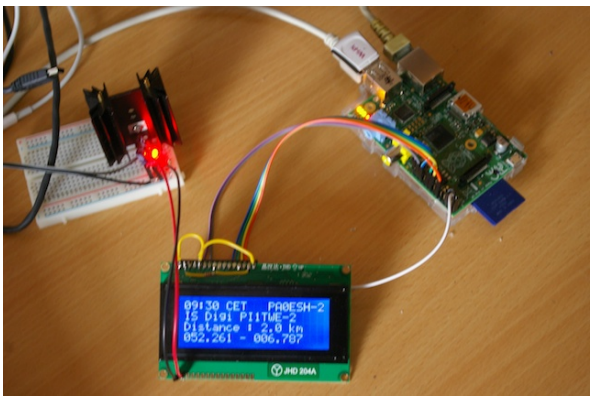
This article describes my basic activities at getting my Raspberry PI do the iGate / Digipeater work, either using a Argendata SMT KISS modem, or using a sound modem with a cheap USB sound dongle.
On December the 20th 2012, I received a nice little plastic box, containing the raspberry PI. It came together with a SD-card with the Raspbian "wheezy" operating system, and I configured myself in parallel an SD card with Arch Linux.
The next steps will be to find which software is most suited to the needs for , eventually, ending up in PI1TWE-2, the new APRS digipeater / iGate in Hengelo.
Today – april 2013, , i have the little bugger up and running, and both the normal KISS, as well as the ax25 KISS and soundmodem work.
Well, most of the time. What is left to do is to develop a circuit to do the TX from one of the GPIO pins.

Last but not least i would like to thank Phil - M0DNY for not only using his website, but also helping me out from his holiday address. See his website



https://www.thecraag.com/Main_Page

Setting up a Raspberry PI Mod B for Radio amateur – APRS use.
(work in progress – dated: 1 May 2013)

This article describes my efforts in getting a Raspberry PI mod B 512K working as an iGate / Digipeater.
I visited many sites to get information, learned a lot along the way and decided to use https://www.thecraag.com/Raspberry_Pi_APRS_Digipeater and write my own experiences as a noob on Linux along the way

The aim is come at:
A working Raspberry PI with Arch Linux as OS, an APRS iGATE / Digipeater and you will have the choice between APRX and DIXPRS, and as TNC either a sound modem using a cheap USB Sound dongle based on C-Media chip technology, or a KISS hardware TNC such as the TNC-X

Since DIXPRS requires Python 2.7 and ARCHLINUX comes with Python 3, I have decided for the time being to stick with APRX as programme for APRS on the ArchLinux distro's
However, you will find elsewere on my site info on the Raspian Wheezy distro using Python 2.7.3, hence the use of dixprs is possible.

Preparing a SD card with Arch Linux
Arch Linux ARM is based on Arch Linux, which aims for simplicity and full control to the end user. Note that this distribution may not be suitable for beginners. The latest version of this image uses the hard-float ABI, and boots to a command prompt in around ten seconds. Get it from the Raspberry site: http://www.raspberrypi.org/downloads
Default login :Username: root Password: root
Copying an image to the SD card in Mac OS X (command line) (for windows, see the Raspberry's website for the windows solution...

*· From the terminal run **df -h***
· Connect the SD card reader with the SD card inside or use the build in card reader
*· Run **df -h** again and look for the new device that wasn't listed last time. Record the device name of the file system's partition, for example, **/dev/disk1s2***
· Unmount the partition so that you will be allowed to overwrite the disk:
· **sudo diskutil unmount /dev/diskXsY** *(X maybe 1 or 2... Y the same |or: open Disk Utility and unmount the partition of the SD card (do not eject it, or you have to reconnect it)*

Using the device name of the partition work out the raw device name for the entire disk, by omitting the final "sx" and replacing "disk" with "rdisk" (**this is very important:** you **will** lose all data on the hard drive on your computer if you get the wrong device name). Make sure the device name is the name of the whole SD card as described above, not just a partition of it (for example, rdisk3, not rdisk3s1. Similarly you might have another SD drive name/number like rdisk2 or rdisk4, etc. -- recheck by using the **df -h** command both before & after you insert your SD card reader into your Mac if you have any doubts!):

For example, **/dev/disk2s1** => **/dev/rdisk2**

In the terminal write the image to the card with this command, using the raw disk device name from above (read **carefully** the above step, to be sure you use the correct rdisk# here!):

dd bs=1m if=~/path/where/imigage/is/archlinux-hf-2012-09-18.img of=/dev/rdisk2

CLONING A GOOD CARD

a. sudo dd if=/dev/disk2 of=/path/to/your/image//arch-aprx.img bs=1m
b. sudo dd bs=1m if=/path/to/your/image//arch-aprx.img of=/dev/rdisk2
If the above command report an error (dd: bs: illegal numeric value), please change bs=1M to bs=1m
After the dd command finishes, eject the card:
 # sudo diskutil eject /dev/rdisk3 *(or: open Disk Utility and eject the SD card)*
Insert it in the Raspberry Pi, and power it up !

For the rest of this article it is assumed that you are logged in as root on your Raspberry
Hence after login, change the root password to something more spectacular
The command # passwd will do it for you
Note: in case your sd card is larger than 2Gb in size, follow this article to make maximum use of the space on the card. http://www.raspberrypi-spy.co.uk/2012/06/resize-sd-card-partitions/
Upgrade of OS and Setting up of the wireless (if available)

PREPARING THE RASPBERRY PI MOD II

Connect to Raspberry PI using a wired connection, find the IP address, SSH into it and then execute:
 # pacman –Suy *(this will take some time so grab a cup of coffee or so)*
Set your locale
The default system locale is configured in /etc/locale.conf. To set the default locale, do the following:
· Make the locales available to the system by uncommenting them in /etc/locale.gen and then executing locale-gen as root.
· The locale set via localectl must be one of the uncommented locales in /etc/locale.gen.
Here is an example file:
/etc/locale.conf
 # Enable UTF-8 with Dutch settings.
LANG="en_NL.UTF-8"
 # Keep the default sort order (e.g. files starting with a '.'
 # should appear at the start of a directory listing.)
LC_COLLATE="C"
 # Set the short date to mm-dd-yyyy (test with "date +%c")
LC_TIME="en_NL.UTF-8"
If you intend to use your Raspberry on the net with WIFI, do the following, other while skip this section.

Wireless

You will need to install additional programs to be able to configure and manage wireless network profiles for netcfg.
NetworkManager and Wicd are other popular alternatives.
Install the required packages:
pacman -S wireless_tools wpa_supplicant wpa_actiond dialog iw
If your wireless adapter requires a firmware (as described in the above Establish an internet connection section and also here), install the package containing your firmware. For example:
pacman -S zd1211-firmware
After finishing the rest of this installation and rebooting, you can connect to the network with wifi-menu <interface> (where <interface> is the interface of your wireless chipset), which will generate a profile file in /etc/network.d named after the SSID. There are also templates available in /etc/network.d/examples/ for manual configuration.
wifi-menu <interface>

Warning: If you're using wifi-menu, this must be done *after* your reboot when you're no longer chrooted. The process spawned by this command will conflict with the one you have running outside of the chroot. Alternatively, you could just configure a network profile manually using the templates previously mentioned so that you don't have to worry about using wifi-menu at all.
Enable the net-auto-wireless service, which will connect to known networks and gracefully handle roaming and disconnects:
systemctl enable net-auto-wireless.service

Note: Netcfg also provides net-auto-wired, which can be used in conjunction with net-auto-wireless.
Make sure that the correct wireless interface (e.g. wlp3s0) is set in /etc/conf.d/netcfg:
# nano /etc/conf.d/netcfg
WIRELESS_INTERFACE="wlp3s0"

Reboot again…….

testing the network with :
 # iw dev wlan0 link
[root@alarmpi ~]# iw dev wlan0 link
Connected to 60:33:4b:e6:6a:25 (on wlan0)
SSID: XXXXXXXXXXXXX (your network)
freq: 2462
RX: 71084 bytes (549 packets)
TX: 2651 bytes (36 packets)
signal: -79 dBm
tx bitrate: 72.2 MBit/s MCS 7 short GI
bss flags: short-preamble short-slot-time

dtim period: 3
beacon int: 100
and check your ip adres:
 # ifconfig | less
output:
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 10.0.1.23 netmask 255.255.255.0 broadcast 10.0.1.255
inet6 fe80::20f:54ff:fe02:28da prefixlen 64 scopeid 0x20<link>
ether 00:0f:54:02:28:da txqueuelen 1000 (Ethernet)
RX packets 25 bytes 2746 (2.6 KiB)

Finish installation of remainder of mandatory packages
 # pacman –S screen (handy to have a process run in a screen)
 # pacman -S base-devel (for final compiling aprx)
 # pacman -S alsa-utils (testing your soundcard / dongle)
 # pacman -S screen base-devel alsa-utils python2
Installing ax25 library and other test tools

First Install yaourt, which is another package manager to handle packages outside Arch Linux repository….
 # pacman –S yaourt
Install pkg-config
 # yaourt –SA libax25 ax25-apps ax25-tools
For some of the packages, the PKGBUILD may need alteration. Yaourt will prompt you to edit the PKGBUILD during the install. Change the line:
arch=('i686' 'x86_64') ==> arch=('i686' 'x86_64' 'armv6h')

Installing the sound modem

(checking pkg-config is at least version 0.9.0... yes)
 # yaourt –SA soundmodem
You will have to adapt the arch again – so choose yes, and indicate which editor you favour…

Preparing the sound modules:
Make sure the USB sound dongle or other soundcard is connected.
Test by :
 # aplay –l
If the result shows you a sound module, you’re good to go.
However, you may end up with two sound-cards, the build-in one from RPI being the default, and that is not what you want for unattended aprs operation.
So here's the recipe to change the default sound-card to the USB dongle.

Set the default sound card

If your sound card order changes on boot, you can specify their order in any file ending with .conf in /etc/modprobe.d (/etc/modprobe.d/alsa-base.conf is suggested). For example, if you want your mia sound card to be #0:

 # nano /etc/modprobe.d/alsa-base.conf

options snd slots=snd_bcm2835,snd_usb_audio options snd_usb_audio index=0 options snd_bcm2835 index=2

snd_usb_audio and snd_bcm2835 are the modules used by the respective cards. This configuration assumes you have one usbsound card and the bcm2835 (onboard).

You can also provide an index of -2 to instruct ALSA to never use a card as the primary one.

Since we are using alsa, we start-up the alsamixer

 # alsamixer

Set your levels, both playback and record (pres the F5 / F6)

For more info see: http://www.alsa-project.org/main/index.php/SoundcardTesting

If necessary you can test our sound-card now by executing **# speaker-test -c 2**

It will generate pink noise from both channels.

If it works then test the recording of some (APRS) audio – hook up your speaker or radio lf_out to the mic input and execute the following…

Record a file : **# arecord -vv testsound.wav (Ctrl-C to stop)**

play back that same file : **# aplay testsound.wav**

You will hear the quality of the sound and have to determine if it is good enough to be put on the air…..

Next, create or edit the soundmodem config file in /etc/ax25. (nano /etc/ax25/soundmodem.conf)

Sample file – do change the callsign and make sure the interface is the same as in axports

<?xml version="1.0"?>
<modem>
<configuration name="AX25">
<chaccess txdelay="150" slottime="100" ppersist="40" fulldup="0" txtail="10"/>
<audio type="alsa" device="plughw:0,0" halfdup="1" capturechannelmode="Mono"/>
<ptt file="none" hamlib_model="" hamlib_params="" gpio="0"/>
<channel name="sm0"><mod mode="afsk" bps="1200" f0="1200" f1="2200" diffenc="1" inlv="8" fec="1" tunelen="32" synclen="32"/><demod mode="afsk" bps="1200" f0="1200" f1="2200" diffdec="1" inlv="8" fec="3" mintune="16" minsync="16"/><pkt mode="MKISS" ifname="sm0" hwaddr="PA0ESH-12" ip="44.94.11.8" netmask="255.255.255.0" broadcast="44.94.11.255" file="/dev/soundmodem0" unlink="1"/></channel></configuration>

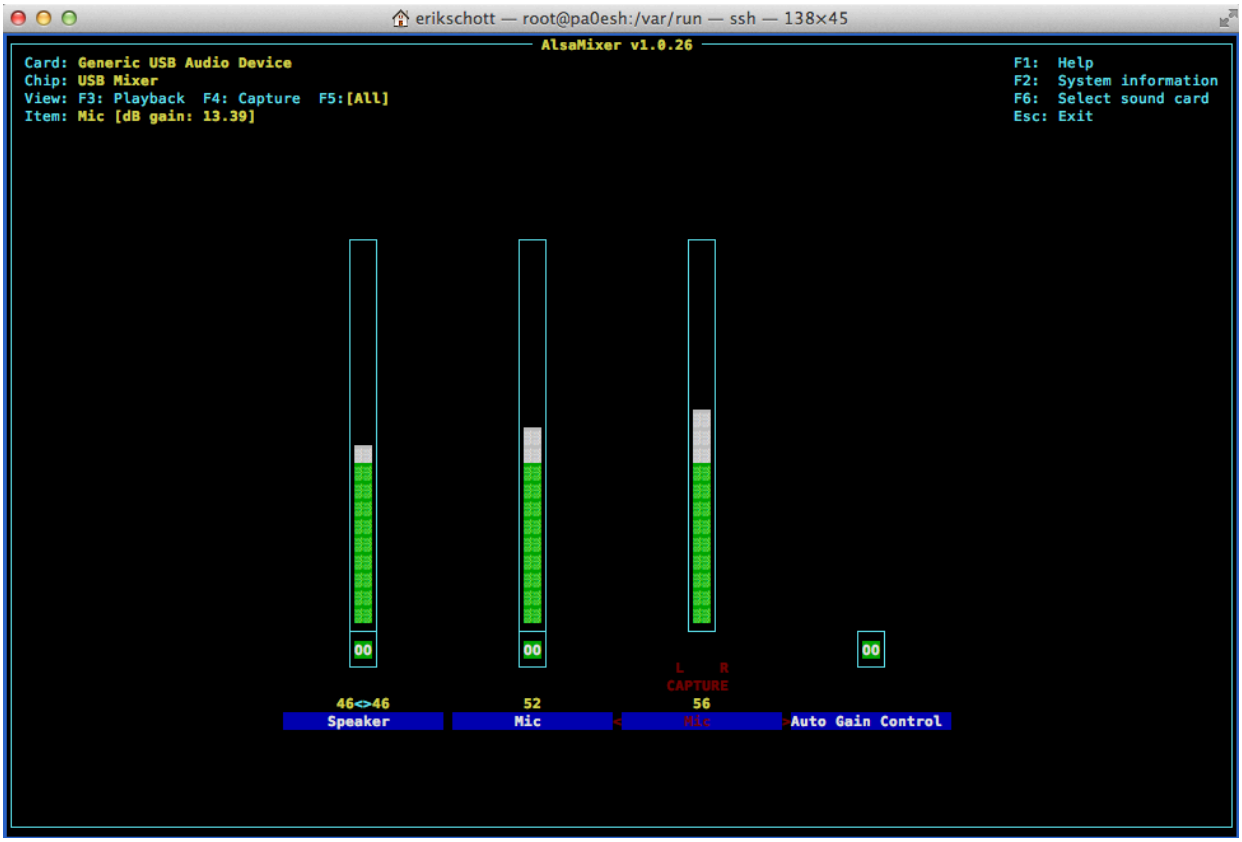
</modem>
change hwaddr="XXXX-1" into your own callsign and ssid
Then run:
soundmodem -v5 > /tmp/soundmodem.log &

(in case you are not logged in as root – then add sudo before make install.)

The order is first ax25 service using the soundmodem and only then aprs. I noticed that i needed some extra delay at boot time hence the updated doham script.
Once the soundmodem configuration is setup you can use the folowing script to bring up your AX25 system using the soundmodem driver.

```
#!/bin/sh
#BOF doham script
# start ax25 with the soundmodem driver using the port after a wait of
30-60 seconds
# defined in /etc/ax25/axports
sleep 30
/usr/sbin/soundmodem /etc/ax25/soundmodem.conf -R -
M >/dev/null 2>/dev/null&
sleep 1
/sbin/route add -host 44.94.11.8 dev sm0
# route to  the netrom IP capable nodes
/sbin/route add -
net 44.94.11.0 netmask 255.255.255.0 gw 44.94.11.8 dev sm0
sleep 1
# listen for various incoming connects
# (do not forgetto firs configure then in /etc/ax25/ax25d.conf)
/usr/sbin/ax25d
sleep 1
# listen for stations heard
/usr/sbin/mheardd
# now start aprx
/sbin/aprx
#END doham script
```

Place it anywhere in the path (I use etc/ax25/ directory), chmod 755 and run it from the prompt to test.
The red led in the sound dongle should start flashing.. in case of problems check with alsamixer that the card is not muted etc. The levels in the picture worked for me.



Once you are satisfied – write the service file for aprx called aprx.service and put it in the directory /etc/systemd/system/
chmod 755 **/etc/systemd/system/aprx.service** and then execute **# systemctl enable /etc/systemd/system/aprx.service**

Below is the file content.

```
# BOF aprx.service
[Unit]
Description=APRX Server, an iGate and Digipeater
# After=dev-ttyUSB0.device in case you work with a TNC using a usb to serial cable.
[Service]
 User=root
Type=oneshot
RemainAfterExit=yes
ExecStart=/etc/ax25/doham
[Install]
WantedBy=multi-user.target
# EOF aprx.service
```

Reboot and check that aprx is up and running…..

APRX installation

Download the latest tarball from here
http://ham.zmailer.org/oh2mqk/aprx/aprx-2.07.svn539.tar.gz
Best is to do that in your /tmp directory

$ cd /tmp
$ wget http://ham.zmailer.org/oh2mqk/aprx/aprx-2.07.svn539.tar.gz
$ tar -xvzf aprx-2.07.svn539.tar.gz
$ cd cd aprx-2.07.svn539
$ ./configure
$ make clean && make && make install

Now read the aprx manual and make your own settings in /etc/aprx.conf

You can test aprx by the command

$ aprx –ddv

see what errors appear, correct and you should be up and running.
Adjust TX delat and Tail in the soundmodem.conf file

[root@pa0esh ~]# ***systemctl status aprx.service***
Since doham has started you can also do the following from the command prompt.
mheard
listen -a -p sm0 (provided the ax25 port is called sm0)
call .....