL=="ttyUSB*", ATTRS{idVendor}=="1a86", ATTRS{idProduct}=="7523", MODE:="0777", SYMLINK+="myserial"
```

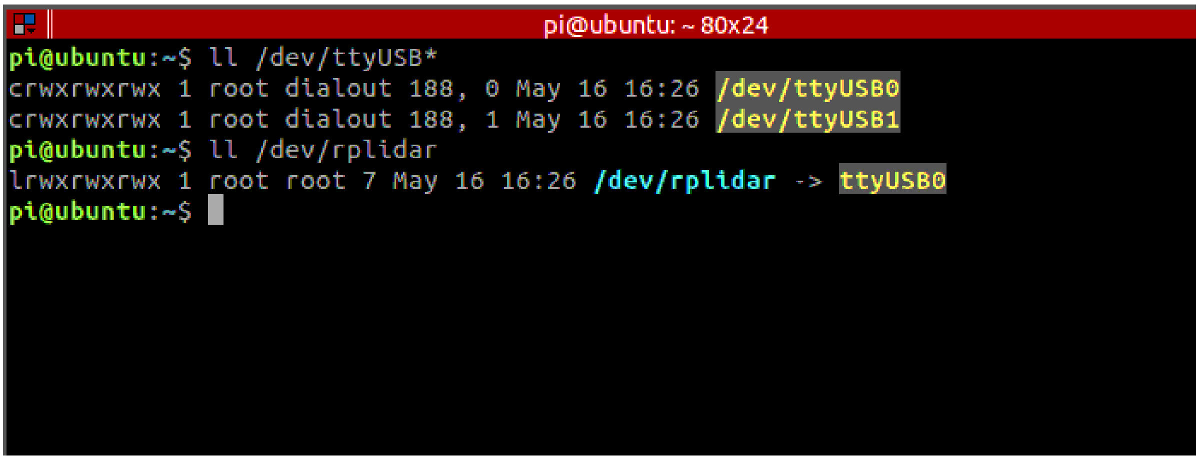
As shown in the figure above, the myserial related binding content is commented, saved and quit, and the terminal enters the following command to reload the rule.

```
sudo udevadm trigger
sudo service udev reload
sudo service udev restart
```

2.1. Bind the device of the ROS expansion board

```
ll /dev/ttyUSB*
ll /dev/rplidar
```

- First, ttyUSB0 and ttyUSB1 can be obtained without accessing the voice control board



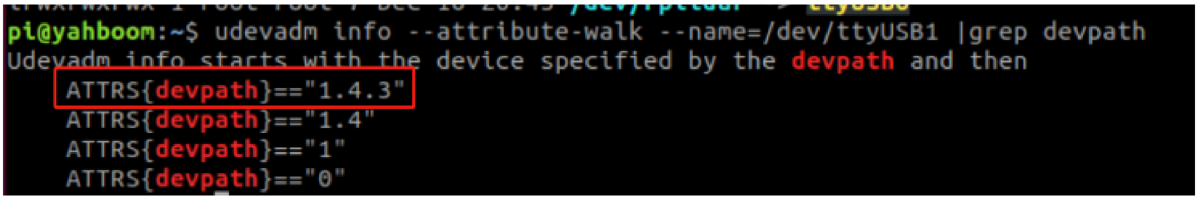
```
pi@ubuntu:~$ ll /dev/ttyUSB*
crwxrwxrwx 1 root dialout 188, 0 May 16 16:26 /dev/ttyUSB0
crwxrwxrwx 1 root dialout 188, 1 May 16 16:26 /dev/ttyUSB1
pi@ubuntu:~$ ll /dev/rplidar
lrwxrwxrwx 1 root root 7 May 16 16:26 /dev/rplidar -> ttyUSB0
pi@ubuntu:~$
```

ttyUSB0 is the radar device , Then ttyUSB1 is the device of the ROS expansion board

- Second, we check the port information of the ROS expansion board, mainly to check the device path information and terminal input.

```
udevadm info --attribute-walk --name=/dev/ttyUSB1 |grep devpath
```

The following information is obtained (based on actual conditions). The red box indicates the path information of the device.



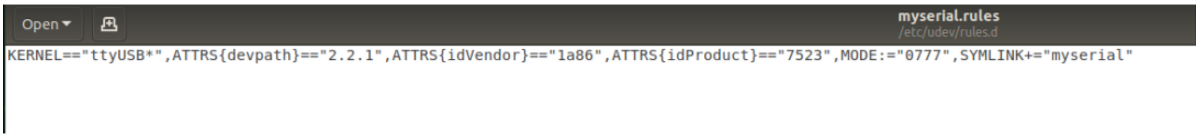
```
pi@yahboom:~$ udevadm info --attribute-walk --name=/dev/ttyUSB1 |grep devpath
Udevadm info starts with the device specified by the devpath and then
    ATTRS{devpath}=="1.4.3"
    ATTRS{devpath}=="1.4"
    ATTRS{devpath}=="1"
    ATTRS{devpath}=="0"
```

- Third, We modify (/etc/udev/rules.d/rplidar.rules) this file, bind the port number of the ROS expansion board, and enter the terminal

```
sudo gedit /etc/udev/rules.d/myserial.rules
```

Add content as shown below

```
KERNEL=="ttyUSB*",ATTRS{devpath}=="1.4.3",ATTRS{idVendor}=="1a86",ATTRS{idProduct
}=="7523",MODE:="0777",SYMLINK+="myserial"
```



```
KERNEL=="ttyUSB*",ATTRS{devpath}=="2.2.1",ATTRS{idVendor}=="1a86",ATTRS{idProduct}=="7523",MODE:="0777",SYMLINK+="myserial"
```

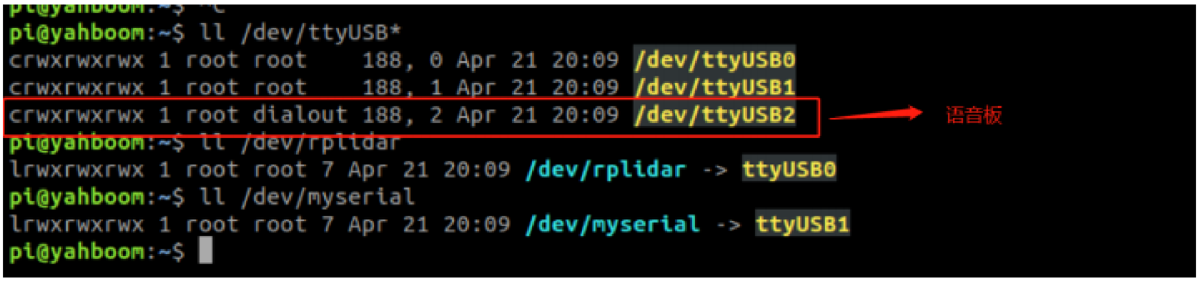
- Last, Save and exit. Run the following three commands to reload the device

```
sudo udevadm trigger
sudo service udev reload
sudo service udev restart
```

2.2. bind the device of the voice board

- To access the voice board, run the following command on the terminal to view the device number

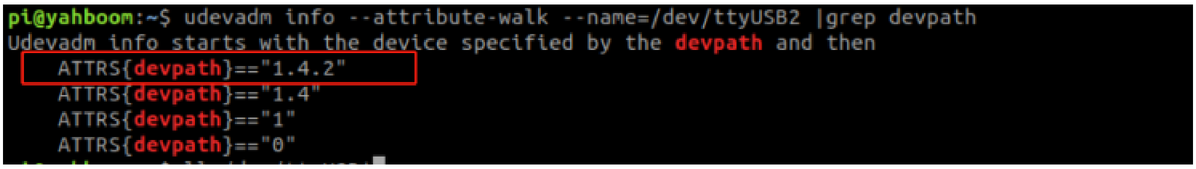
```
ll /dev/ttyUSB*
```



```
pi@yahboom:~$ ^C
pi@yahboom:~$ ll /dev/ttyUSB*
crwxrwxrwx 1 root root    188, 0 Apr 21 20:09 /dev/ttyUSB0
crwxrwxrwx 1 root root    188, 1 Apr 21 20:09 /dev/ttyUSB1
crwxrwxrwx 1 root dialout 188, 2 Apr 21 20:09 /dev/ttyUSB2
pi@yahboom:~$ ll /dev/rplidar
lrwxrwxrwx 1 root root 7 Apr 21 20:09 /dev/rplidar -> ttyUSB0
pi@yahboom:~$ ll /dev/myserial
lrwxrwxrwx 1 root root 7 Apr 21 20:09 /dev/myserial -> ttyUSB1
pi@yahboom:~$
```

Here we find that the voice board is identified by the system as /dev/ttyUSB2, so we enter the following command to view the path of the device

```
udevadm info --attribute-walk --name=/dev/ttyUSB2 |grep devpath
```



```
pi@yahboom:~$ udevadm info --attribute-walk --name=/dev/ttyUSB2 |grep devpath
Udevadm info starts with the device specified by the devpath and then
    ATTRS{devpath}=="1.4.2"
    ATTRS{devpath}=="1.4"
    ATTRS{devpath}=="1"
    ATTRS{devpath}=="0"
```

- Then, we modify (/etc/udev/rules.d/rplidar.rules) this file, bind the port number of the voice board, and enter the terminal

```
sudo gedit /etc/udev/rules.d/myspeech.rules
```

Add content as shown below

```
KERNEL=="ttyUSB*",ATTRS{devpath}=="1.4.2",ATTRS{idVendor}=="1a86",ATTRS{idProduct
}=="7523",MODE:="0777",SYMLINK+="myspeech"
```

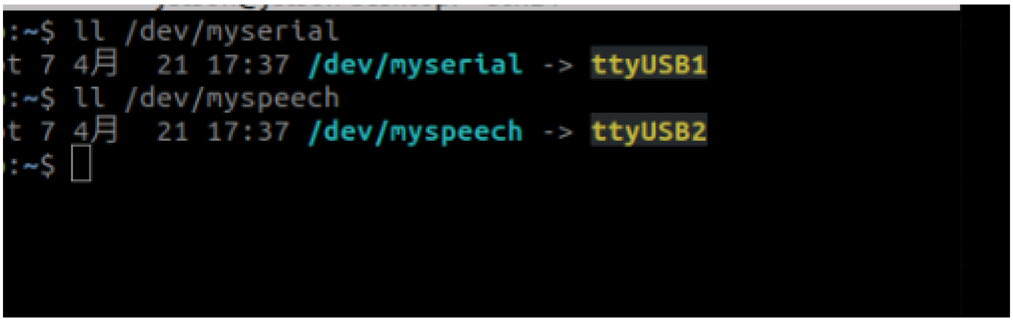
- Save and exit. Run the following three commands to reload the device

```
sudo udevadm trigger
sudo service udev reload
sudo service udev restart
```

2.3. Test

Attention their wiring position. This is fixed after binding, do not change the position at will, otherwise the system will not recognize the device.

```
ll /dev/myserial
ll /dev/myspeech
```



```
:~$ ll /dev/myserial
t 7 4月  21 17:37 /dev/myserial -> ttyUSB1
:~$ ll /dev/myspeech
t 7 4月  21 17:37 /dev/myspeech -> ttyUSB2
:~$ 
```

As shown in the figure above, if /dev/myserial and /dev/myspeech can be identified and the ttyUSB* port numbers behind them are inconsistent, the binding is successful

Attention: The ROS expansion board and voice board cannot be plugged into other ports after binding, otherwise the device number will not be recognized. If both are connected to the HUB expansion board, the HUB board cannot be plugged into other mainboard ports

2.4. Notes

After the voice board is bound to the container, you need to mount /dev/myspeech to identify the voice board in docker, Mount mode reference [07、Docker-05、Enter the robot's Docker container]