I bought a Raspberry Pi September 2012 and I was making little projects with it and also tried to implement an APRS system.

First of all I had to buy an external USB audio adapter (sound card) because the one that the RPi has didnt work with soundmodem software.

At this point my RPi was configured with ssh/vnc servers running I also configured the RPi to work with an USB Wi-Fi adapter because it's more practical to me work in that way. As the RPi has two USB ports one were used for Wi-Fi and the other for the sound card.

Installing required packages
sudo apt-get update
sudo apt-get install xastir
sudo apt-get install soundmodem
sudo apt-get install alsamixergui

Updating packages
sudo apt-get update
sudo apt-get upgrade

Configuring soundmodem
(this command will launch a GUI to configure soundmodem if you are using VNC this won't work you have to be on your RPi with a screen a mouse and a keyboard to configure soundmodem, you also can configure soundmodem in this file which is the same that soundmodemconfig GUI configure /etc/ax25/soundmodem.conf)

sudo soundmodemconfig

Note: If previous command doesn't work (example using remote connection like vncserver) try using "xhost +" command before previous command.

New configuration "APRS" (you can use other name instead of APRS)

- io: alsa
  - ➢ channel access: txdelay 150, slottime 100, p-persistence 40, full duplex ( ), txtail 10 / 0(short end)
  - ➢ plughw, 1,0
    (check what is yours with cat /proc/asound/cards)
    you also can use hw for this point but I read that plughw is better, with below command you can see your device number in my case my device is "1" and the "0" is the output for the speaker so the result is "plughw, 1,0"
    pi@raspberrypi ~ $ cat /proc/asound/cards
    0 [ALSA]: BRCM bcm2835 ALSbcm2835 ALSA - bcm2835 ALSA
    bcm2835 ALSA (RPi soundcard)

1 [AUDIO]: USB-Audio - USB AUDIO
USB AUDIO at usb-bcm2708_usb-1.3, full speed (USB sound card)

- ( ) half duplex
- mono
- PTT Driver(this part is to activate our radio PTT via serial port each time that Xastir is transmitting an APRS packet for this I had to use and USB serial adapter and then build a little interface to my Kenwood TH-G71 radio image below and also the original diagram from kenwood manual) The usb serial adapter by default is located almost always in /dev/ttyUSB0 so that will be the content of PTT driver config.

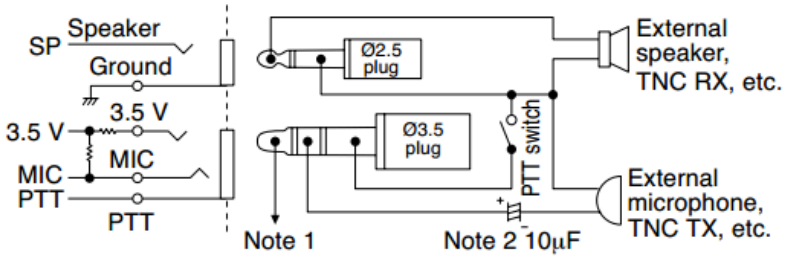
Interface for the radio

In my case I checked first my radio manual (Kenwood TH-G71) to find how to activate external transmission

then I found on Internet a simple radio interface via RTS serial port. This will work with soundmodem. So check if you have a different one.

CONNECTING OTHER EXTERNAL EQUIPMENT

When connecting an external speaker, an external microphone, or other equipment such as a TNC for packet radio to the SP jack or MIC jack, refer to the diagram below.

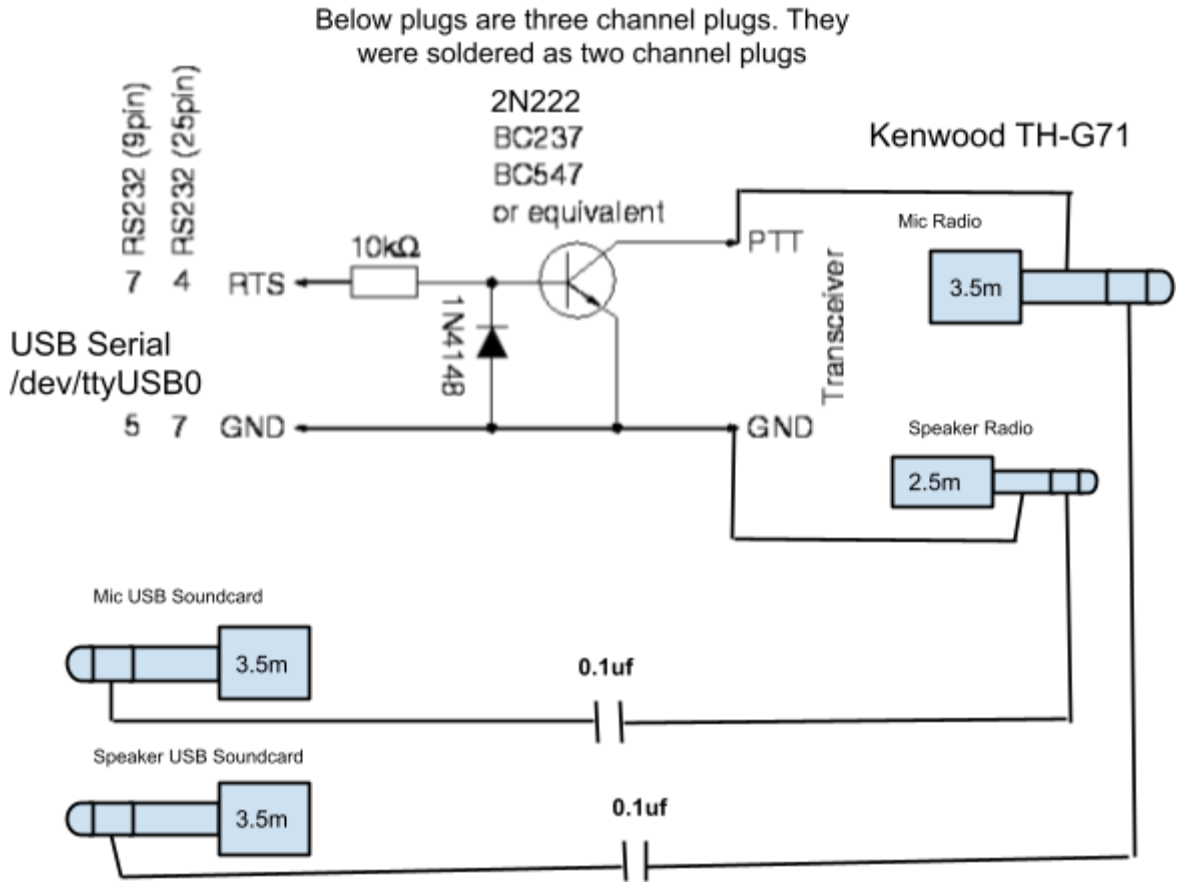


Note 1: *Voltage is developed across the 100 Ω resistor in the 3.5 V line in the transceiver. When 2 mA flows, approximately 3.3 V is developed.*

Note 2: *A 10 µF capacitor is not required in the following cases:*

- *When the other equipment has DC blocking capacitors.*
- *When a 2-terminal electret condenser microphone is used.*

So after several testings I got below diagram that works for me.
By the way I use my audio output and input as mono configuration so we only have ground and signal y each plug

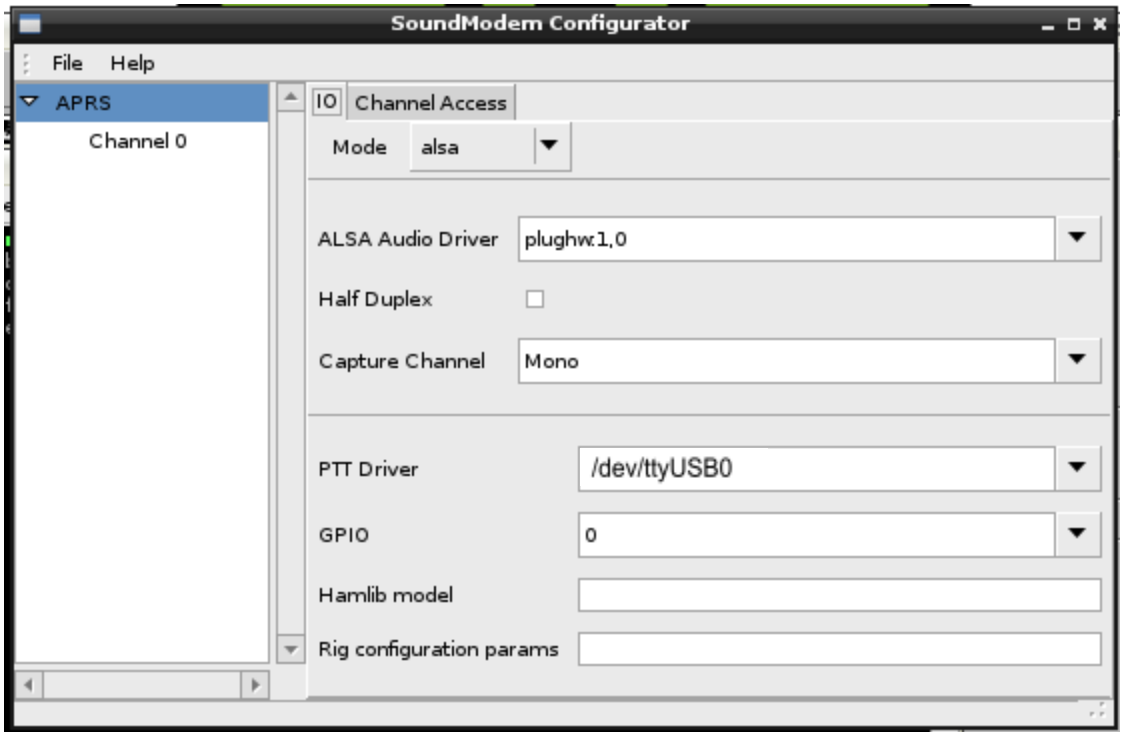


Then you have to add a new channel

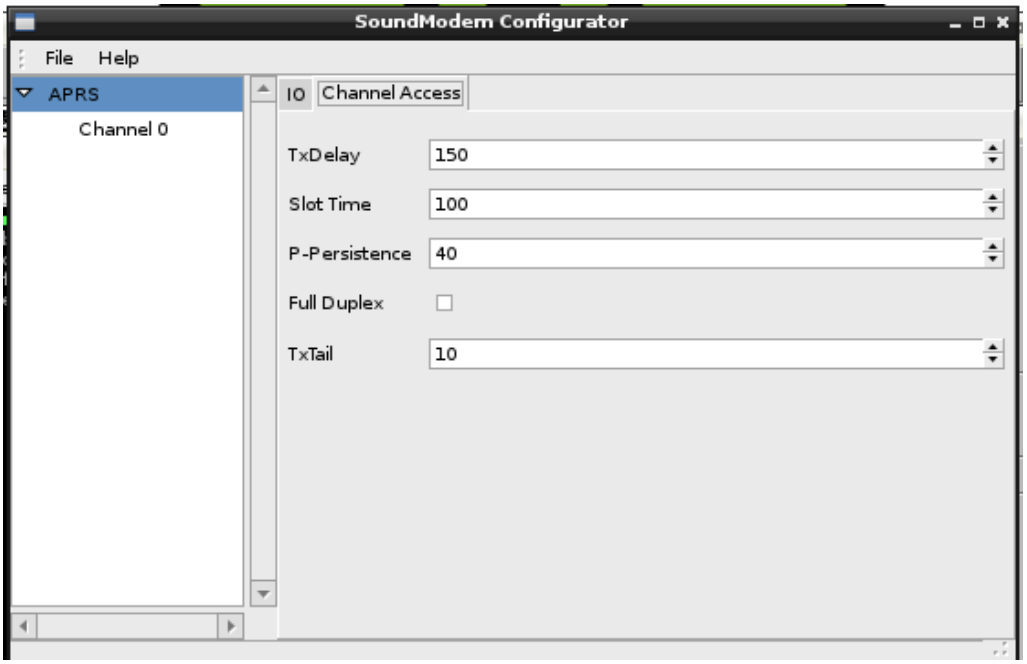
- channel 0
  - ➢ modulator
    afsk, 1200, 1200, 2200, dif.encoding
  - ➢ demodulator afsk, 1200, 1200, 2200, dif.encoding
  - ➢ packet io
    kiss, /dev/soundmodem0

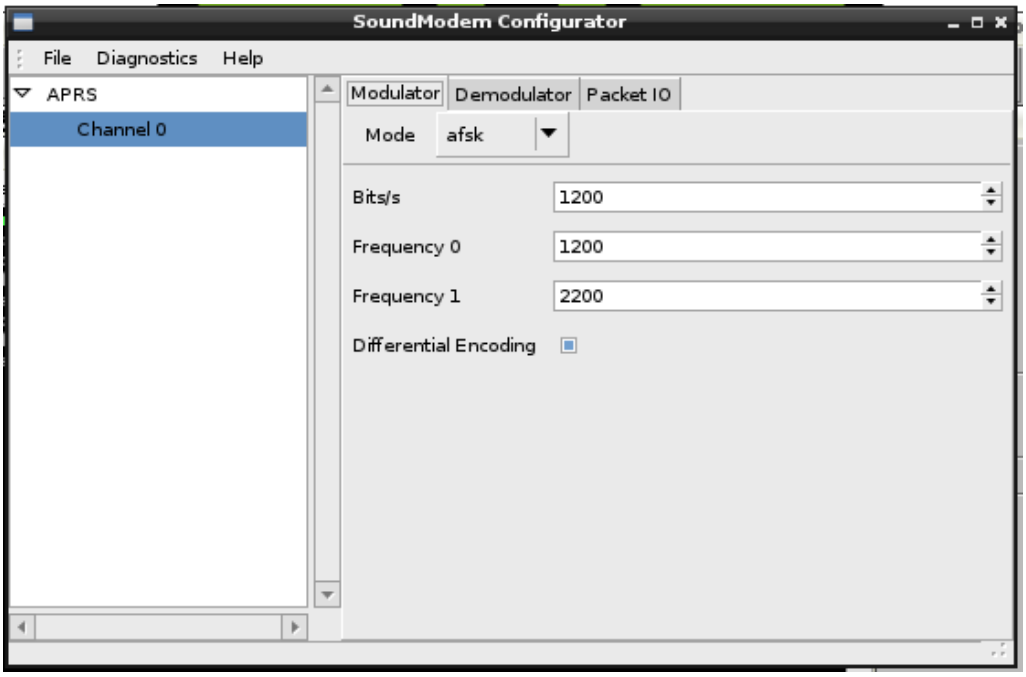
Check is config saved? relaunch config or check the config file /etc/ax25/soundmodem.conf
Check spectrum and scope on diagnostics tab are working (if you connect an audio input to your mic plug from the soundcard you will see that the spectrum windows will be registering the mic input also you can check if you already builded your radio interface that transmission is working by pressing PTT button)

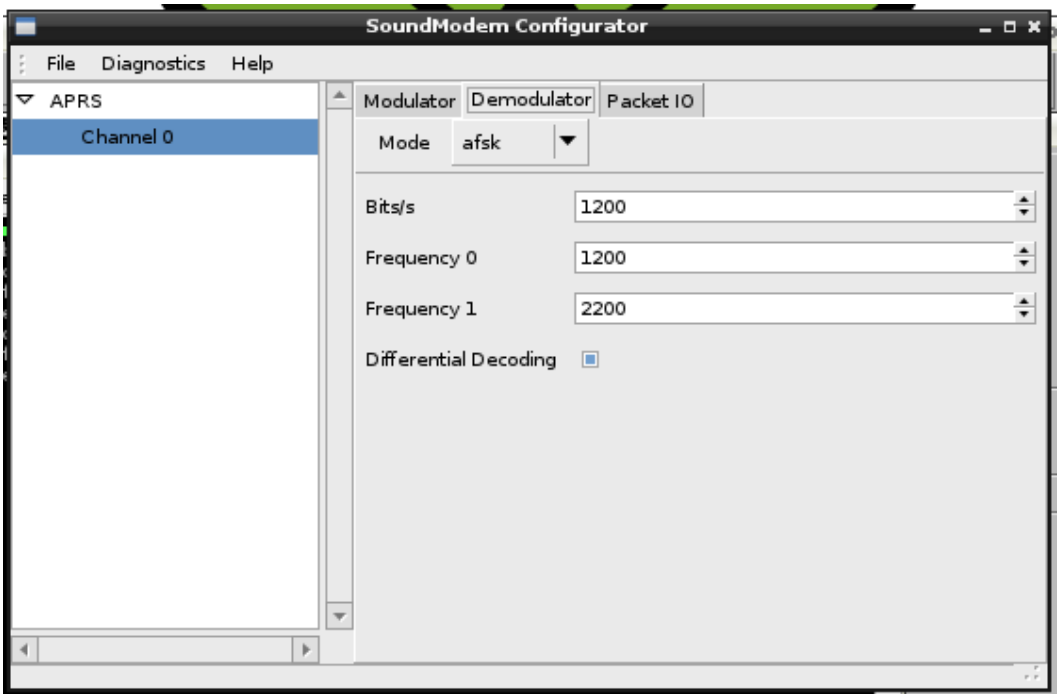
You can also check below screenshots

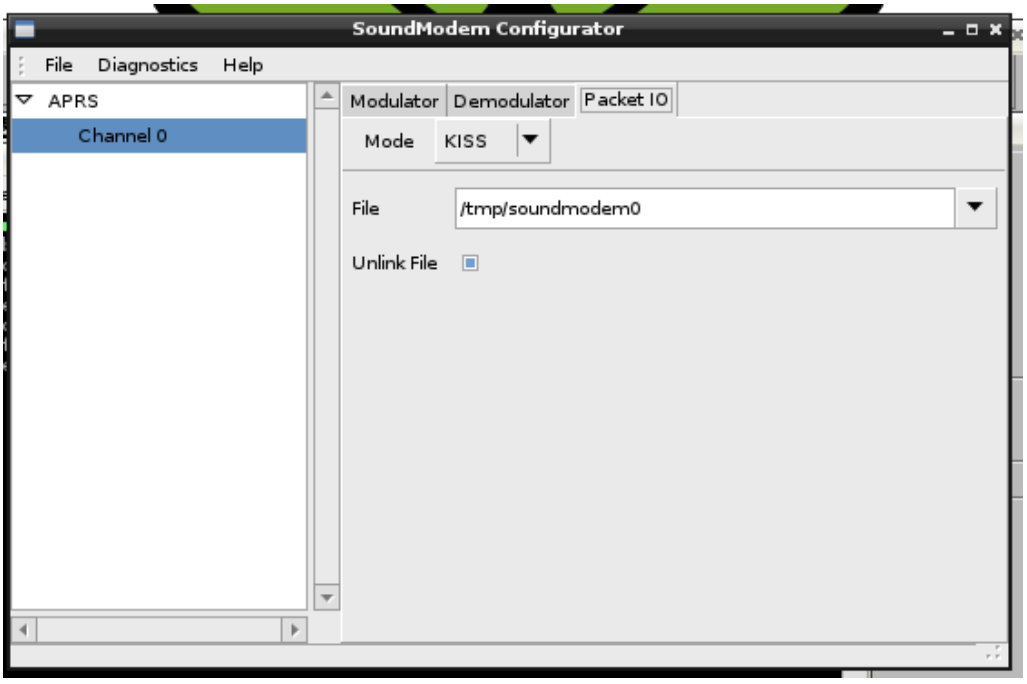


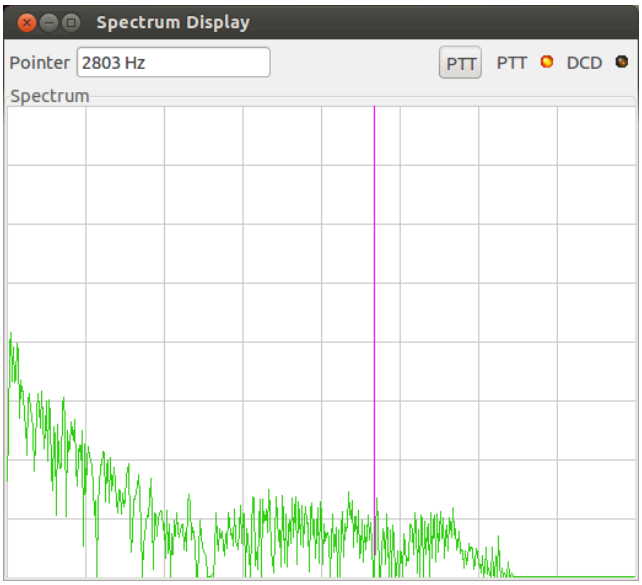
It’s a good practice set the TxDelay to 300 and TxTail to 100 so that will give enough time to soundmodem to activate the radio this will avoid packet loss.






SoundModem Configurator
File
Diagnostics
Help
APRS
Channel 0
Modulator
Demodulator
Packet IO
Mode
afsk
Bits/s
1200
Frequency 0
1200
Frequency 1
2200
Differential Decoding

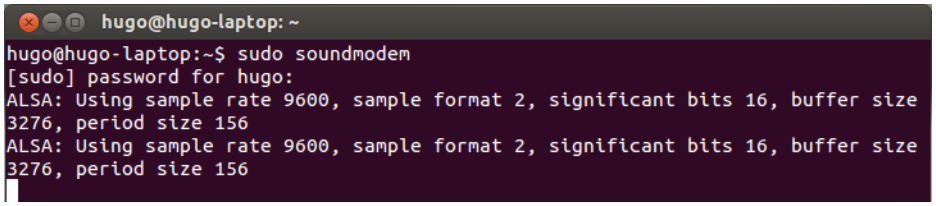

SoundModem Configurator
File
Diagnostics
Help
APRS
Channel 0
Modulator
Demodulator
Packet IO
Mode
KISS
File
/tmp/soundmodem0
Unlink File



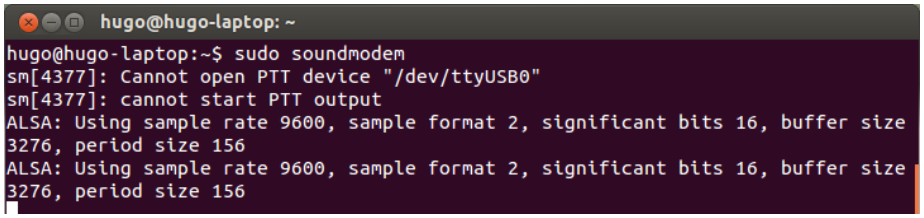
When all configurations are done you can start soundmodem but always as sudo user so it's important to start alway soundmodem before xastir because xastir will use soundmodem as a TNC. so to start it only type in a console:

sudo soundmodem

if you see something like below image its done:



If you get an error like below:



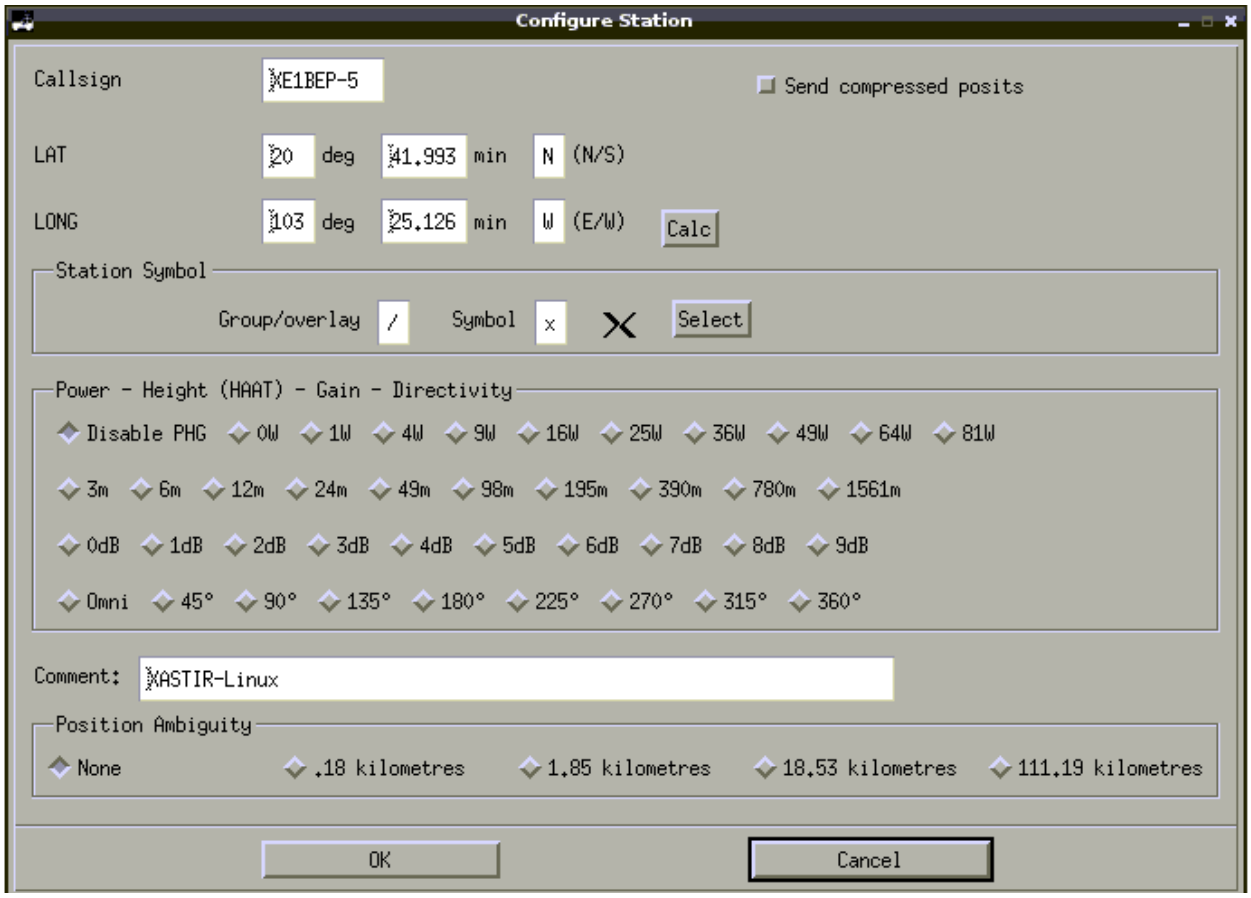
check for your serial port permissions or check if it's connected properly to your RPi (some cases this also can be due to power supply issues so chose at least 1A power supply to avoid this).

Configuring Xastir

I will suggest that it's better start xastir from console instead of our GUI environment because if something fails at least you will see the error reflected in the console. To start xastir from console just type "xastir" + enter

File → Configure→ Station

In this box I only modified the callsign, the coordinates were configured automatically with the GPS once I setup the interface for the GPS device. Also you can modify the symbol the will show on the map by default is the X for unix/linux devices (check the correct number for your station -5 it's for only internet objects -10 will work for Igates you can check this in APRS documentation)



File→ Configure→ Timing

I configured this to send a APRS packet every 30 seconds (this was only for testing)

Configuring interfaces

I already configured three interfaces on Xastir (activate from startup option will activate the interfaces once you start Xastir):

- KISS TNC interface for transmit APRS packet through the soundcard and also to activate the radio transmission .
- Internet interface to log into APRS-IS for this one you will need a password that you can create with callpass command on our linux terminal that is already included with xastir software.
- Serial GPS interface will get coordinates from any serial GPS that is connected to your RPi I used Garmin GPS with NMEA output at 4800 bps.

Permissions issues

If you get the error "Interface Error! Error opening interface 0 Hard Fail" or something like that

As root: chmod 4755 /usr/bin/xastir

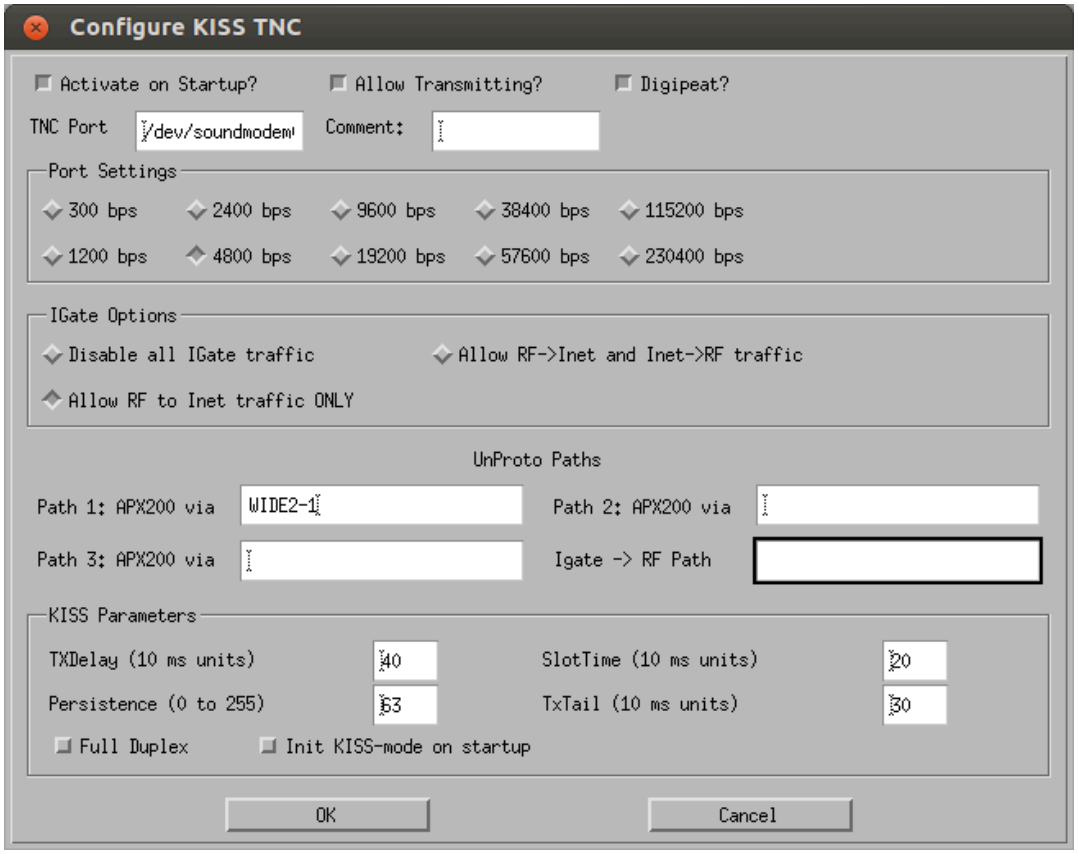
That gives Xastir SUID Root privileges when opening/closing/manipulating ports.

KISS TNC

Interface → Interface control → add → Serial KISS TNC (TNC Port: /dev/soundmodem0)

Check also digipath in this case WIDE2-1 is recommended for fixed stations.

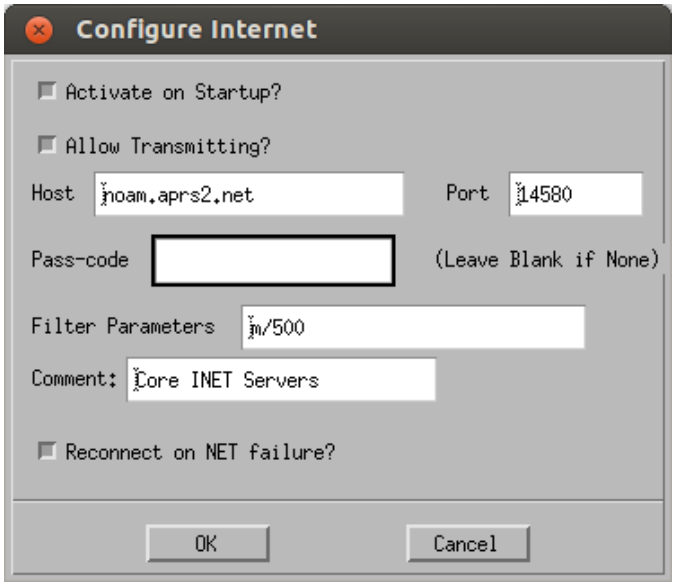
If you want your RPi only as beaconing unit Igate options: Disable all Igate Traffic



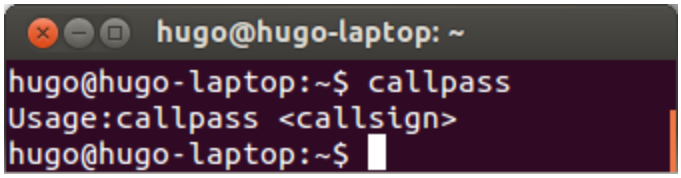
Internet Server

Interface → Interface control → add → Internet Server

Host is for north america it may change if your in another region



If you don't have pass-code you can generate your own passcode with callpass



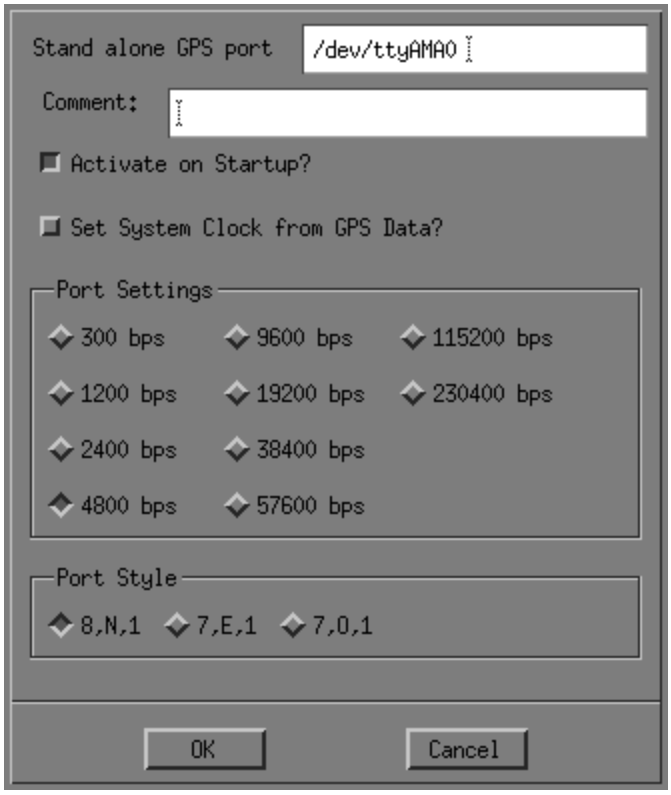
callpass xe1bep

and then enter that will generate a pass-code

Serial GPS

Interface → Interface control → add → Serial GPS

In order to safe USB ports (RPi model B only have two ports I used one for USB soundcard and the other one for USB serial to activate radio transmission) I decided to use GPIO serial port from RPi (/dev/ttyAMA0) as default this port is a console output but we can disable that and use it for our own purpose. This configuration will work with NMEA output at 4800 bps.



Using UART /dev/ttyAMA0 serial port

Step One: Edit /boot/cmdline.txt
Next, enter the following command from the command line:
Copy Code

sudo nano /boot/cmdline.txt
And change:
dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait

to:
dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait

Step Two: Edit /etc/inittab
From the command prompt enter the following command:
Copy Code

sudo nano /etc/inittab
And change:
#Spawn a getty on Raspberry Pi serial line
T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100

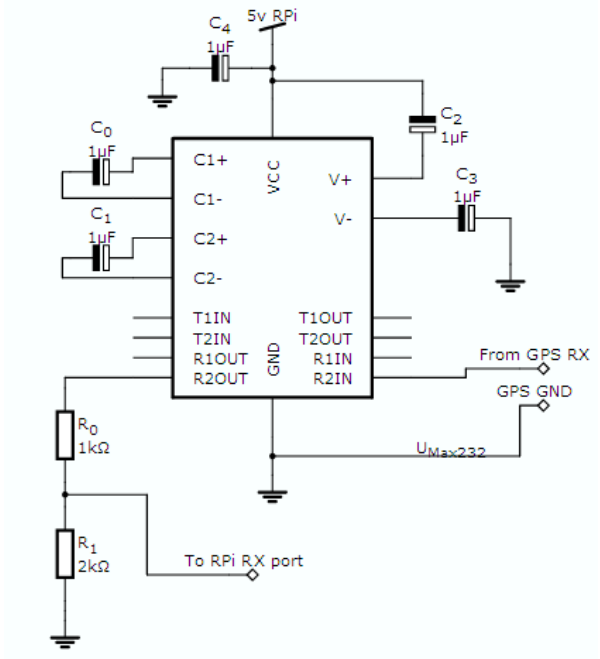
to:
#Spawn a getty on Raspberry Pi serial line
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100

Step Three: Reboot your Pi
After rebooting the Pi for the above changes to take effect.

In below page is the circuit to connect GPS to the RPi only we need to identify a couple of things, we will use GPIO 15 (RXD) to receive GPS data al also we can use 5v and Ground pins from RPI as the power source of the serial circuit included below.

GPS Device (Garmin Venture)

I used this GPS because I received it as a gift from XE2O and he told me that it would work because supports serial actually by default this GPS is sending data to its serial output once is turn on. I setup the serial output as NMEA output at 4800 bps.
(the GPS voltage output is 5.5v so to prevent any issues with the RPi I used below circuit, that will convert any input signal since 12v to 3.3v that is the input level of the RPi, if you have a GPS module that already has a 3.3v output you can omit this part).



Start desktop on boot (this is required if you want to run xastir from startup)
To auto run programs on GUI mode you have to set up the raspberry pi to start automatically the GUI, with the below command you can configure that.

sudo raspi-config

then select boot_behaviour and after that choose Desktop (GUI interface) option and restart your Raspberry Pi

Auto running terminal applications (non GUI) to automate soundmodem from startup
First ensure your program is executable by finding it in the file manager. Right click on the file and select properties. Select the permissions tab, check the 'Make the file executable' box and press OK. Of from the command line use:

sudo chmod +x /home/pi/projects/my_project.a

Or using a different tool set its chmod to 755 (rwxr-xr-x). It doesn't matter if the user is root.
Doesn't work?

> We've found that when copying a new file to the rpi using WinSCP, changing its properties to 755 and verifying all is OK that if we kill the power and power up again the executable doesn't run. However if we use sudo reboot at the command line it works as it should. It seems there is some sort of caching action going on so after doing this use sudo reboot the first time rather than cycling the power!

You can setup the auto run using a script (see here), or you can do it directly by editing the rc.local file:

sudo nano /etc/rc.local

After the initial comments (lines beginning with '#') add the following lines:

 # Auto run our application
sudo /home/pi/projects/my_project.a &

in our case we want this to launch soundmodem program so it will be like this:

# Auto run our application
sudo soundmodem &

"sudo" assumes you want your application run with root user privileges (remove if not) and the "&" says do it in the background.
Save it by pressing Ctrl+X, " Y", ENTER
Re-boot your RPi and it will run.
you can see if the process is running with the command "ps -A" that will show you all the process that are running you can grep the particular program with "ps -A | grep soundmodem"

Autostart X Application in Raspberry Pi

Assuming that you're using the official Raspbian (mine is "wheezy") image, the default Window Manager is LXDE. Therefore, to enable an X application to start automatically once X started, you would consult LXDE documentation. This is the direct link: http://wiki.lxde.org/en/Autostart.

What basically you should do is create a directory named **autostart** inside **~/.config**. Here, the ~ means the home directory (/home/pi in most of the cases) of the account that log-in directly to X. Then, in the **autostart** directory, create a ***.desktop** file and make it executable (chmod +x [your_file].desktop). The format of the ***.desktop** file as follows:

[Desktop Entry]
Type=Application
Name=your_application_name
Exec=your_application_path_here

With this we can configure xastir and vnc from startup

this examples were with my RPi Raspbian distro to auto-start VNC server and Xastir

Automating tightVNC

cd /home/pi/.config/autostart
(if the directory don’t exist you can create it “mkdir /home/pi/.config/autostart”)

nano tightvnc.desktop

and paste below lines there

[Desktop Entry]
Type=Application
Name=TightVNC
Exec=vncserver :1
StartupNotify=false

Save it by pressing Ctrl+X, " Y", ENTER

Automating Xastir

cd /home/pi/.config/autostart
(if the directory don't exist you can create it "mkdir /home/pi/.config/autostart")

nano xastir.desktop

and paste below lines there

[Desktop Entry]
Type=Application
Name=xastir
Exec=/usr/bin/xastir

Save it by pressing Ctrl+X, " Y", ENTER

Thats all there is to it. The next time you reboot the tightVNC and Xastir will start automatically.

Useful links

Raspi Config manual
Auto running programs non GUI
Auto running programs GUI (recommended)
Auto running programs GUI
Many example and tutorials
Running VNC Server at Startup
Configure soundmodem and xastir
More soundmodem configurations
Mine configured Raspbian version image
Mine Soundmodem config file
Permissions Error
Max232 connection
Voltage divider (Logic)