A. Access under Linux to the interface to which the Anytone is connected.

Windows and Linux differ significantly in the way they handle interfaces. Under Linux, everything is mapped as a file and integrated into the file system. The interfaces under Linux can be found in the file system under**/dev/**The hardware interfaces therefore have all the properties of files: There is an owner and the access rights that must be taken into account when using Windows programs under Wine. Although the serial interfaces under Wine are "passed through" to Windows programs via a link, they also take with them all the attributes (including the access rights) that they were given when they were created (e.g. by plugging in the USB cable). The access rights often cause problems when using them under Windows.

If you now connect the Anytone via the USB cable and switch the Anytone on, the**/dev/**the appropriate device file is created. In my case (Lubuntu 24.04.1 LTS) the file **/dev/ttyACM3**The name after tty depends on the interface module and whether other interfaces of this type are already installed. I already have a DXP from microHAM connected to the computer via USB, which means that the interfaces /dev/ttyACM0 to 2 have already been created. The Anytone therefore gets the next number, namely tty/dev/ACM3.

On most Linux systems, the files for serial interfaces belong to the user**root**. They also belong to the user group**dialout**. Both the owner (root) and users who are members of the dialout group have access rights to**rw**set. Root and dialout members can therefore read (rw = read write) characters from the interface and also write characters to the interface. The situation is different for all other users (other): Here the access rights are only set to - (neither r nor w), so no access to the interface. However, since Windows programs under Wine do not run under the root user, nor is the Wine user a member of the dialout group, the Windows program acknowledges the attempt to open the interface as an error.

Because many other programs can run under Wine, I don't think it's advisable to change the default users of Wine. Alternatively, you can change the access rights of the interface file so that it is accessible to all users. The command for this is:

sudo chmod o+rw /dev/ttyACM3(enter your file here).

However, this command must be given again each time the Anytone is connected to the computer and switched on.

However, by changing the way the operating system handles plugging in the USB cable, you can have the access rights changed automatically as soon as you plug it in. To do this, you take advantage of the fact that the udev daemon, which is responsible for creating the device files, gets its instructions on how to handle certain hardware from configuration files. As is usual with Linux, the configuration files are text files, so they can be edited using the operating system's standard text editor.

As we all know, there are many ways to get to Rome, so there are several options here too. I will show you two options here:

1. an easy-to-implement solution that simply changes the access rights to all devices connected via USB and
2. a slightly more complicated one that only changes the Anytone interface and offers other options such as a unique name for the interface and other user and/or group affiliations for the interface.

The configuration files for the**udev-daemon**are located in the directory /etc/udev/rules.d . To avoid messing up the system and not being able to fix it properly later, simply create a new file:

sudo nano /etc/udev/rules.d/80-myusb.rules

The name of the file does not matter, the only important thing is that the file extension is .rules (the udev daemon only reads files with the extension .rules).

1. Simple solution

The following line is now written into this file:

KERNEL=="ttyACM[0-9]*",MODE="0666"

After saving and turning the devices off and on again, the udev daemon will now make all /dev/ttyACM devices writable for all users; the access rights are now set to rw for all users.

2. More complex solution

In contrast to solution 1, where the access rights are set for all serial devices connected via USB, solution 2 applies to exactly the one device (Anytone) that we connect; there is no change for all other USB devices. However, it is essential that the udev daemon clearly distinguishes the plugged-in device from all the others. Normally, a newly plugged-in device gets the next free file name for the device category in question. In my case, /dev/ttyACM0 or /dev/ttyACM3, depending on which device is plugged in and switched on first. This is where Linux differs from Windows: Windows always assigns the same COM interface to every device it has recognized, while under Linux the same device can get a different device file name depending on when it was plugged in.

To make the udev daemon only perform actions specific to a device when it is plugged in, you can use the process that runs every time a device is plugged in (plug and play under Linux). After plugging in, the plugged-in device transmits numerous properties such as manufacturer, product number, type of device, etc. The udev daemon then matches this information line by line with the instructions in the .rules files. Each line in a .rules file contains conditions (one or more) and instructions (one or more). If all conditions in a line are met, the instruction is executed.

Now you "only" have to find the information from the data stream that is transmitted during the plugging process that is unique to the individual device. If you have found information that applies to several devices (e.g. the manufacturer), all devices with this manufacturer ID are treated according to the instructions, which is usually not desired. In my .rules files I use the manufacturer ID, the product ID and the serial number. Note: This is not enough for devices that have several serial interfaces, such as many transceivers; here you also have to select using the interface number. However, this does not apply to our case of the Anytone with only one interface.

First we need to know which interface file was assigned to the device when it was plugged in. Again, there are two possibilities:

Either

Display all tty devices when the device is not yet switched on: **ls**

-al /dev/tty*

Then switch on the device, wait until the Anytone has booted completely and look for the tty devices a second time (ls -al /dev/tty*). Both listings differ in one file name, in my case it is /dev/ttyACM3.

Or

With the device still turned off, enter the following command:

sudo dmesg -W

The screen output waits and does not return with a new prompt. Now switch on Anytone and wait for the boot process to complete. Several lines appear on the screen. One of the last lines contains the name of the device file, always starting with tty. In my case, it is ttyACM3. Release the screen again by entering Ctrl C.

In both cases, leave the device switched on. Now that you know the correct file name of the device, you can continue.

sudo udevadm info --attribute-walk /dev/ttyACM3 > anytone.txt

The file name to which the output is redirected (anytone.txt) does not matter, just remember. Instead of ttyACM3, enter the file name that was assigned to the Anytone in your case (see above).

This file now stores all the information that the udev daemon receives. We now look to see which product ID etc. was transmitted:

grep idProduct anytone.txt (Note upper and lower case)

We get several lines; the first one is interesting, in my case

ATTRS{idProduct}=="018a"

We continue in the same way with idVendor (manufacturer) and serial (serial number). We always take the first line that grep throws out.

Now we can fill our .rules file with the condition. So create a new file and enter the information obtained through grep into the file:

sudo nano /etc/udev/rules.d/80-anytone.rules

For my Anytone, the following conditions must be entered in a single line:

 # SUBSYSTEM=="tty", ATTRS{idProduct}=="018a", ATTRS{idVendor}=="28e9", ATTRS{serial}=="000000010000"

Everything in one line, broken here only for better readability. We also start the line with a hashtag so as not to activate the still incomplete rule. In addition, at the beginning **SUBSYSTEM=="tty"**Please do not use the keyword SUBSYSTEMS (with an S at the end), which also appears in the file anytone.txt. The individual conditions are separated from each other by commas and spaces. Instead of my values (those in quotation marks), enter your values, exactly as the previous grep query returned them, e.g. including the many leading zeros in the serial number.

Now we have to write the statement that the udev daemon should execute if all conditions are met. Multiple statements are again separated by commas and spaces.

The following instructions are interesting for our case; you don't have to include them all. You just decide which instruction(s) you want.

MODE="0666"	sets the access rights to rw for all users
OWNER="egon"	assigns the interface to the user egon
GROUP="hunde"	assigns the interface to the group hunde

Note: Here you have to enter the user and group under which Wine works. This is usually also the user and group under which the user has logged in.

SYMLINK+="ttyAnytone"	creates a device file with a unique name via link
ENV{ID_MM_DEVICE_IGNORE}="1"	switches off modem initialization via AT commands. It wasn't necessary for me because the Anytone wasn't classified as a modem. But my DXO from microHAM was.

Insert the desired instructions in the same line after the conditions, delete the hashtag character at the beginning of the line after completing the line, save the file and switch Anytone off and on again. From now on, the Anytone should be connected to Linux as desired. Please check with ls -al /dev/tty*.

My line for the Anytone looks like this (line breaks here only for better readability):

SUBSYSTEM=="tty", ATTRS{idProduct}=="018a", ATTRS{idVendor}=="28e9", ATTRS{serial}=="000000010000", OWNER="xxx", GROUP="xxx", SYMLINK+ ="ttyAnytone"

B. Passing the Linux interface to Wine

Wine maps the Linux interfaces via link as a com device under the home directory of the logged in user in the directory dosdevices, subdirectory of .wine; in Linux notation, therefore, under**~/.wine/dosdevices**To know which COM port needs to be addressed with the Windows program under Wine, simply display the port directory:

ls -al ~/.wine/dosdevices/com*

All com files are displayed with their link to the actual Linux devices. Since we previously found out which interface the Anytone is connected to (in my case to /dev/ttyACM3 or / dev/ttyAnytone), we only need to look at which com file the correct interface is linked to. In my case, this is**com43**If Wine has not yet set the link (the first time you need to boot Wine), reboot Wine after closing all Windows programs:

wine boot

Now the Anytone can always be accessed under Wine using the same interface.

If you want the device to be accessible in Wine under a specific COM port, e.g. because the Windows program only offers the interfaces from COM1 to COM 20, you can do this with an entry in the registry:

Under Linux with Anytone enabled**winetricks**A window will open in which **Select standard wineprepix**activate, on**OK**A new window will open in which**start regedit** select and again**OK**Press . The Registry Editor will now start. There we go to

HKEY_LOCAL_MACHINE\Software\Wine\Ports

There we can right-click,**new**and then**string**create a new entry for a port. We create a new entry with the name com1 and enter our Linux file name for the Anytone device file as data, i.e. /dev/ttyACM3 or /dev/ttyAnytone. If we do not enter any data, this com device will be omitted from the link under Wine. I first created four empty entries from com1 to com4, then later assigned com1 to the link to /dev/Anytone, com2 to com 4 are still reserved for possible future applications that need a special com interface.

Finally, close the Registry Editor window and replace the two winetricks windows with **Cancel**also close. After booting Wine (bootwine), the interface for the Anytone is now correctly assigned under Wine.

With these settings, the Linux-Wine CPS (Wine D878UV_WineHQ_3.05) runs without any problems.