

配置CAN手册

先配置can环境

```
1 配置can
2 sudo apt install can-utils
3 sudo apt install net-tools
```

打开 ARX_CAN 文件夹：

```
1 |— ARX_CAN      #设置CAN（全局适用）
2 |   |— arx_can
3 |   |— arx_can.rules
4 |   |— can.sh
5 |   |— search.sh
6 |   |— set.sh
7 |— py          #python SDK
8 |— ROS         #ros1 SDK
9 |— ROS2        #ros2 SDK
10
```

文件	作用
arx_can	查询can
arx_can.rules	can规则
search.sh	搜索can设备
set.sh	设置规则生效
can.sh	启动can设备

这些文件是配置can和打开can的，用来让skd通过can与机器人通信。

整个过程分4步

- 1、执行search.sh
- 2、修改arx_can.rules
- 3、执行set.sh
- 4、执行can.sh

在开启can之前，需要确认自己机械臂的使用场景：

- 1、使用单个机械臂
- 2、使用四台机械臂

情况一

在单臂控制的背景下，只需要使用 “arxcan0”

在这种情况下使用的sdk包有：ARX_xx_python、ARX_xx_ros、ARX_xx_ros2。

1、执行search.sh

运行：

```
1 ./search.sh
2 //若无法运行则执行: sh search.sh
3 //后面的脚本都可以这样做
```

来搜索can设备（黑色can板）的ID，注意要把usb插入电脑，才能搜索到，然后终端会显示（不同的can设备，显示的ID也不同，下图的ID号，只是一个例子）

```
ATTRS{serial}=="209738784D4D"
ATTRS{serial}=="0000:00:14.0"
```

2、修改arx_can.rules

将 “209738784D4D” ,修改到 “arx_can.rules” 文件中，对应的 “arxcan0” ，保存即可，如下图所示：

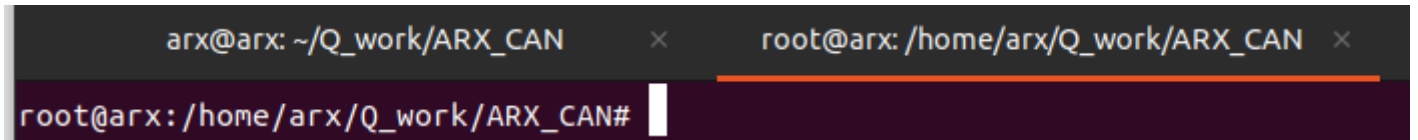
```
1 SUBSYSTEM=="tty", ATTRS{idVendor}=="16d0", ATTRS{idProduct}=="117e", ATTRS{serial}=="209738784D4D", SYMLINK+="arxcan0"
2 SUBSYSTEM=="tty", ATTRS{idVendor}=="16d0", ATTRS{idProduct}=="117e", ATTRS{serial}=="2068327F5052", SYMLINK+="arxcan1"
3 SUBSYSTEM=="tty", ATTRS{idVendor}=="16d0", ATTRS{idProduct}=="117e", ATTRS{serial}=="207D32895052", SYMLINK+="arxcan2"
4 SUBSYSTEM=="tty", ATTRS{idVendor}=="16d0", ATTRS{idProduct}=="117e", ATTRS{serial}=="2068327F5052", SYMLINK+="arxcan3"
```

3、执行set.sh

运行：

```
1 ./set.sh
```

出现类似窗口：



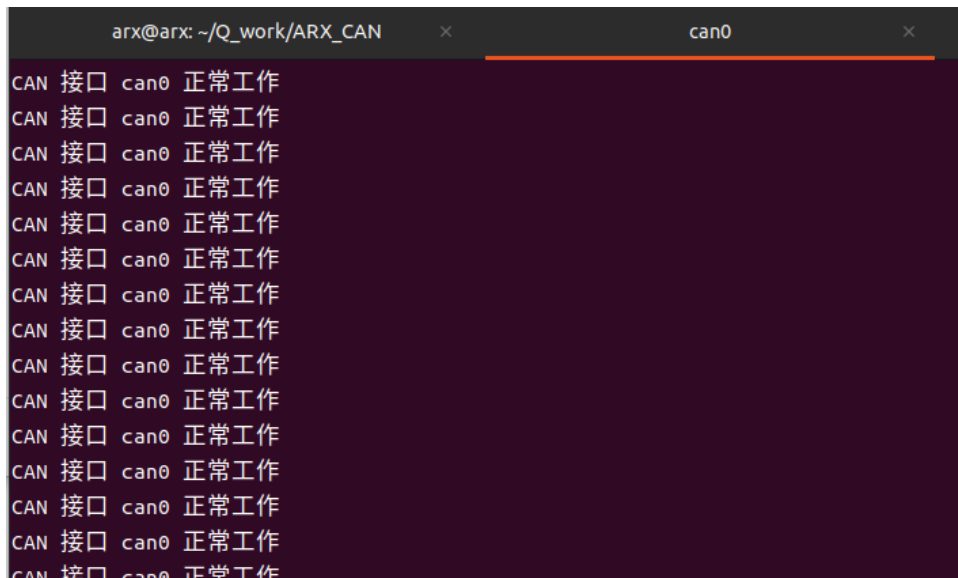
```
arx@arx: ~/Q_work/ARX_CAN x root@arx: /home/arx/Q_work/ARX_CAN x
root@arx: /home/arx/Q_work/ARX_CAN#
```

只要没报错就行。

以上操作仅在第一次运行程序前进行，只要can设备不更改，以后及无需再次配置。

4、执行can.sh

```
1 ./can.sh
```



```
arx@arx: ~/Q_work/ARX_CAN x can0 x
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
CAN 接口 can0 正常工作
```

如果关掉上述窗口，can口不会关闭。

如果不关，这些脚本会有自动重连功能，可以自己拔掉usb再插上试试。

情况二

在使用四台臂（两主臂，两从臂）进行数据采集时，需要用到“arx_can.rules”文件中所有的can。

在这种情况下使用的sdk有：ARX_xx_Remote_SDK_ROS、ARX_xx_Remote_SDK_ROS2。

1、执行search.sh

运行：

```
1 ./search.sh
2 //若无法运行则执行：sh search.sh
3 //后面的脚本都可以这样做
```

来搜索can设备（黑色can板）的ID，注意要把usb插入电脑，才能搜索到，且一次只能插入一个usb，然后终端会显示（不同的can设备，显示的ID也不同，下图的ID号，只是一个例子）

```
ATTRS{serial}=="209738784D4D"
ATTRS{serial}=="0000:00:14.0"
```

2、修改arx_can.rules

将“209738784D4D”,修改到“arx_can.rules”文件中，对应的“arxcan0”，即左手主臂，保存即可，如下图所示：

```
1 SUBSYSTEM=="tty", ATTRS{idVendor}=="16d0", ATTRS{idProduct}=="117e", ATTRS{serial}=="209738784D4D", SYMLINK+="arxcan0"
2 SUBSYSTEM=="tty", ATTRS{idVendor}=="16d0", ATTRS{idProduct}=="117e", ATTRS{serial}=="2068327F5052", SYMLINK+="arxcan1"
3 SUBSYSTEM=="tty", ATTRS{idVendor}=="16d0", ATTRS{idProduct}=="117e", ATTRS{serial}=="207D32895052", SYMLINK+="arxcan2"
4 SUBSYSTEM=="tty", ATTRS{idVendor}=="16d0", ATTRS{idProduct}=="117e", ATTRS{serial}=="2068327F5052", SYMLINK+="arxcan3"
```

与情况一不同的是，1 2步需要重复四次

机器编号	CAN ID	备注
master1	0 (arxcan0)	左手主臂（末端示教器）
follow1	1 (arxcan1)	左手从臂（末端夹爪）
master2	2 (arxcan2)	右手主臂（末端示教器）
follow2	3 (arxcan3)	右手从臂（末端夹爪）

注意，与arxcan1和arxcan3对应的机械臂必须是带有夹爪的机械臂(从臂)。

与arxcan0和arxcan2对应的机械臂必须是带有示教器的机械臂（主臂）。

3、执行set.sh

运行：

```
1 ./set.sh
```

出现类似窗口：

```
arx@arx: ~/Q_work/ARX_CAN x root@arx: /home/arx/Q_work/ARX_CAN x  
root@arx: /home/arx/Q_work/ARX_CAN#
```

只要没报错就行。

以上操作仅在第一次运行程序前进行，只要can设备不更改，以后及无需再次配置。

4、执行can.sh

打开can.sh文件：

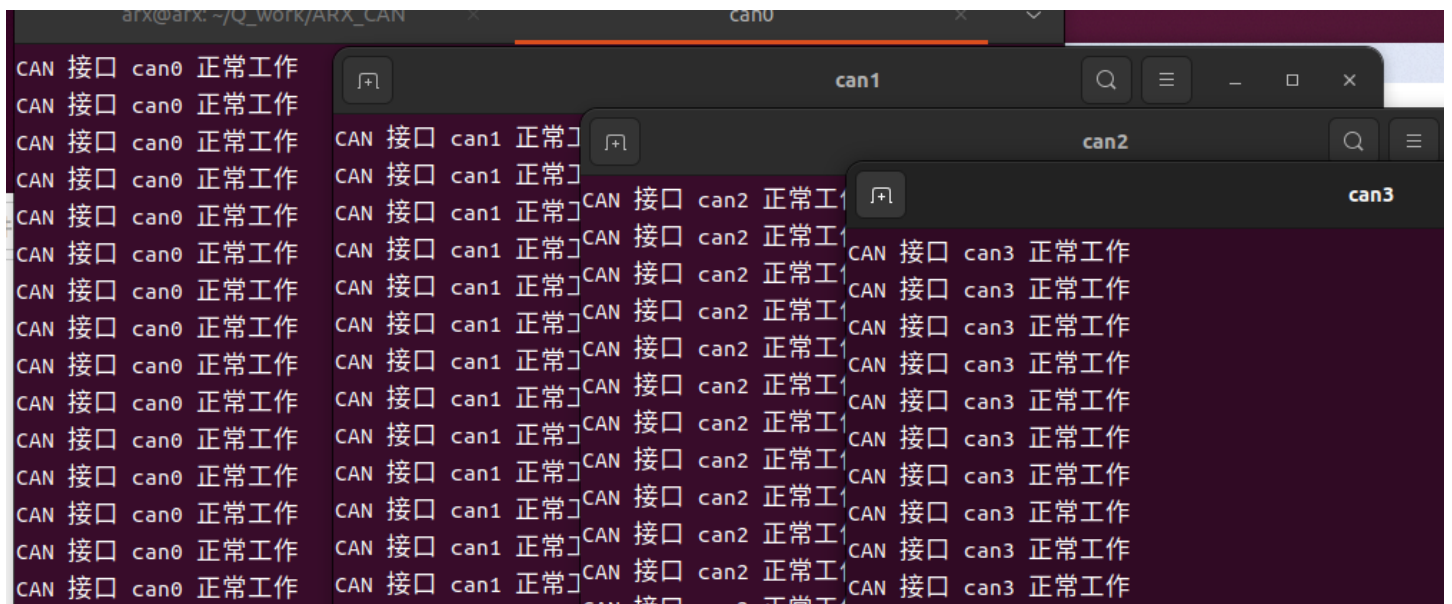
```
1 #!/bin/bash  
2  
3 workspace=$(pwd)  
4  
5 source ~/.bashrc  
6  
7  
8 gnome-terminal -t "can0" -x sudo bash -c "cd ${workspace}/arx_can; ./arx_can0.sh; exec bash;"  
9 #sleep 0.1  
10 #gnome-terminal -t "can1" -x sudo bash -c "cd ${workspace}/arx_can; ./arx_can1.sh; exec bash;"  
11 #sleep 0.1  
12 #gnome-terminal -t "can2" -x sudo bash -c "cd ${workspace}/arx_can; ./arx_can2.sh; exec bash;"  
13 #sleep 0.1  
14 #gnome-terminal -t "can3" -x sudo bash -c "cd ${workspace}/arx_can; ./arx_can3.sh; exec bash;"  
15
```

取消can.sh中的注释（如果进行单臂的控制必须要注释掉另外三个can。）。

```
1 #!/bin/bash  
2  
3 workspace=$(pwd)  
4  
5 source ~/.bashrc  
6  
7  
8 gnome-terminal -t "can0" -x sudo bash -c "cd ${workspace}/arx_can; ./arx_can0.sh; exec bash;"  
9 sleep 0.1  
10 gnome-terminal -t "can1" -x sudo bash -c "cd ${workspace}/arx_can; ./arx_can1.sh; exec bash;"  
11 sleep 0.1  
12 gnome-terminal -t "can2" -x sudo bash -c "cd ${workspace}/arx_can; ./arx_can2.sh; exec bash;"  
13 sleep 0.1  
14 gnome-terminal -t "can3" -x sudo bash -c "cd ${workspace}/arx_can; ./arx_can3.sh; exec bash;"  
15
```

保存上述文件后，运行

```
1 ./can.sh
```



如果关掉上述窗口，can口不会关闭。

如果不关，这些脚本会有自动重连功能，可以自己拔掉usb再插上试试。

总结：

控制单个臂就开启一个can（can0），控制多个臂就开启多个can，但是要注意can的id和设备的对应关系。

如果can始终打开的不正常，可以拔插usb，或者重新机械臂，再试试。或者检测有哪些线松了。

can的设置是全局的，在整个电脑里都会起作用。