

## Inhaltsverzeichnis

1. Packet Radio via Soundkarte unter Linux .....	72
2. Benutzer:OE2WAO .....	19
3. Benutzer:OE5HPM .....	36
4. Kategorie:Packet-Radio und I-Gate .....	53

## Packet Radio via Soundkarte unter Linux

[Versionsgeschichte interaktiv durchsuchen](#)

[Visuell Wikitext](#)

### Version vom 3. Mai 2011, 17:57 Uhr (Quelltext anzeigen)

[OE2WAO](#) ([Diskussion](#) | [Beiträge](#))

(Die Seite wurde neu angelegt: „[Kategorie: Packet-Radio und I-Gate](#) == Das Projekt == Dieser (USB) Soundkartentreiber befindet sich in der Entwicklung und soll es ermöglichen mit 2 Kanälen ...“)

### Aktuelle Version vom 22. November 2019, 18:48 Uhr (Quelltext anzeigen)

[OE5HPM](#) ([Diskussion](#) | [Beiträge](#))

([→Der Source Code](#))

(26 dazwischenliegende Versionen von 3 Benutzern werden nicht angezeigt)

Zeile 1:

```
[[Kategorie:Packet-Radio und I-Gate]]
```

Zeile 1:

```
[[Kategorie:Packet-Radio und I-Gate]]
```

+ == Das Projekt ==

+ Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.<br>

+ Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.<br>

+ Weiterer Vorteil ist die Möglichkeit des "Multibaud" Digiq, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG).

+ In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.

+

+ [[Bild:Soundmodem-box.gif|Soundmodem Schema]]

- +
- + **==Der Source Code==**
- + [\[https://github.com/oe5hpm/dxIAPRS\]](https://github.com/oe5hpm/dxIAPRS)
- + <https://github.com/oe5hpm/dxIAPRS><br>
- + [\[http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/\]](http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/)<http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/>
- +
- + **==Der kompilierte Treiber==**
- + [\[\[Media:soundmodem i386 linux. zip|Soundmodem-bin\]\]](#) - Der fertig kompilierte Soundmodem Treiber
- +
- + [\[\[Media:udpbox i386 linux bin. zip|udpbox-bin\]\]](#) - UDP Filter und RAW-Monitor Konverter und (neu) mit aprs-diqr, Bake, User-Message-Receiver
- +
- + [\[\[Media:udpgate i386 linux bin. zip|udpgate-bin\]\]](#) - I-Gate, APRS-IS, APRS-Server mit Rangefilter, HTML-Statistik, Log
- +
- + [\[\[Media:udphub i386 linux bin. zip|udphub-bin\]\]](#) - axudp Hub zum HAMNET-PR-Login ohne IP Beschränkung
- +
- + [\[\[Media:udpflex i386 linux bin. zip|udpflex-bin\]\]](#) - Interface com-port (/dev/ttySxx) mit KISS oder RMNC bidirektional auf axudp
- +
- + **==Starten bzw. Aufrufen des Treibers==**

- + mit oss testen 1200 + 9600 baud monitor (ohne kiss oder udp)
- + `./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g`
- +
- + mit alsa:
- + `aoss ./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g`
- +
- + APRS mit Xastir KISS-Interface, PTT auf ttyS0:
- + `aoss ./afskmodem -i /tmp /soundmodem -t /dev/ttyS0 -f 32000 -M 0 -i`
- + Xastir
- + `"interface" > "interface control" > "add" "serial kiss tnc"`
- + `"add" "tnc port" /tmp/soundmodem`
- + `"interface control" "start"`
- +
- + 2-Frequenz-halbduplex-Diqi mit 1200 / 1200+9600Bd xnet mit UDP und LPT PTT:<br>
- + Bei UDP ist die Startreihenfolge egal, die Programme können auch auf verschiedenen Rechnern laufen
- + `sudo nice -n -19 aoss ./afskmodem \`
- + `-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \`
- + `-C 0 -b 1 -r 300 -C 1 -b 2 \`
- + `-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 0 \`
- + `-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 0 \`

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U  
127.0.0.1:9202/9212 -m 0**

+ **linuxsnet (XNET) AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9200 d9210  
127.0.0.1**

+ **attach ip1 axudp 2 1 l9201 d9211  
127.0.0.1**

+ **attach ip2 axudp 3 1 l9202 d9212  
127.0.0.1**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+

+ **XNET mit KISS und TTY ptt (XNET  
nach dem Modem starten!)**

+ **aoss ./afskmodem \**

+ **-t /dev/ttyS0 -f 24000 -i /tmp  
/soundmodem \**

+ **-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \**

+ **-M 0 -c 1 -b 1200 -q 200 -m 0 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \**

+ **-M 2 -c 0 -b 9600 -a -g -q 200 -m 0**

+ **linuxsnet AUTOEXEC.NET**

+ **attach sdev0 kiss 1 3 38400 /tmp  
/soundmodem**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+

+ **144.800MHz 1200Bd. 70cm**  
+ **1200+9600Bd xnet, aprsdigi, aprsd**  
+ **und udpbox:**

+ **APRS hört auf allen Userzugängen**  
+ **und sendet zum IGATE. <br>**

+ **Senden auf 144.800 nur APRS**  
+ **Messages.<br>**

+ **Auf dem 1200Bd 70cm Zugang**  
+ **normales PR + APRS.<br>**

+

+ **1. Modem sendet alle Ports zu**  
+ **udpbox Port 920x und hört auf Port**  
+ **921x:<br>**

+ **(auf langsamen Rechnern oder bei**  
+ **hoher CPU last hilft Priotität mit nice**  
+ **oder renice erhöhen)**

+ **sudo nice -n -19 aoss ./afskmodem \**

+ **-p /dev/parport0 -f 44100 -c 2 -s 9 -l**  
+ **256 -b 6 -e 7 \**

+ **-C 0 -b 1 -r 300 -C 1 -b 2 \**

+ **-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:**  
+ **9200/9210 -m 2 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -U**  
+ **127.0.0.1:9201/9211 -m 2 \**

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U**  
+ **127.0.0.1:9202/9212 -m 2**

+

+ **2. XNET empfängt von udpbox und**  
+ **sendet direkt zum Modem:<br>**

+ **linuxsnet AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9300 d9210**  
+ **127.0.0.1**

+ **attach ip1 axudp 2 1 l9301 d9211**  
+ **127.0.0.1**

+ **attach ip2 axudp 3 1 l9302 d9212**  
+ **127.0.0.1**

- + **po 1 baud 1200**
- + **po 2 baud 1200**
- + **po 3 baud 9600**
- +
- + **3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)<br>**
- + **sendet alle UI zu aprsd auf 192.168.1.1:9000<br>**
- + **sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1 <br>**
- + **./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\**
- + **-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\**
- + **-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\**
- + **-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v**
- +
- + **4. udpbox als aprs-digi (Beispiel)<br>**
- + **-R empfangen axudp auf Port 9000**
- +
- + **-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt**
- +
- + **-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg**
- + **wie z.b. gar nicht aprs-frames, thirdparty-messages, Status-Meldungen**
- + **weq. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten**

- + (siehe aprs Protokollbeschreibung [APRS101.pdf](#))
- +
- + -x filtere Frames mit TCPIP oder NOCALL weg
- +
- + -p sende nur (soweit vom Absender richtig adressierte) Frames die
- + nach direkt gehört aussehen (first hop digi), sende aber den Rest
- + vom Pfad nach Protokoll modifiziert mit für weitere Hops
- + füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein
- + andernfalls bliebe der digi unsichtbar.
- + Es werden alle Adressierungsarten akzeptiert einschliesslich der
- + effizientesten mit Destination-SSID. Damit kann bei (insbesondere von
- + Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa
- + 30% eingespart werden.
- +
- + -t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg,
- + lasse aber (retryende) User Messages nach 28s durch
- +
- + -b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls
- + gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt
- + wurde, alle 30min



- +
- + **-k** Filtere alles (ausser User-msq) ausserhalb Umkreis koordinate(grad) /radius(km)
- +
- + **-c** sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt
- + **mit Linefeed an Rechner 192.168.1.24: 2000**
- +
- + **-r** sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100
- +
- + **-v** sagt was es tut und warum auf dem standard output
- +
- + **./udpbox -v -u OE0AAA-12:/tmp /msg12.txt -R 0.0.0.0:9000\**
- + **-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\**
- + **-p 5,6,7,8,9 -t 1680,28 -b 300: aprsbeacon.txt -k 48.2/-13.1/40\**
- + **-c 192.168.1.24:2000 -e -r 127.0.0.1: 9100<br>**
- +
- + **Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format**
- + **an zb. aprsd für lgate senden (-m 127.0.0.1:9304)**
- + **Weitere parameter siehe -h**
- +
- + **Bakentext File Beispiel:**
- + **(Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen**

- + aber Call und Koordinaten ausbessern)
- + Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit
- + zb. von einem Messwert Programm modifiziert werden.
- +
- + OE0AAA-11>TEST,TRACE2-2:!9000.00 N/18000.00E#PHG3750Test Digi
- +
- + dazu auf 266MHz Geode CPU mit billiq-USB-Sound"karte" optimierter Modemstart
- + (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)
- +
- + aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\
- + -l 128 -b 1 -M 0 -U 127.0.0.1:9000 /9100 -m 0 -e 50 &
- +
- + Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund
- +
- +
- + I-Gate mit udpgate
- +
- + Rx-Igate kompatibel nach:
- + [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)
- +
- + ./udpgate -R 0.0.0.0:9000 -t 14580 -s OE0AAA-10 -n 10:netbeacon.txt\

+ **-q www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501**

+

+ **-t TCP Port für Connects mit Aprs-Gattern wie xastir oder weitere igates**

+

+ **-s Call des Servers**

+

+ **-n alle 10 Min Netzbake mit Server Position**

+ **File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!)**

+ **grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol**

+

+ **!8959.00N/01300.20E&lgate Nordpol**

+

+ **-q Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)**

+

+ **-p passwort oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist**

+

+ **-f Filterparameter werden zum Server gesendet**

+

+ **-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel**

+ **(das File wird nach jeder Zeile geschlossen und kann gekürzt /gelöscht werden)**

The diagram illustrates the structure of a project page. On the left, a table of contents lists sections with their corresponding line numbers. On the right, the main content area shows the actual HTML structure of the page, including the header, navigation links, and the main text blocks.

Linie	Inhalt
1	Header
2	Navigation
3	Navigation
4	Navigation
5	Navigation
6	Navigation
7	Navigation
8	Navigation
9	Navigation
10	Navigation
11	Navigation
12	Navigation
13	Navigation
14	Navigation
15	Navigation
16	Navigation
17	Navigation
18	Navigation
19	Navigation
20	Navigation
21	Navigation
22	Navigation
23	Navigation
24	Navigation
25	Navigation
26	Navigation
27	Navigation
28	Navigation
29	Navigation
30	Navigation
31	Navigation
32	Navigation
33	Navigation
34	Navigation
35	Navigation
36	Navigation
37	Navigation
38	Navigation
39	Navigation
40	Navigation
41	Navigation
42	Navigation
43	Navigation
44	Navigation
45	Navigation
46	Navigation
47	Navigation
48	Navigation
49	Navigation
50	Navigation
51	Navigation
52	Navigation
53	Navigation
54	Navigation
55	Navigation
56	Navigation
57	Navigation
58	Navigation
59	Navigation
60	Navigation
61	Navigation
62	Navigation
63	Navigation
64	Navigation
65	Navigation
66	Navigation
67	Navigation
68	Navigation
69	Navigation
70	Navigation
71	Navigation
72	Navigation
73	Navigation
74	Navigation
75	Navigation
76	Navigation
77	Navigation
78	Navigation
79	Navigation
80	Navigation
81	Navigation
82	Navigation
83	Navigation
84	Navigation
85	Navigation
86	Navigation
87	Navigation
88	Navigation
89	Navigation
90	Navigation
91	Navigation
92	Navigation
93	Navigation
94	Navigation
95	Navigation
96	Navigation
97	Navigation
98	Navigation
99	Navigation
100	Navigation

The main content area on the right shows the HTML structure of the page. It includes a header, a navigation bar, and a main text block. The text block contains several paragraphs of text, including a section titled "Das Projekt" and a section titled "Dieser (USB) Soundkartentreiber".

**Aktuelle Version vom 22. November 2019, 18:48 Uhr**

# Inhaltsverzeichnis

1 Das Projekt ..... 84

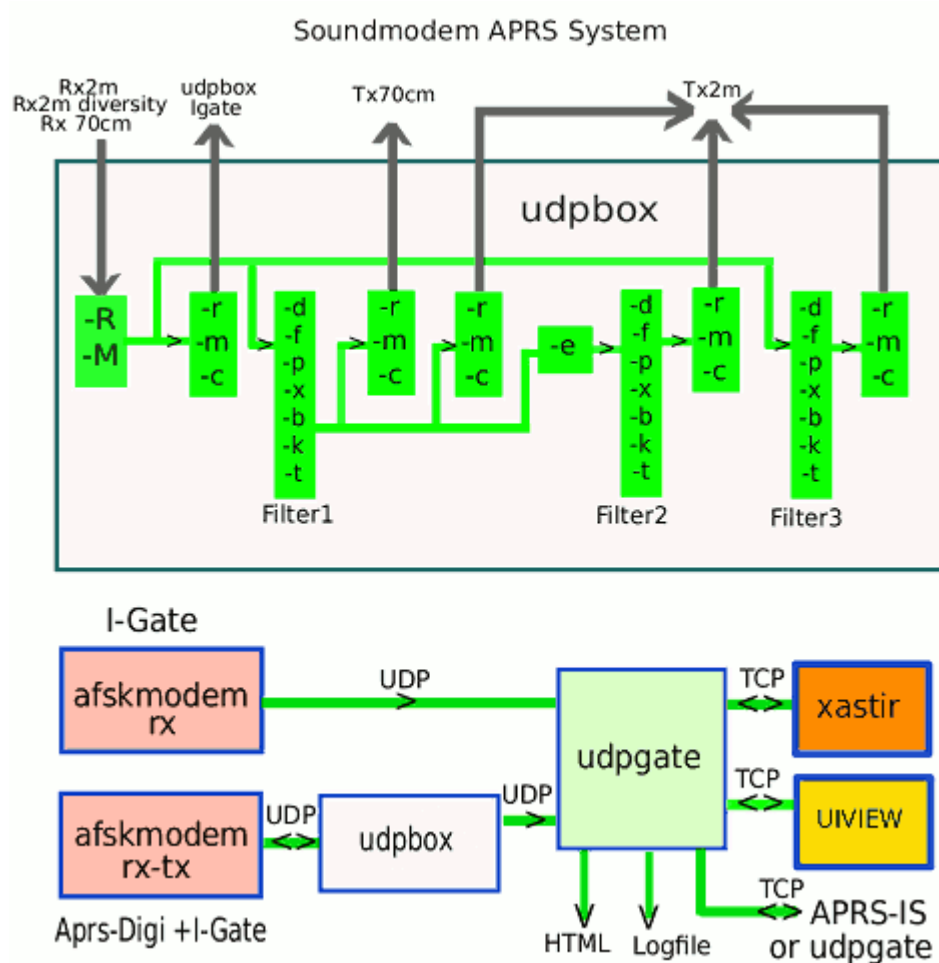
2 Der Source Code .....	84
3 Der kompilierte Treiber .....	84
4 Starten bzw. Aufrufen des Treibers .....	85

## Das Projekt

Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.

Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.

Weiterer Vorteil ist die Möglichkeit des **"Multibaud"** Digi, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG). In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.



## Der Source Code

[1]<https://github.com/oe5hpm/dxIAPRS>

[2]<http://gitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/>

## Der kompilierte Treiber

**Soundmodem-bin** - Der fertig kompilierte Soundmodem Treiber

**udpbox-bin** - UDP Filter und RAW-Monitor Konverter und (neu) mit aprs-digi, Bake, User-Message-Receiver

**udpgate-bin** - I-Gate, APRS-IS, APRS-Server mit Rangefilter, HTML-Statistik, Log

**udphub-bin** - axudp Hub zum HAMNET-PR-Login ohne IP Beschränkung

**udpflex-bin** - Interface com-port (/dev/ttySxx) mit KISS oder RMNC bidirektional auf axudp

## Starten bzw. Aufrufen des Treibers

mit oss testen 1200 + 9600 baud monitor (ohne kiss oder udp)

```
./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

mit alsa:

```
aoss ./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

APRS mit Xastir KISS-Interface, PTT auf ttyS0:

```
aoss ./afskmodem -i /tmp/soundmodem -t /dev/ttyS0 -f 32000 -M 0 -i
```

Xastir

```
"interface" > "interface control" > "add" "serial kiss tnc"  
"add" "tnc port" /tmp/soundmodem  
"interface control" "start"
```

2-Frequenz-halbduplex-Digi mit 1200 / 1200+9600Bd xnet mit UDP und LPT PTT:

Bei UDP ist die Startreihenfolge egal, die Programme können auch auf verschiedenen Rechnern laufen

```
sudo nice -n -19 aoss ./afskmodem \  
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \  
-C 0 -b 1 -r 300 -C 1 -b 2 \  
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 0 \  
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 0 \  
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 0
```

linuxsnet (XNET) AUTOEXEC.NET

```
attach ip0 axudp 1 1 l9200 d9210 127.0.0.1
attach ip1 axudp 2 1 l9201 d9211 127.0.0.1
attach ip2 axudp 3 1 l9202 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

XNET mit KISS und TTY ptt (XNET nach dem Modem starten!)

```
aoss ./afskmodem \
-t /dev/ttyS0 -f 24000 -i /tmp/soundmodem \
-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \
-M 0 -c 1 -b 1200 -q 200 -m 0 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \
-M 2 -c 0 -b 9600 -a -g -q 200 -m 0
```

linuxsnet AUTOEXEC.NET

```
attach sdev0 kiss 1 3 38400 /tmp/soundmodem
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

144.800MHz 1200Bd, 70cm 1200+9600Bd xnet, aprsdigi, aprsd und udpbox: APRS hört auf allen Userzugängen und sendet zum IGATE.

Senden auf 144.800 nur APRS Messages.

Auf dem 1200Bd 70cm Zugang normales PR + APRS.

1. Modem sendet alle Ports zu udpbox Port 920x und hört auf Port 921x:

(auf langsamen Rechnern oder bei hoher CPU last hilft Priotität mit nice oder renice erhöhen)

```
sudo nice -n -19 aoss ./afskmodem \
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \
-C 0 -b 1 -r 300 -C 1 -b 2 \
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 2 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 2 \
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 2
```

2. XNET empfängt von udpbox und sendet direkt zum Modem:

linuxsnet AUTOEXEC.NET

```
attach ip0 axudp 1 1 l9300 d9210 127.0.0.1
attach ip1 axudp 2 1 l9301 d9211 127.0.0.1
attach ip2 axudp 3 1 l9302 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```



### 3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)

sendet alle UI zu aprsd auf 192.168.1.1:9000

sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1

```
./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\  
-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\  
-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\  
-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v
```

### 4. udpbox als aprs-digi (Beispiel)

-R empfangen axudp auf Port 9000

-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt

-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg wie z.B. garnicht aprs-frames, thirdparty-messages, Status-Meldungen weg. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten (siehe aprs Protokollbeschreibung APRS101.pdf)

-x filtere Frames mit TCPIP oder NOCALL weg

-p sende nur (soweit vom Absender richtig adressierte) Frames die nach direkt gehört aussehen (first hop digi), sende aber den Rest vom Pfad nach Protokoll modifiziert mit für weitere Hops füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein anderfalls bliebe der digi unsichtbar. Es werden alle Adressierungsarten akzeptiert einschliesslich der effizientesten mit Destination-SSID. Damit kann bei (insbesondere von Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa 30% eingespart werden.

-t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg, lasse aber (retryende) User Messages nach 28s durch

-b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt wurde, alle 30min

-k Filtere alles (ausser User-msg) ausserhalb Umkreis koordinate(grad)/radius(km)

-c sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt mit Linefeed an Rechner 192.168.1.24:2000

-r sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100

-v sagt was es tut und warum auf dem standard output

```
./udpbox -v -u OE0AAA-12:/tmp/msg12.txt -R 0.0.0.0:9000\  
-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\  
-p 5,6,7,8,9 -t 1680,28 -b 300:aprsbeacon.txt -k 48.2/-13.1/40\  
-c 192.168.1.24:2000 -e -r 127.0.0.1:9100
```

Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format an zb. aprsd für lgate senden (-m 127.0.0.1:9304) Weitere parameter siehe -h

Bakentext File Beispiel: (Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen aber Call und Koordinaten ausbessern) Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit zb. von einem Messwert Programm modifiziert werden.

```
0E0AAA-11>TEST,TRACE2-2:!9000.00N/18000.00E#PHG3750Test Digi
```

dazu auf 266MHz Geode CPU mit billig-USB-Sound"karte" optimierter Modemstart (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)

```
aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\  
-l 128 -b 1 -M 0 -U 127.0.0.1:9000/9100 -m 0 -e 50 &
```

Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund

I-Gate mit udpgate

Rx-Igate kompatibel nach: [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)

```
./udpgate -R 0.0.0.0:9000 -t 14580 -s 0E0AAA-10 -n 10:netbeacon.txt\  
-g www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501
```

-t TCP Port für Connects mit Aprs-Gaffern wie xastir oder weitere igates

-s Call des Servers

-n alle 10 Min Netzbake mit Server Position File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!) grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol

```
!8959.00N/01300.20E&Igate Nordpol
```

-g Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)

-p password oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist

-f Filterparameter werden zum Server gesendet

-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel (das File wird nach jeder Zeile geschlossen und kann gekürzt/gelöscht werden)

-w www Port

www-Statistik-Beispiel

Dieses Projekt ist Open Source - Haftung, Verantwortung und Spaß übernimmt jeder selbst.

## Packet Radio via Soundkarte unter Linux: Unterschied zwischen den Versionen

[Versionsgeschichte interaktiv durchsuchen](#)

[Visuell Wikitext](#)

### Version vom 3. Mai 2011, 17:57 Uhr (Quelltext anzeigen)

[OE2WAO](#) ([Diskussion](#) | [Beiträge](#))

(Die Seite wurde neu angelegt: „[Kategorie: Packet-Radio und I-Gate](#) == Das Projekt == Dieser (USB) Soundkartentreiber befindet sich in der Entwicklung und soll es ermöglichen mit 2 Kanälen ...“)

### Aktuelle Version vom 22. November 2019, 18:48 Uhr (Quelltext anzeigen)

[OE5HPM](#) ([Diskussion](#) | [Beiträge](#))

([→Der Source Code](#))

(26 dazwischenliegende Versionen von 3 Benutzern werden nicht angezeigt)

Zeile 1:

[[Kategorie:Packet-Radio und I-Gate]]

Zeile 1:

[[Kategorie:Packet-Radio und I-Gate]]

+

**== Das Projekt ==**

+

**Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.<br>**

+

**Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.<br>**

+

**Weiterer Vorteil ist die Möglichkeit des "Multibaud" Digiq, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG).**

+

**In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.**

+

**[[Bild:Soundmodem-box.gif|Soundmodem Schema]]**

- +
- + **==Der Source Code==**
- + [\[https://github.com/oe5hpm/dxIAPRS\]](https://github.com/oe5hpm/dxIAPRS)
- + <https://github.com/oe5hpm/dxIAPRS><br>
- + [\[http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/\]](http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/)<http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/>
- +
- + **==Der kompilierte Treiber==**
- + [\[\[Media:soundmodem i386 linux. zip|Soundmodem-bin\]\]](#) - Der fertig kompilierte Soundmodem Treiber
- +
- + [\[\[Media:udpbox i386 linux bin. zip|udpbox-bin\]\]](#) - UDP Filter und RAW-Monitor Konverter und (neu) mit aprs-diqr, Bake, User-Message-Receiver
- +
- + [\[\[Media:udpgate i386 linux bin. zip|udpgate-bin\]\]](#) - I-Gate, APRS-IS, APRS-Server mit Rangefilter, HTML-Statistik, Log
- +
- + [\[\[Media:udphub i386 linux bin. zip|udphub-bin\]\]](#) - axudp Hub zum HAMNET-PR-Login ohne IP Beschränkung
- +
- + [\[\[Media:udpflex i386 linux bin. zip|udpflex-bin\]\]](#) - Interface com-port (/dev/ttySxx) mit KISS oder RMNC bidirektional auf axudp
- +
- + **==Starten bzw. Aufrufen des Treibers==**

- + mit oss testen 1200 + 9600 baud monitor (ohne kiss oder udp)
- + `./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g`
- +
- + mit alsa:
- + `aoss ./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g`
- +
- + APRS mit Xastir KISS-Interface, PTT auf ttyS0:
- + `aoss ./afskmodem -i /tmp /soundmodem -t /dev/ttyS0 -f 32000 -M 0 -i`
- + Xastir
- + `"interface" > "interface control" > "add" "serial kiss tnc"`
- + `"add" "tnc port" /tmp/soundmodem`
- + `"interface control" "start"`
- +
- + 2-Frequenz-halbduplex-Diqi mit 1200 / 1200+9600Bd xnet mit UDP und LPT PTT:<br>
- + Bei UDP ist die Startreihenfolge egal, die Programme können auch auf verschiedenen Rechnern laufen
- + `sudo nice -n -19 aoss ./afskmodem \`
- + `-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \`
- + `-C 0 -b 1 -r 300 -C 1 -b 2 \`
- + `-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 0 \`
- + `-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 0 \`

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U  
127.0.0.1:9202/9212 -m 0**

+ **linuxsnet (XNET) AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9200 d9210  
127.0.0.1**

+ **attach ip1 axudp 2 1 l9201 d9211  
127.0.0.1**

+ **attach ip2 axudp 3 1 l9202 d9212  
127.0.0.1**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+

+ **XNET mit KISS und TTY ptt (XNET  
nach dem Modem starten!)**

+ **aoss ./afskmodem \**

+ **-t /dev/ttyS0 -f 24000 -i /tmp  
/soundmodem \**

+ **-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \**

+ **-M 0 -c 1 -b 1200 -q 200 -m 0 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \**

+ **-M 2 -c 0 -b 9600 -a -g -q 200 -m 0**

+ **linuxsnet AUTOEXEC.NET**

+ **attach sdev0 kiss 1 3 38400 /tmp  
/soundmodem**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+

+ **144.800MHz 1200Bd. 70cm**  
**1200+9600Bd xnet, aprsdigi, aprsd**  
**und udpbox:**

+ **APRS hört auf allen Userzugängen**  
**und sendet zum IGATE. <br>**

+ **Senden auf 144.800 nur APRS**  
**Messages.<br>**

+ **Auf dem 1200Bd 70cm Zugang**  
**normales PR + APRS.<br>**

+

+ **1. Modem sendet alle Ports zu**  
**udpbox Port 920x und hört auf Port**  
**921x:<br>**

+ **(auf langsamen Rechnern oder bei**  
**hoher CPU last hilft Priotität mit nice**  
**oder renice erhöhen)**

+ **sudo nice -n -19 aoss ./afskmodem \**

+ **-p /dev/parport0 -f 44100 -c 2 -s 9 -l**  
**256 -b 6 -e 7 \**

+ **-C 0 -b 1 -r 300 -C 1 -b 2 \**

+ **-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:**  
**9200/9210 -m 2 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -U**  
**127.0.0.1:9201/9211 -m 2 \**

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U**  
**127.0.0.1:9202/9212 -m 2**

+

+ **2. XNET empfängt von udpbox und**  
**sendet direkt zum Modem:<br>**

+ **linuxsnet AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9300 d9210**  
**127.0.0.1**

+ **attach ip1 axudp 2 1 l9301 d9211**  
**127.0.0.1**

+ **attach ip2 axudp 3 1 l9302 d9212**  
**127.0.0.1**

- + **po 1 baud 1200**
- + **po 2 baud 1200**
- + **po 3 baud 9600**
- +
- + **3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)<br>**
- + **sendet alle UI zu aprsd auf 192.168.1.1:9000<br>**
- + **sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1 <br>**
- + **./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\**
- + **-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\**
- + **-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\**
- + **-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v**
- +
- + **4. udpbox als aprs-digi (Beispiel)<br>**
- + **-R empfangen axudp auf Port 9000**
- +
- + **-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt**
- +
- + **-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg**
- + **wie z.b. gar nicht aprs-frames, thirdparty-messages, Status-Meldungen**
- + **weq. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten**



- + (siehe aprs Protokollbeschreibung [APRS101.pdf](#))
- +
- + -x filtere Frames mit TCPIP oder NOCALL weg
- +
- + -p sende nur (soweit vom Absender richtig adressierte) Frames die
- + nach direkt gehört aussehen (first hop digi), sende aber den Rest
- + vom Pfad nach Protokoll modifiziert mit für weitere Hops
- + füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein
- + andernfalls bliebe der digi unsichtbar.
- + Es werden alle Adressierungsarten akzeptiert einschliesslich der
- + effizientesten mit Destination-SSID. Damit kann bei (insbesondere von
- + Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa
- + 30% eingespart werden.
- +
- + -t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg,
- + lasse aber (retryende) User Messages nach 28s durch
- +
- + -b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls
- + gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt
- + wurde, alle 30min

- +
- + **-k** Filtere alles (ausser User-msq) ausserhalb Umkreis koordinate(grad) /radius(km)
- +
- + **-c** sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt
- + mit Linefeed an Rechner 192.168.1.24: 2000
- +
- + **-r** sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100
- +
- + **-v** sagt was es tut und warum auf dem standard output
- +
- + **./udpbox -v -u OE0AAA-12:/tmp /msg12.txt -R 0.0.0.0:9000\**
- + **-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\**
- + **-p 5,6,7,8,9 -t 1680,28 -b 300: aprsbeacon.txt -k 48.2/-13.1/40\**
- + **-c 192.168.1.24:2000 -e -r 127.0.0.1: 9100<br>**
- +
- + **Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format**
- + **an zb. aprsd für lgate senden (-m 127.0.0.1:9304)**
- + **Weitere parameter siehe -h**
- +
- + **Bakentext File Beispiel:**
- + **(Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen**

- + aber Call und Koordinaten ausbessern)
- + Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit
- + zb. von einem Messwert Programm modifiziert werden.
- +
- + OE0AAA-11>TEST,TRACE2-2:!9000.00 N/18000.00E#PHG3750Test Digi
- +
- + dazu auf 266MHz Geode CPU mit billiq-USB-Sound"karte" optimierter Modemstart
- + (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)
- +
- + aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\
- + -l 128 -b 1 -M 0 -U 127.0.0.1:9000 /9100 -m 0 -e 50 &
- +
- + Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund
- +
- +
- + I-Gate mit udpgate
- +
- + Rx-Igate kompatibel nach:
- + [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)
- +
- + ./udpgate -R 0.0.0.0:9000 -t 14580 -s OE0AAA-10 -n 10:netbeacon.txt\

- + **-q www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501**
- +
- + **-t TCP Port für Connects mit Aprs-Gattern wie xastir oder weitere igates**
- +
- + **-s Call des Servers**
- +
- + **-n alle 10 Min Netzbake mit Server Position**
- + **File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!)**
- + **grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol**
- +
- + **!8959.00N/01300.20E&lgate Nordpol**
- +
- + **-q Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)**
- +
- + **-p passwort oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist**
- +
- + **-f Filterparameter werden zum Server gesendet**
- +
- + **-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel**
- + **(das File wird nach jeder Zeile geschlossen und kann gekürzt /gelöscht werden)**

```
graph TD
    subgraph Left_Path [ ]
        direction TB
        L1[== Das Projekt ==]
        L2[Dieser (USB) Soundkartentreiber befindet sich in der Entwicklung und soll es ermöglichen mit 2 Kanälen mehrere Modems zugleich unter Linux zu initialisieren.<br>]
        L3[Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei verwendung eines getrennten Senders und Empfängers.]
        L4>Weiterer Vorteil ist die Möglichkeit des "Multibaude" Digiq, also mehrere Geschwindigkeiten auf einer Frequenz, FSK und AFSK gemischt.
        L5>In Stereo kann theoretisch ein multibaude FSK AFSK KISS als auch AXUDP AX.25 Modem betrieben werden.
        L6[ ]
    end

    subgraph Right_Path [ ]
        direction TB
        R1[+ [ ]
        R2[-w www Port [ ]
        R3[+ [ ]
        R4[+ [ ]
        R5[+ [[Bild:udpqate-html.gif|center|www-Statistik-Beispiel]] [ ]
        R6[ ]
    end

    L6 --> D1[Dieses Projekt ist Open Source - Haftung, Verantwortung und Spaß übernimmt jeder selbst.]
    R6 --> D2[Dieses Projekt ist Open Source - Haftung, Verantwortung und Spaß übernimmt jeder selbst.]
```

The diagram illustrates two parallel development paths for a project. On the left, a vertical sequence of boxes describes project goals and features: starting with '== Das Projekt ==' followed by four orange boxes detailing a USB sound card driver, KISS drivers supporting up to 16 modems at various baud rates, Multibaud Digiq support, and stereo capability. This path concludes with a grey box stating the project is open source. On the right, a similar vertical sequence begins with a blue box containing '-w www Port', followed by three empty white boxes, then a blue box with an image placeholder, and ends with a blue box containing '<br>'. This path also concludes with a grey box stating the project is open source.

**Aktuelle Version vom 22. November 2019, 18:48 Uhr**

# Inhaltsverzeichnis

1 Das Projekt ..... 31

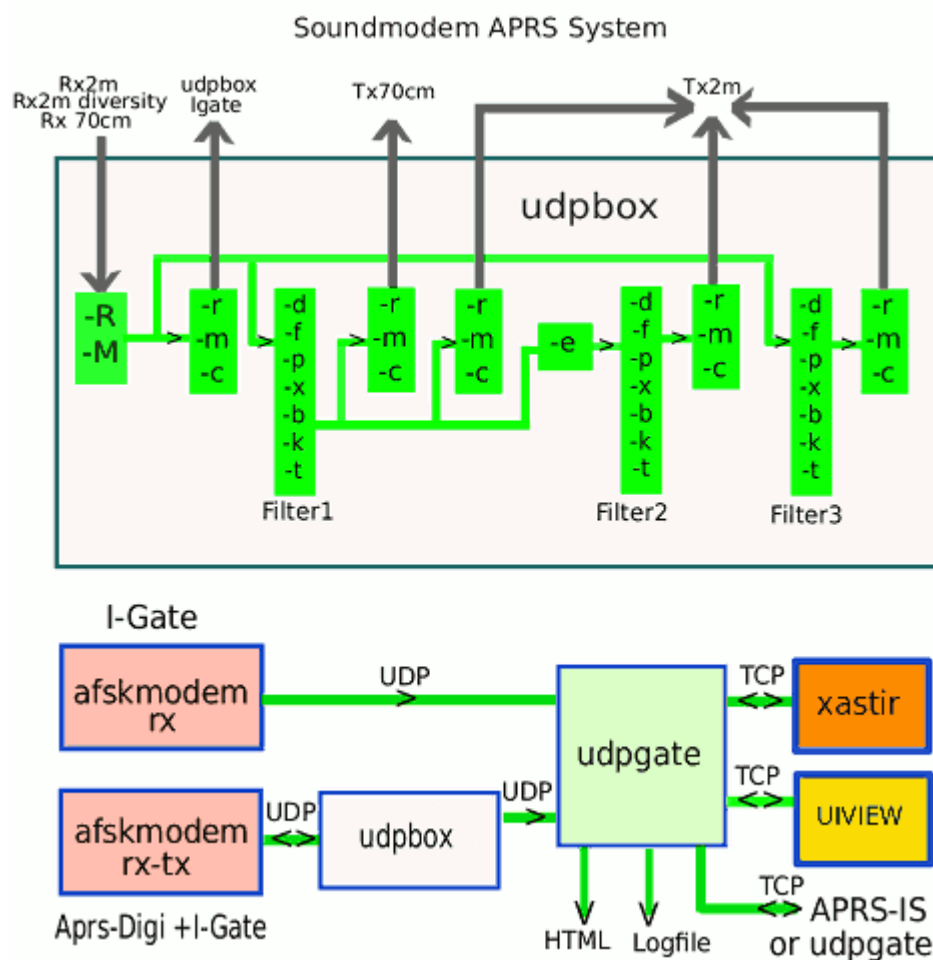
2 Der Source Code .....	31
3 Der kompilierte Treiber .....	31
4 Starten bzw. Aufrufen des Treibers .....	32

## Das Projekt

Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.

Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.

Weiterer Vorteil ist die Möglichkeit des **"Multibaud"** Digi, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG). In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.



## Der Source Code

[1]<https://github.com/oe5hpm/dxIAPRS>

[2]<http://gitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/>

## Der kompilierte Treiber

**Soundmodem-bin** - Der fertig kompilierte Soundmodem Treiber

**udpbox-bin** - UDP Filter und RAW-Monitor Konverter und (neu) mit aprs-digi, Bake, User-Message-Receiver

**udpgate-bin** - I-Gate, APRS-IS, APRS-Server mit Rangefilter, HTML-Statistik, Log

**udphub-bin** - axudp Hub zum HAMNET-PR-Login ohne IP Beschränkung

**udpflex-bin** - Interface com-port (/dev/ttySxx) mit KISS oder RMNC bidirektional auf axudp

## Starten bzw. Aufrufen des Treibers

mit oss testen 1200 + 9600 baud monitor (ohne kiss oder udp)

```
./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

mit alsa:

```
aoss ./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

APRS mit Xastir KISS-Interface, PTT auf ttyS0:

```
aoss ./afskmodem -i /tmp/soundmodem -t /dev/ttyS0 -f 32000 -M 0 -i
```

Xastir

```
"interface" > "interface control" > "add" "serial kiss tnc"  
"add" "tnc port" /tmp/soundmodem  
"interface control" "start"
```

2-Frequenz-halbduplex-Digi mit 1200 / 1200+9600Bd xnet mit UDP und LPT PTT:

Bei UDP ist die Startreihenfolge egal, die Programme können auch auf verschiedenen Rechnern laufen

```
sudo nice -n -19 aoss ./afskmodem \  
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \  
-C 0 -b 1 -r 300 -C 1 -b 2 \  
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 0 \  
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 0 \  
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 0
```

linuxsnet (XNET) AUTOEXEC.NET



```
attach ip0 axudp 1 1 l9200 d9210 127.0.0.1
attach ip1 axudp 2 1 l9201 d9211 127.0.0.1
attach ip2 axudp 3 1 l9202 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

XNET mit KISS und TTY ptt (XNET nach dem Modem starten!)

```
aoss ./afskmodem \
-t /dev/ttyS0 -f 24000 -i /tmp/soundmodem \
-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \
-M 0 -c 1 -b 1200 -q 200 -m 0 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \
-M 2 -c 0 -b 9600 -a -g -q 200 -m 0
```

linuxsnet AUTOEXEC.NET

```
attach sdev0 kiss 1 3 38400 /tmp/soundmodem
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

144.800MHz 1200Bd, 70cm 1200+9600Bd xnet, aprsdigi, aprsd und udpbox: APRS hört auf allen Userzugängen und sendet zum IGATE.

Senden auf 144.800 nur APRS Messages.

Auf dem 1200Bd 70cm Zugang normales PR + APRS.

1. Modem sendet alle Ports zu udpbox Port 920x und hört auf Port 921x:

(auf langsamen Rechnern oder bei hoher CPU last hilft Priotität mit nice oder renice erhöhen)

```
sudo nice -n -19 aoss ./afskmodem \
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \
-C 0 -b 1 -r 300 -C 1 -b 2 \
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 2 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 2 \
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 2
```

2. XNET empfängt von udpbox und sendet direkt zum Modem:

linuxsnet AUTOEXEC.NET

```
attach ip0 axudp 1 1 l9300 d9210 127.0.0.1
attach ip1 axudp 2 1 l9301 d9211 127.0.0.1
attach ip2 axudp 3 1 l9302 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)  
sendet alle UI zu aprsd auf 192.168.1.1:9000  
sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1

```
./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\  
-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\  
-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\  
-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v
```

4. udpbox als aprs-digi (Beispiel)

-R empfangen axudp auf Port 9000

-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt

-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg wie z.B. garnicht aprs-frames, thirdparty-messages, Status-Meldungen weg. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten (siehe aprs Protokollbeschreibung APRS101.pdf)

-x filtere Frames mit TCPIP oder NOCALL weg

-p sende nur (soweit vom Absender richtig adressierte) Frames die nach direkt gehört aussehen (first hop digi), sende aber den Rest vom Pfad nach Protokoll modifiziert mit für weitere Hops füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein anderfalls bliebe der digi unsichtbar. Es werden alle Adressierungsarten akzeptiert einschliesslich der effizientesten mit Destination-SSID. Damit kann bei (insbesondere von Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa 30% eingespart werden.

-t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg, lasse aber (retryende) User Messages nach 28s durch

-b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt wurde, alle 30min

-k Filtere alles (ausser User-msg) ausserhalb Umkreis koordinate(grad)/radius(km)

-c sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt mit Linefeed an Rechner 192.168.1.24:2000

-r sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100

-v sagt was es tut und warum auf dem standard output

```
./udpbox -v -u OE0AAA-12:/tmp/msg12.txt -R 0.0.0.0:9000\  
-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\  
-p 5,6,7,8,9 -t 1680,28 -b 300:aprsbeacon.txt -k 48.2/-13.1/40\  
-c 192.168.1.24:2000 -e -r 127.0.0.1:9100
```

Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format an zb. aprsd für lgate senden (-m 127.0.0.1:9304) Weitere parameter siehe -h

Bakentext File Beispiel: (Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen aber Call und Koordinaten ausbessern) Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit zb. von einem Messwert Programm modifiziert werden.

```
0E0AAA-11>TEST,TRACE2-2:!9000.00N/18000.00E#PHG3750Test Digi
```

dazu auf 266MHz Geode CPU mit billig-USB-Sound"karte" optimierter Modemstart (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)

```
aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\  
-l 128 -b 1 -M 0 -U 127.0.0.1:9000/9100 -m 0 -e 50 &
```

Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund

I-Gate mit udpgate

Rx-Igate kompatibel nach: [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)

```
./udpgate -R 0.0.0.0:9000 -t 14580 -s 0E0AAA-10 -n 10:netbeacon.txt\  
-g www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501
```

-t TCP Port für Connects mit Aprs-Gaffern wie xastir oder weitere igates

-s Call des Servers

-n alle 10 Min Netzbake mit Server Position File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!) grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol

```
!8959.00N/01300.20E&Igate Nordpol
```

-g Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)

-p password oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist

-f Filterparameter werden zum Server gesendet

-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel (das File wird nach jeder Zeile geschlossen und kann gekürzt/gelöscht werden)

-w www Port

[www-Statistik-Beispiel](#)

Dieses Projekt ist Open Source - Haftung, Verantwortung und Spaß übernimmt jeder selbst.

## Packet Radio via Soundkarte unter Linux: Unterschied zwischen den Versionen

[Versionsgeschichte interaktiv durchsuchen](#)

[Visuell Wikitext](#)

### Version vom 3. Mai 2011, 17:57 Uhr (Quelltext anzeigen)

[OE2WAO](#) ([Diskussion](#) | [Beiträge](#))

(Die Seite wurde neu angelegt: „[Kategorie: Packet-Radio und I-Gate](#) == Das Projekt == Dieser (USB) Soundkartentreiber befindet sich in der Entwicklung und soll es ermöglichen mit 2 Kanälen ...“)

### Aktuelle Version vom 22. November 2019, 18:48 Uhr (Quelltext anzeigen)

[OE5HPM](#) ([Diskussion](#) | [Beiträge](#))

([→Der Source Code](#))

(26 dazwischenliegende Versionen von 3 Benutzern werden nicht angezeigt)

Zeile 1:

[[Kategorie:Packet-Radio und I-Gate]]

Zeile 1:

[[Kategorie:Packet-Radio und I-Gate]]

+

**== Das Projekt ==**

+

**Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.<br>**

+

**Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.<br>**

+

**Weiterer Vorteil ist die Möglichkeit des "Multibaud" Digiq, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG).**

+

**In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.**

+

**[[Bild:Soundmodem-box.gif|Soundmodem Schema]]**

- +
- + **==Der Source Code==**
- + **[<https://github.com/oe5hpm/dxIAPRS>]**
- + **[https://github.com/oe5hpm](https://github.com/oe5hpm/dxIAPRS)**  
**/dxIAPRS<br>**
- + **[[http://qitlab.oe5xbl.ampr.org/oe5hpm](http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/)**  
**/dxIAPRS/]http://qitlab.oe5xbl.ampr.**  
**org/oe5hpm/dxIAPRS/**
- +
- + **==Der kompilierte Treiber==**
- + **[[Media:soundmodem i386 linux.**  
**zip|Soundmodem-bin]] - Der fertig**  
**kompilierte Soundmodem Treiber**
- +
- + **[[Media:udpbox i386 linux bin.**  
**zip|udpbox-bin]] - UDP Filter und**  
**RAW-Monitor Konverter und (neu) mit**  
**aprs-diqi, Bake, User-Message-**  
**Receiver**
- +
- + **[[Media:udpgate i386 linux bin.**  
**zip|udpgate-bin]] - I-Gate, APRS-IS,**  
**APRS-Server mit Rangefilter, HTML-**  
**Statistik, Log**
- +
- + **[[Media:udphub i386 linux bin.**  
**zip|udphub-bin]] - axudp Hub zum**  
**HAMNET-PR-Login ohne IP**  
**Beschränkung**
- +
- + **[[Media:udpflex i386 linux bin.**  
**zip|udpflex-bin]] - Interface com-port (**  
**/dev/ttySxx) mit KISS oder RMNC**  
**bidirektional auf axudp**
- +
- + **==Starten bzw. Aufrufen des**  
**Treibers==**

```

+ mit oss testen 1200 + 9600 baud
+ monitor (ohne kiss oder udp)
+
+ ./afskmodem -f 32000 -M 0 -c 0 -b
+ 1200 -M 1 -c 0 -b 9600 -a -g
+
+ mit alsa:
+
+ aoss ./afskmodem -f 32000 -M 0 -c 0 -
+ b 1200 -M 1 -c 0 -b 9600 -a -g
+
+
+ APRS mit Xastir KISS-Interface, PTT
+ auf ttyS0:
+
+ aoss ./afskmodem -i /tmp
+ /soundmodem -t /dev/ttyS0 -f 32000 -
+ M 0 -i
+
+ Xastir
+
+ "interface" > "interface control" >
+ "add" "serial kiss tnc"
+
+ "add" "tnc port" /tmp/soundmodem
+
+ "interface control" "start"
+
+
+ 2-Frequenz-halbduplex-Diqi mit 1200
+ / 1200+9600Bd xnet mit UDP und LPT
+ PTT:<br>
+
+ Bei UDP ist die Startreihenfolge egal,
+ die Programme können auch auf
+ verschiedenen Rechnern laufen
+
+ sudo nice -n -19 aoss ./afskmodem \
+
+ -p /dev/parport0 -f 44100 -c 2 -s 9 -l
+ 256 -b 6 -e 7 \
+
+ -C 0 -b 1 -r 300 -C 1 -b 2 \
+
+ -M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:
+ 9200/9210 -m 0 \
+
+ -M 1 -c 0 -b 1200 -H 40 -q 200 -U
+ 127.0.0.1:9201/9211 -m 0 \

```

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U  
127.0.0.1:9202/9212 -m 0**

+ **linuxsnet (XNET) AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9200 d9210  
127.0.0.1**

+ **attach ip1 axudp 2 1 l9201 d9211  
127.0.0.1**

+ **attach ip2 axudp 3 1 l9202 d9212  
127.0.0.1**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+

+ **XNET mit KISS und TTY ptt (XNET  
nach dem Modem starten!)**

+ **aoss ./afskmodem \**

+ **-t /dev/ttyS0 -f 24000 -i /tmp  
/soundmodem \**

+ **-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \**

+ **-M 0 -c 1 -b 1200 -q 200 -m 0 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \**

+ **-M 2 -c 0 -b 9600 -a -g -q 200 -m 0**

+ **linuxsnet AUTOEXEC.NET**

+ **attach sdev0 kiss 1 3 38400 /tmp  
/soundmodem**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+

+ **144.800MHz 1200Bd. 70cm**  
**1200+9600Bd xnet, aprsdigi, aprsd**  
**und udpbox:**

+ **APRS hört auf allen Userzugängen**  
**und sendet zum IGATE. <br>**

+ **Senden auf 144.800 nur APRS**  
**Messages.<br>**

+ **Auf dem 1200Bd 70cm Zugang**  
**normales PR + APRS.<br>**

+

+ **1. Modem sendet alle Ports zu**  
**udpbox Port 920x und hört auf Port**  
**921x:<br>**

+ **(auf langsamen Rechnern oder bei**  
**hoher CPU last hilft Priotität mit nice**  
**oder renice erhöhen)**

+ **sudo nice -n -19 aoss ./afskmodem \**

+ **-p /dev/parport0 -f 44100 -c 2 -s 9 -l**  
**256 -b 6 -e 7 \**

+ **-C 0 -b 1 -r 300 -C 1 -b 2 \**

+ **-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:**  
**9200/9210 -m 2 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -U**  
**127.0.0.1:9201/9211 -m 2 \**

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U**  
**127.0.0.1:9202/9212 -m 2**

+

+ **2. XNET empfängt von udpbox und**  
**sendet direkt zum Modem:<br>**

+ **linuxsnet AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9300 d9210**  
**127.0.0.1**

+ **attach ip1 axudp 2 1 l9301 d9211**  
**127.0.0.1**

+ **attach ip2 axudp 3 1 l9302 d9212**  
**127.0.0.1**



- + **po 1 baud 1200**
- + **po 2 baud 1200**
- + **po 3 baud 9600**
- +
- + **3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)<br>**
- + **sendet alle UI zu aprsd auf 192.168.1.1:9000<br>**
- + **sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1 <br>**
- + **./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\**
- + **-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\**
- + **-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\**
- + **-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v**
- +
- + **4. udpbox als aprs-digi (Beispiel)<br>**
- + **-R empfangen axudp auf Port 9000**
- +
- + **-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt**
- +
- + **-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg**
- + **wie z.b. gar nicht aprs-frames, thirdparty-messages, Status-Meldungen**
- + **weq. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten**

- + (siehe aprs Protokollbeschreibung [APRS101.pdf](#))
- +
- + -x filtere Frames mit TCPIP oder NOCALL weg
- +
- + -p sende nur (soweit vom Absender richtig adressierte) Frames die
- + nach direkt gehört aussehen (first hop digi), sende aber den Rest
- + vom Pfad nach Protokoll modifiziert mit für weitere Hops
- + füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein
- + andernfalls bliebe der digi unsichtbar.
- + Es werden alle Adressierungsarten akzeptiert einschliesslich der
- + effizientesten mit Destination-SSID. Damit kann bei (insbesondere von
- + Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa
- + 30% eingespart werden.
- +
- + -t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg,
- + lasse aber (retryende) User Messages nach 28s durch
- +
- + -b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls
- + gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt
- + wurde, alle 30min

- +
- + **-k** Filtere alles (ausser User-msq) ausserhalb Umkreis koordinate(grad) /radius(km)
- +
- + **-c** sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt
- + mit Linefeed an Rechner 192.168.1.24: 2000
- +
- + **-r** sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100
- +
- + **-v** sagt was es tut und warum auf dem standard output
- +
- + **./udpbox -v -u OE0AAA-12:/tmp /msg12.txt -R 0.0.0.0:9000\**
- + **-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\**
- + **-p 5,6,7,8,9 -t 1680,28 -b 300: aprsbeacon.txt -k 48.2/-13.1/40\**
- + **-c 192.168.1.24:2000 -e -r 127.0.0.1: 9100<br>**
- +
- + **Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format**
- + **an zb. aprsd für lgate senden (-m 127.0.0.1:9304)**
- + **Weitere parameter siehe -h**
- +
- + **Bakentext File Beispiel:**
- + **(Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen**

- + aber Call und Koordinaten ausbessern)
- + Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit
- + zb. von einem Messwert Programm modifiziert werden.
- +
- + OE0AAA-11>TEST,TRACE2-2:!9000.00 N/18000.00E#PHG3750Test Digi
- +
- + dazu auf 266MHz Geode CPU mit billiq-USB-Sound"karte" optimierter Modemstart
- + (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)
- +
- + aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\
- + -l 128 -b 1 -M 0 -U 127.0.0.1:9000 /9100 -m 0 -e 50 &
- +
- + Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund
- +
- +
- + I-Gate mit udpgate
- +
- + Rx-Igate kompatibel nach:
- + [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)
- +
- + ./udpgate -R 0.0.0.0:9000 -t 14580 -s OE0AAA-10 -n 10:netbeacon.txt\

+ **-q www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501**

+

+ **-t TCP Port für Connects mit Aprs-Gattern wie xastir oder weitere igates**

+

+ **-s Call des Servers**

+

+ **-n alle 10 Min Netzbake mit Server Position**

+ **File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!)**

+ **grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol**

+

+ **!8959.00N/01300.20E&lgate Nordpol**

+

+ **-q Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)**

+

+ **-p passwort oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist**

+

+ **-f Filterparameter werden zum Server gesendet**

+

+ **-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel**

+ **(das File wird nach jeder Zeile geschlossen und kann gekürzt /gelöscht werden)**

The diagram illustrates the structure of a project page. It is divided into two main sections: a table of contents on the left and a main content area on the right.

**Table of Contents (Left):**

- == Das Projekt ==**
- Dieser (USB) Soundkartentreiber befindet sich in der Entwicklung und soll es ermöglichen mit 2 Kanälen mehrere Modems zugleich unter Linux zu initialisieren.**
- Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei verwendung eines getrennten Senders und Empfängers.**
- Weiterer Vorteil ist die Möglichkeit des "Multibaud" Digi, also mehrere Geschwindigkeiten auf einer Frequenz, FSK und AFSK gemischt.**
- In Stereo kann theoretisch ein multibaud FSK AFSK KISS als auch AXUDP AX.25 Modem betrieben werden.**

**Main Content Area (Right):**

- [[Bild:udpqate-html.gif|center|www-Statistik-Beispiel]]**

The diagram shows that the table of contents on the left corresponds to the main content area on the right. The first item in the table of contents, "== Das Projekt ==", is highlighted in yellow, indicating it is the current page. The other items are highlighted in orange, indicating they are links to other pages. The main content area on the right shows the content of the current page, which is the "udpqate-html.gif" image.

**Aktuelle Version vom 22. November 2019, 18:48 Uhr**

# Inhaltsverzeichnis

1 Das Projekt ..... 48

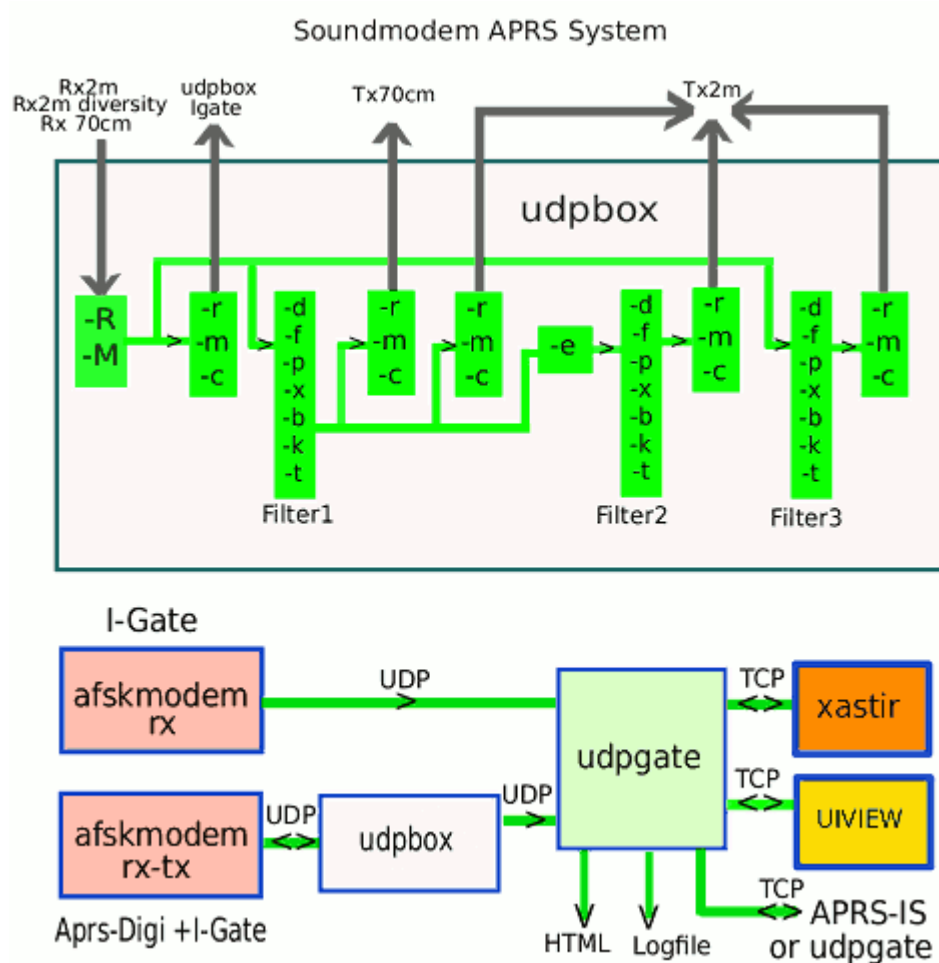
2 Der Source Code .....	48
3 Der kompilierte Treiber .....	48
4 Starten bzw. Aufrufen des Treibers .....	49

## Das Projekt

Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.

Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.

Weiterer Vorteil ist die Möglichkeit des **"Multibaud"** Digi, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG). In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.



## Der Source Code

[1]<https://github.com/oe5hpm/dxIAPRS>

[2]<http://gitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/>

## Der kompilierte Treiber

**Soundmodem-bin** - Der fertig kompilierte Soundmodem Treiber

**udpbox-bin** - UDP Filter und RAW-Monitor Konverter und (neu) mit aprs-digi, Bake, User-Message-Receiver



**udpgate-bin** - I-Gate, APRS-IS, APRS-Server mit Rangefilter, HTML-Statistik, Log

**udphub-bin** - axudp Hub zum HAMNET-PR-Login ohne IP Beschränkung

**udpflex-bin** - Interface com-port (/dev/ttySxx) mit KISS oder RMNC bidirektional auf axudp

## Starten bzw. Aufrufen des Treibers

mit oss testen 1200 + 9600 baud monitor (ohne kiss oder udp)

```
./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

mit alsa:

```
aoss ./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

APRS mit Xastir KISS-Interface, PTT auf ttyS0:

```
aoss ./afskmodem -i /tmp/soundmodem -t /dev/ttyS0 -f 32000 -M 0 -i
```

Xastir

```
"interface" > "interface control" > "add" "serial kiss tnc"
"add" "tnc port" /tmp/soundmodem
"interface control" "start"
```

2-Frequenz-halbduplex-Digi mit 1200 / 1200+9600Bd xnet mit UDP und LPT PTT:

Bei UDP ist die Startreihenfolge egal, die Programme können auch auf verschiedenen Rechnern laufen

```
sudo nice -n -19 aoss ./afskmodem \
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \
-C 0 -b 1 -r 300 -C 1 -b 2 \
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 0 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 0 \
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 0
```

linuxsnet (XNET) AUTOEXEC.NET

```
attach ip0 axudp 1 1 l9200 d9210 127.0.0.1
attach ip1 axudp 2 1 l9201 d9211 127.0.0.1
attach ip2 axudp 3 1 l9202 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

XNET mit KISS und TTY ptt (XNET nach dem Modem starten!)

```
aoss ./afskmodem \
-t /dev/ttyS0 -f 24000 -i /tmp/soundmodem \
-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \
-M 0 -c 1 -b 1200 -q 200 -m 0 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \
-M 2 -c 0 -b 9600 -a -g -q 200 -m 0
```

linuxsnet AUTOEXEC.NET

```
attach sdev0 kiss 1 3 38400 /tmp/soundmodem
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

144.800MHz 1200Bd, 70cm 1200+9600Bd xnet, aprsdigi, aprsd und udpbox: APRS hört auf allen Userzugängen und sendet zum IGATE.

Senden auf 144.800 nur APRS Messages.

Auf dem 1200Bd 70cm Zugang normales PR + APRS.

1. Modem sendet alle Ports zu udpbox Port 920x und hört auf Port 921x:

(auf langsamen Rechnern oder bei hoher CPU last hilft Priotität mit nice oder renice erhöhen)

```
sudo nice -n -19 aoss ./afskmodem \
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \
-C 0 -b 1 -r 300 -C 1 -b 2 \
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 2 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 2 \
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 2
```

2. XNET empfängt von udpbox und sendet direkt zum Modem:

linuxsnet AUTOEXEC.NET

```
attach ip0 axudp 1 1 l9300 d9210 127.0.0.1
attach ip1 axudp 2 1 l9301 d9211 127.0.0.1
attach ip2 axudp 3 1 l9302 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

### 3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)

sendet alle UI zu aprsd auf 192.168.1.1:9000

sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1

```
./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\  
-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\  
-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\  
-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v
```

### 4. udpbox als aprs-digi (Beispiel)

-R empfangen axudp auf Port 9000

-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt

-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg wie z.B. garnicht aprs-frames, thirdparty-messages, Status-Meldungen weg. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten (siehe aprs Protokollbeschreibung APRS101.pdf)

-x filtere Frames mit TCPIP oder NOCALL weg

-p sende nur (soweit vom Absender richtig adressierte) Frames die nach direkt gehört aussehen (first hop digi), sende aber den Rest vom Pfad nach Protokoll modifiziert mit für weitere Hops füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein anderfalls bliebe der digi unsichtbar. Es werden alle Adressierungsarten akzeptiert einschliesslich der effizientesten mit Destination-SSID. Damit kann bei (insbesondere von Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa 30% eingespart werden.

-t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg, lasse aber (retryende) User Messages nach 28s durch

-b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt wurde, alle 30min

-k Filtere alles (ausser User-msg) ausserhalb Umkreis koordinate(grad)/radius(km)

-c sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt mit Linefeed an Rechner 192.168.1.24:2000

-r sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100

-v sagt was es tut und warum auf dem standard output

```
./udpbox -v -u OE0AAA-12:/tmp/msg12.txt -R 0.0.0.0:9000\  
-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\  
-p 5,6,7,8,9 -t 1680,28 -b 300:aprsbeacon.txt -k 48.2/-13.1/40\  
-c 192.168.1.24:2000 -e -r 127.0.0.1:9100
```

Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format an zb. aprsd für lgate senden (-m 127.0.0.1:9304) Weitere parameter siehe -h

Bakentext File Beispiel: (Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen aber Call und Koordinaten ausbessern) Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit zb. von einem Messwert Programm modifiziert werden.

```
0E0AAA-11>TEST,TRACE2-2:!9000.00N/18000.00E#PHG3750Test Digi
```

dazu auf 266MHz Geode CPU mit billig-USB-Sound"karte" optimierter Modemstart (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)

```
aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\  
-l 128 -b 1 -M 0 -U 127.0.0.1:9000/9100 -m 0 -e 50 &
```

Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund

I-Gate mit udpgate

Rx-Igate kompatibel nach: [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)

```
./udpgate -R 0.0.0.0:9000 -t 14580 -s 0E0AAA-10 -n 10:netbeacon.txt\  
-g www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501
```

-t TCP Port für Connects mit Aprs-Gaffern wie xastir oder weitere igates

-s Call des Servers

-n alle 10 Min Netzbake mit Server Position File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!) grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol

```
!8959.00N/01300.20E&Igate Nordpol
```

-g Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)

-p password oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist

-f Filterparameter werden zum Server gesendet

-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel (das File wird nach jeder Zeile geschlossen und kann gekürzt/gelöscht werden)

-w www Port

www-Statistik-Beispiel

Dieses Projekt ist Open Source - Haftung, Verantwortung und Spaß übernimmt jeder selbst.

## Packet Radio via Soundkarte unter Linux: Unterschied zwischen den Versionen

[Versionsgeschichte interaktiv durchsuchen](#)

[Visuell Wikitext](#)

### Version vom 3. Mai 2011, 17:57 Uhr (Quelltext anzeigen)

[OE2WAO](#) ([Diskussion](#) | [Beiträge](#))

(Die Seite wurde neu angelegt: „[Kategorie: Packet-Radio und I-Gate](#) == Das Projekt == Dieser (USB) Soundkartentreiber befindet sich in der Entwicklung und soll es ermöglichen mit 2 Kanälen ...“)

### Aktuelle Version vom 22. November 2019, 18:48 Uhr (Quelltext anzeigen)

[OE5HPM](#) ([Diskussion](#) | [Beiträge](#))

([→Der Source Code](#))

(26 dazwischenliegende Versionen von 3 Benutzern werden nicht angezeigt)

Zeile 1:

[[Kategorie:Packet-Radio und I-Gate]]

Zeile 1:

[[Kategorie:Packet-Radio und I-Gate]]

+

== Das Projekt ==

+

Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.<br>

+

Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.<br>

+

Weiterer Vorteil ist die Möglichkeit des "Multibaud" Digiq, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG).

+

In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.

+

[[Bild:Soundmodem-box.gif|Soundmodem Schema]]

- +
- + **==Der Source Code==**
- + [\[https://github.com/oe5hpm/dxIAPRS\]](https://github.com/oe5hpm/dxIAPRS)
- + <https://github.com/oe5hpm/dxIAPRS><br>
- + [\[http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/\]](http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/)<http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/>
- +
- + **==Der kompilierte Treiber==**
- + [\[\[Media:soundmodem i386 linux. zip|Soundmodem-bin\]\]](#) - Der fertig kompilierte Soundmodem Treiber
- +
- + [\[\[Media:udpbox i386 linux bin. zip|udpbox-bin\]\]](#) - UDP Filter und RAW-Monitor Konverter und (neu) mit aprs-diqr, Bake, User-Message-Receiver
- +
- + [\[\[Media:udpgate i386 linux bin. zip|udpgate-bin\]\]](#) - I-Gate, APRS-IS, APRS-Server mit Rangefilter, HTML-Statistik, Log
- +
- + [\[\[Media:udphub i386 linux bin. zip|udphub-bin\]\]](#) - axudp Hub zum HAMNET-PR-Login ohne IP Beschränkung
- +
- + [\[\[Media:udpflex i386 linux bin. zip|udpflex-bin\]\]](#) - Interface com-port (/dev/ttySxx) mit KISS oder RMNC bidirektional auf axudp
- +
- + **==Starten bzw. Aufrufen des Treibers==**

```

+ mit oss testen 1200 + 9600 baud
+ monitor (ohne kiss oder udp)
+
+ ./afskmodem -f 32000 -M 0 -c 0 -b
+ 1200 -M 1 -c 0 -b 9600 -a -g
+
+ mit alsa:
+
+ aoss ./afskmodem -f 32000 -M 0 -c 0 -
+ b 1200 -M 1 -c 0 -b 9600 -a -g
+
+
+ APRS mit Xastir KISS-Interface, PTT
+ auf ttyS0:
+
+ aoss ./afskmodem -i /tmp
+ /soundmodem -t /dev/ttyS0 -f 32000 -
+ M 0 -i
+
+ Xastir
+
+ "interface" > "interface control" >
+ "add" "serial kiss tnc"
+
+ "add" "tnc port" /tmp/soundmodem
+
+ "interface control" "start"
+
+
+ 2-Frequenz-halbduplex-Diqi mit 1200
+ / 1200+9600Bd xnet mit UDP und LPT
+ PTT:<br>
+
+ Bei UDP ist die Startreihenfolge egal,
+ die Programme können auch auf
+ verschiedenen Rechnern laufen
+
+ sudo nice -n -19 aoss ./afskmodem \
+
+ -p /dev/parport0 -f 44100 -c 2 -s 9 -l
+ 256 -b 6 -e 7 \
+
+ -C 0 -b 1 -r 300 -C 1 -b 2 \
+
+ -M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:
+ 9200/9210 -m 0 \
+
+ -M 1 -c 0 -b 1200 -H 40 -q 200 -U
+ 127.0.0.1:9201/9211 -m 0 \

```

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U  
127.0.0.1:9202/9212 -m 0**

+ **linuxsnet (XNET) AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9200 d9210  
127.0.0.1**

+ **attach ip1 axudp 2 1 l9201 d9211  
127.0.0.1**

+ **attach ip2 axudp 3 1 l9202 d9212  
127.0.0.1**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+

+ **XNET mit KISS und TTY ptt (XNET  
nach dem Modem starten!)**

+ **aoss ./afskmodem \**

+ **-t /dev/ttyS0 -f 24000 -i /tmp  
/soundmodem \**

+ **-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \**

+ **-M 0 -c 1 -b 1200 -q 200 -m 0 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \**

+ **-M 2 -c 0 -b 9600 -a -g -q 200 -m 0**

+ **linuxsnet AUTOEXEC.NET**

+ **attach sdev0 kiss 1 3 38400 /tmp  
/soundmodem**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+



+ **144.800MHz 1200Bd. 70cm**  
+ **1200+9600Bd xnet, aprsdigi, aprsd**  
+ **und udpbox:**

+ **APRS hört auf allen Userzugängen**  
+ **und sendet zum IGATE. <br>**

+ **Senden auf 144.800 nur APRS**  
+ **Messages.<br>**

+ **Auf dem 1200Bd 70cm Zugang**  
+ **normales PR + APRS.<br>**

+

+ **1. Modem sendet alle Ports zu**  
+ **udpbox Port 920x und hört auf Port**  
+ **921x:<br>**

+ **(auf langsamen Rechnern oder bei**  
+ **hoher CPU last hilft Priotität mit nice**  
+ **oder renice erhöhen)**

+ **sudo nice -n -19 aoss ./afskmodem \**

+ **-p /dev/parport0 -f 44100 -c 2 -s 9 -l**  
+ **256 -b 6 -e 7 \**

+ **-C 0 -b 1 -r 300 -C 1 -b 2 \**

+ **-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:**  
+ **9200/9210 -m 2 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -U**  
+ **127.0.0.1:9201/9211 -m 2 \**

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U**  
+ **127.0.0.1:9202/9212 -m 2**

+

+ **2. XNET empfängt von udpbox und**  
+ **sendet direkt zum Modem:<br>**

+ **linuxsnet AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9300 d9210**  
+ **127.0.0.1**

+ **attach ip1 axudp 2 1 l9301 d9211**  
+ **127.0.0.1**

+ **attach ip2 axudp 3 1 l9302 d9212**  
+ **127.0.0.1**

- + **po 1 baud 1200**
- + **po 2 baud 1200**
- + **po 3 baud 9600**
- +
- + **3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)<br>**
- + **sendet alle UI zu aprsd auf 192.168.1.1:9000<br>**
- + **sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1 <br>**
- + **./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\**
- + **-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\**
- + **-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\**
- + **-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v**
- +
- + **4. udpbox als aprs-digi (Beispiel)<br>**
- + **-R empfangen axudp auf Port 9000**
- +
- + **-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt**
- +
- + **-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg**
- + **wie z.b. gar nicht aprs-frames, thirdparty-messages, Status-Meldungen**
- + **weq. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten**

- + (siehe aprs Protokollbeschreibung [APRS101.pdf](#))
- +
- + -x filtere Frames mit TCPIP oder NOCALL weg
- +
- + -p sende nur (soweit vom Absender richtig adressierte) Frames die
- + nach direkt gehört aussehen (first hop digi), sende aber den Rest
- + vom Pfad nach Protokoll modifiziert mit für weitere Hops
- + füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein
- + andernfalls bliebe der digi unsichtbar.
- + Es werden alle Adressierungsarten akzeptiert einschliesslich der
- + effizientesten mit Destination-SSID. Damit kann bei (insbesondere von
- + Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa
- + 30% eingespart werden.
- +
- + -t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg,
- + lasse aber (retryende) User Messages nach 28s durch
- +
- + -b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls
- + gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt
- + wurde, alle 30min

- +
- + `-k` Filtere alles (ausser User-msq) ausserhalb Umkreis koordinate(grad) /radius(km)
- +
- + `-c` sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt
- + mit Linefeed an Rechner 192.168.1.24: 2000
- +
- + `-r` sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100
- +
- + `-v` sagt was es tut und warum auf dem standard output
- +
- + `./udpbox -v -u OE0AAA-12:/tmp /msg12.txt -R 0.0.0.0:9000\`
- + `-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\`
- + `-p 5,6,7,8,9 -t 1680,28 -b 300: aprsbeacon.txt -k 48.2/-13.1/40\`
- + `-c 192.168.1.24:2000 -e -r 127.0.0.1: 9100<br>`
- +
- + Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format
- + an zb. aprsd für lgate senden (-m 127.0.0.1:9304)
- + Weitere parameter siehe -h
- +
- + Bakentext File Beispiel:
- + (Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen

- + aber Call und Koordinaten ausbessern)
- + Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit
- + zb. von einem Messwert Programm modifiziert werden.
- +
- + OE0AAA-11>TEST,TRACE2-2:!9000.00 N/18000.00E#PHG3750Test Digi
- +
- + dazu auf 266MHz Geode CPU mit billiq-USB-Sound"karte" optimierter Modemstart
- + (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)
- +
- + aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\
- + -l 128 -b 1 -M 0 -U 127.0.0.1:9000 /9100 -m 0 -e 50 &
- +
- + Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund
- +
- +
- + I-Gate mit udpgate
- +
- + Rx-Igate kompatibel nach: [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)
- +
- + ./udpgate -R 0.0.0.0:9000 -t 14580 -s OE0AAA-10 -n 10:netbeacon.txt\

+ **-q www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501**

+

+ **-t TCP Port für Connects mit Aprs-Gattern wie xastir oder weitere igates**

+

+ **-s Call des Servers**

+

+ **-n alle 10 Min Netzbake mit Server Position**

+ **File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!)**

+ **grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol**

+

+ **!8959.00N/01300.20E&lgate Nordpol**

+

+ **-q Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)**

+

+ **-p passwort oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist**

+

+ **-f Filterparameter werden zum Server gesendet**

+

+ **-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel**

+ **(das File wird nach jeder Zeile geschlossen und kann gekürzt /gelöscht werden)**

The diagram illustrates the structure of a project page. On the left, a table of contents lists sections with their corresponding line numbers. On the right, the main content area shows the actual HTML structure of the page, including a header, a table of contents, and a main content block.

Linie	Inhalt
1	== Das Projekt ==
2	Dieser (USB) Soundkartentreiber befindet sich in der Entwicklung und soll es ermöglichen mit 2 Kanälen mehrere Modems zugleich unter Linux zu initialisieren.
3	Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei verwendung eines getrennten Senders und Empfängers.
4	Weiterer Vorteil ist die Möglichkeit des "Multibaud" Digi, also mehrere Geschwindigkeiten auf einer Frequenz, FSK und AFSK gemischt.
5	In Stereo kann theoretisch ein multibaud FSK AFSK KISS als auch AXUDP AX.25 Modem betrieben werden.

The main content area on the right shows the following structure:

- Header: + [ ]
- Table of Contents: + [-w www Port] + [ ] + [ ] + [[Bild:udpqate-html.gif|center|www-Statistik-Beispiel]] + [ ]
- Main Content: + [ ]

The footer of the page contains the text: Dieses Projekt ist Open Source - Haftung, Verantwortung und Spaß übernimmt jeder selbst.

**Aktuelle Version vom 22. November 2019, 18:48 Uhr**

# Inhaltsverzeichnis

1 Das Projekt ..... 65

2 Der Source Code .....	65
3 Der kompilierte Treiber .....	65
4 Starten bzw. Aufrufen des Treibers .....	66

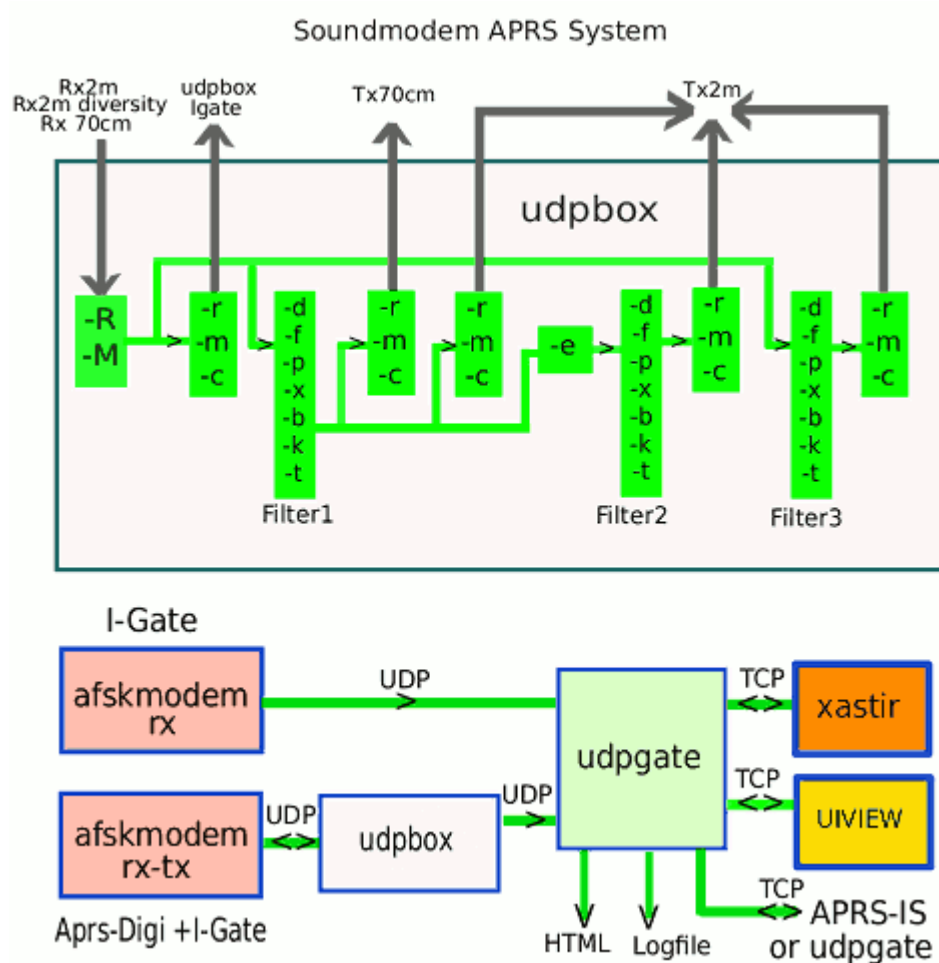


## Das Projekt

Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.

Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.

Weiterer Vorteil ist die Möglichkeit des **"Multibaud"** Digi, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG). In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.



## Der Source Code

[1]<https://github.com/oe5hpm/dxIAPRS>

[2]<http://gitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/>

## Der kompilierte Treiber

**Soundmodem-bin** - Der fertig kompilierte Soundmodem Treiber

**udpbox-bin** - UDP Filter und RAW-Monitor Konverter und (neu) mit aprs-digi, Bake, User-Message-Receiver

**udpgate-bin** - I-Gate, APRS-IS, APRS-Server mit Rangefilter, HTML-Statistik, Log

**udphub-bin** - axudp Hub zum HAMNET-PR-Login ohne IP Beschränkung

**udpflex-bin** - Interface com-port (/dev/ttySxx) mit KISS oder RMNC bidirektional auf axudp

## Starten bzw. Aufrufen des Treibers

mit oss testen 1200 + 9600 baud monitor (ohne kiss oder udp)

```
./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

mit alsa:

```
aoss ./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

APRS mit Xastir KISS-Interface, PTT auf ttyS0:

```
aoss ./afskmodem -i /tmp/soundmodem -t /dev/ttyS0 -f 32000 -M 0 -i
```

Xastir

```
"interface" > "interface control" > "add" "serial kiss tnc"  
"add" "tnc port" /tmp/soundmodem  
"interface control" "start"
```

2-Frequenz-halbduplex-Digi mit 1200 / 1200+9600Bd xnet mit UDP und LPT PTT:

Bei UDP ist die Startreihenfolge egal, die Programme können auch auf verschiedenen Rechnern laufen

```
sudo nice -n -19 aoss ./afskmodem \  
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \  
-C 0 -b 1 -r 300 -C 1 -b 2 \  
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 0 \  
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 0 \  
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 0
```

linuxsnet (XNET) AUTOEXEC.NET

```
attach ip0 axudp 1 1 l9200 d9210 127.0.0.1
attach ip1 axudp 2 1 l9201 d9211 127.0.0.1
attach ip2 axudp 3 1 l9202 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

XNET mit KISS und TTY ptt (XNET nach dem Modem starten!)

```
aoss ./afskmodem \
-t /dev/ttyS0 -f 24000 -i /tmp/soundmodem \
-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \
-M 0 -c 1 -b 1200 -q 200 -m 0 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \
-M 2 -c 0 -b 9600 -a -g -q 200 -m 0
```

linuxsnet AUTOEXEC.NET

```
attach sdev0 kiss 1 3 38400 /tmp/soundmodem
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

144.800MHz 1200Bd, 70cm 1200+9600Bd xnet, aprsdigi, aprsd und udpbox: APRS hört auf allen Userzugängen und sendet zum IGATE.

Senden auf 144.800 nur APRS Messages.

Auf dem 1200Bd 70cm Zugang normales PR + APRS.

1. Modem sendet alle Ports zu udpbox Port 920x und hört auf Port 921x:

(auf langsamen Rechnern oder bei hoher CPU last hilft Priotität mit nice oder renice erhöhen)

```
sudo nice -n -19 aoss ./afskmodem \
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \
-C 0 -b 1 -r 300 -C 1 -b 2 \
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 2 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 2 \
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 2
```

2. XNET empfängt von udpbox und sendet direkt zum Modem:

linuxsnet AUTOEXEC.NET

```
attach ip0 axudp 1 1 l9300 d9210 127.0.0.1
attach ip1 axudp 2 1 l9301 d9211 127.0.0.1
attach ip2 axudp 3 1 l9302 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

### 3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)

sendet alle UI zu aprsd auf 192.168.1.1:9000

sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1

```
./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\  
-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\  
-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\  
-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v
```

### 4. udpbox als aprs-digi (Beispiel)

-R empfangen axudp auf Port 9000

-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt

-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg wie z.B. garnicht aprs-frames, thirdparty-messages, Status-Meldungen weg. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten (siehe aprs Protokollbeschreibung APRS101.pdf)

-x filtere Frames mit TCPIP oder NOCALL weg

-p sende nur (soweit vom Absender richtig adressierte) Frames die nach direkt gehört aussehen (first hop digi), sende aber den Rest vom Pfad nach Protokoll modifiziert mit für weitere Hops füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein anderfalls bliebe der digi unsichtbar. Es werden alle Adressierungsarten akzeptiert einschliesslich der effizientesten mit Destination-SSID. Damit kann bei (insbesondere von Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa 30% eingespart werden.

-t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg, lasse aber (retryende) User Messages nach 28s durch

-b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt wurde, alle 30min

-k Filtere alles (ausser User-msg) ausserhalb Umkreis koordinate(grad)/radius(km)

-c sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt mit Linefeed an Rechner 192.168.1.24:2000

-r sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100

-v sagt was es tut und warum auf dem standard output

```
./udpbox -v -u OE0AAA-12:/tmp/msg12.txt -R 0.0.0.0:9000\  
-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\  
-p 5,6,7,8,9 -t 1680,28 -b 300:aprsbeacon.txt -k 48.2/-13.1/40\  
-c 192.168.1.24:2000 -e -r 127.0.0.1:9100
```

Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format an zb. aprsd für lgate senden (-m 127.0.0.1:9304) Weitere parameter siehe -h

Bakentext File Beispiel: (Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen aber Call und Koordinaten ausbessern) Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit zb. von einem Messwert Programm modifiziert werden.

```
0E0AAA-11>TEST,TRACE2-2:!9000.00N/18000.00E#PHG3750Test Digi
```

dazu auf 266MHz Geode CPU mit billig-USB-Sound"karte" optimierter Modemstart (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)

```
aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\  
-l 128 -b 1 -M 0 -U 127.0.0.1:9000/9100 -m 0 -e 50 &
```

Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund

I-Gate mit udpgate

Rx-Igate kompatibel nach: [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)

```
./udpgate -R 0.0.0.0:9000 -t 14580 -s 0E0AAA-10 -n 10:netbeacon.txt\  
-g www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501
```

-t TCP Port für Connects mit Aprs-Gaffern wie xastir oder weitere igates

-s Call des Servers

-n alle 10 Min Netzbake mit Server Position File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!) grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol

```
!8959.00N/01300.20E&Igate Nordpol
```

-g Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)

-p password oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist

-f Filterparameter werden zum Server gesendet

-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel (das File wird nach jeder Zeile geschlossen und kann gekürzt/gelöscht werden)

-w www Port

[www-Statistik-Beispiel](#)

Dieses Projekt ist Open Source - Haftung, Verantwortung und Spaß übernimmt jeder selbst.

---

## Seiten in der Kategorie „Packet-Radio und I-Gate“

---

Folgende 19 Seiten sind in dieser Kategorie, von 19 insgesamt.

### C

- [Convers](#)

### D

- [D4C - Digital4Capitals](#)
- [DX-Cluster](#)

### E

- [Email im digitalen Netz](#)

### I

- [IGATE](#)

### L

- [Links](#)
- [Linux und Amateur Packet Radio](#)
- [Linux und Schmalband Packet Radio mit Terminal](#)

### M

- [Mailbox - BBS](#)

### N

- [NF VOX PTT](#)

### P

- [Packet Radio via HAMNET](#)
- [Packet Radio via Soundkarte](#)
- [Packet Radio via Soundkarte unter Linux](#)
- [Packet Radio via TNC](#)
- [PR via Internet](#)
- [PTT Watchdog](#)

### Q

- [QTC-Net](#)

### S

- [SAMNET](#)

## T

- [TCE Tinycore Linux Projekt](#)

## Packet Radio via Soundkarte unter Linux: Unterschied zwischen den Versionen

[Versionsgeschichte interaktiv durchsuchen](#)

[Visuell Wikitext](#)

### Version vom 3. Mai 2011, 17:57 Uhr (Quelltext anzeigen)

[OE2WAO](#) ([Diskussion](#) | [Beiträge](#))

(Die Seite wurde neu angelegt: „[Kategorie: Packet-Radio und I-Gate](#) == Das Projekt == Dieser (USB) Soundkartentreiber befindet sich in der Entwicklung und soll es ermöglichen mit 2 Kanälen ...“)

### Aktuelle Version vom 22. November 2019, 18:48 Uhr (Quelltext anzeigen)

[OE5HPM](#) ([Diskussion](#) | [Beiträge](#))

([→Der Source Code](#))

(26 dazwischenliegende Versionen von 3 Benutzern werden nicht angezeigt)

Zeile 1:

[[Kategorie:Packet-Radio und I-Gate]]

Zeile 1:

[[Kategorie:Packet-Radio und I-Gate]]

+

== Das Projekt ==

+

Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.<br>

+

Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.<br>

+

Weiterer Vorteil ist die Möglichkeit des "Multibaud" Digiq, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG).

+

In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.

+

[[Bild:Soundmodem-box.gif|Soundmodem Schema]]



- +
- + **==Der Source Code==**
- + [\[https://github.com/oe5hpm/dxIAPRS\]](https://github.com/oe5hpm/dxIAPRS)
- + <https://github.com/oe5hpm/dxIAPRS><br>
- + [\[http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/\]](http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/)<http://qitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/>
- +
- + **==Der kompilierte Treiber==**
- + [\[\[Media:soundmodem i386 linux. zip|Soundmodem-bin\]\]](#) - Der fertig kompilierte Soundmodem Treiber
- +
- + [\[\[Media:udpbox i386 linux bin. zip|udpbox-bin\]\]](#) - UDP Filter und RAW-Monitor Konverter und (neu) mit aprs-diqr, Bake, User-Message-Receiver
- +
- + [\[\[Media:udpgate i386 linux bin. zip|udpgate-bin\]\]](#) - I-Gate, APRS-IS, APRS-Server mit Rangefilter, HTML-Statistik, Log
- +
- + [\[\[Media:udphub i386 linux bin. zip|udphub-bin\]\]](#) - axudp Hub zum HAMNET-PR-Login ohne IP Beschränkung
- +
- + [\[\[Media:udpflex i386 linux bin. zip|udpflex-bin\]\]](#) - Interface com-port (/dev/ttySxx) mit KISS oder RMNC bidirektional auf axudp
- +
- + **==Starten bzw. Aufrufen des Treibers==**

```

+ mit oss testen 1200 + 9600 baud
+ monitor (ohne kiss oder udp)
+
+ ./afskmodem -f 32000 -M 0 -c 0 -b
+ 1200 -M 1 -c 0 -b 9600 -a -g
+
+ mit alsa:
+
+ aoss ./afskmodem -f 32000 -M 0 -c 0 -
+ b 1200 -M 1 -c 0 -b 9600 -a -g
+
+
+ APRS mit Xastir KISS-Interface, PTT
+ auf ttyS0:
+
+ aoss ./afskmodem -i /tmp
+ /soundmodem -t /dev/ttyS0 -f 32000 -
+ M 0 -i
+
+ Xastir
+
+ "interface" > "interface control" >
+ "add" "serial kiss tnc"
+
+ "add" "tnc port" /tmp/soundmodem
+
+ "interface control" "start"
+
+
+ 2-Frequenz-halbduplex-Diqi mit 1200
+ / 1200+9600Bd xnet mit UDP und LPT
+ PTT:<br>
+
+ Bei UDP ist die Startreihenfolge egal,
+ die Programme können auch auf
+ verschiedenen Rechnern laufen
+
+ sudo nice -n -19 aoss ./afskmodem \
+
+ -p /dev/parport0 -f 44100 -c 2 -s 9 -l
+ 256 -b 6 -e 7 \
+
+ -C 0 -b 1 -r 300 -C 1 -b 2 \
+
+ -M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:
+ 9200/9210 -m 0 \
+
+ -M 1 -c 0 -b 1200 -H 40 -q 200 -U
+ 127.0.0.1:9201/9211 -m 0 \

```

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U  
127.0.0.1:9202/9212 -m 0**

+ **linuxsnet (XNET) AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9200 d9210  
127.0.0.1**

+ **attach ip1 axudp 2 1 l9201 d9211  
127.0.0.1**

+ **attach ip2 axudp 3 1 l9202 d9212  
127.0.0.1**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+

+ **XNET mit KISS und TTY ptt (XNET  
nach dem Modem starten!)**

+ **aoss ./afskmodem \**

+ **-t /dev/ttyS0 -f 24000 -i /tmp  
/soundmodem \**

+ **-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \**

+ **-M 0 -c 1 -b 1200 -q 200 -m 0 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \**

+ **-M 2 -c 0 -b 9600 -a -g -q 200 -m 0**

+ **linuxsnet AUTOEXEC.NET**

+ **attach sdev0 kiss 1 3 38400 /tmp  
/soundmodem**

+ **po 1 baud 1200**

+ **po 2 baud 1200**

+ **po 3 baud 9600**

+

+

+ **144.800MHz 1200Bd. 70cm**  
**1200+9600Bd xnet, aprsdigi, aprsd**  
**und udpbox:**

+ **APRS hört auf allen Userzugängen**  
**und sendet zum IGATE. <br>**

+ **Senden auf 144.800 nur APRS**  
**Messages.<br>**

+ **Auf dem 1200Bd 70cm Zugang**  
**normales PR + APRS.<br>**

+

+ **1. Modem sendet alle Ports zu**  
**udpbox Port 920x und hört auf Port**  
**921x:<br>**

+ **(auf langsamen Rechnern oder bei**  
**hoher CPU last hilft Priotität mit nice**  
**oder renice erhöhen)**

+ **sudo nice -n -19 aoss ./afskmodem \**

+ **-p /dev/parport0 -f 44100 -c 2 -s 9 -l**  
**256 -b 6 -e 7 \**

+ **-C 0 -b 1 -r 300 -C 1 -b 2 \**

+ **-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:**  
**9200/9210 -m 2 \**

+ **-M 1 -c 0 -b 1200 -H 40 -q 200 -U**  
**127.0.0.1:9201/9211 -m 2 \**

+ **-M 2 -c 0 -b 9600 -a -q -q 200 -U**  
**127.0.0.1:9202/9212 -m 2**

+

+ **2. XNET empfängt von udpbox und**  
**sendet direkt zum Modem:<br>**

+ **linuxsnet AUTOEXEC.NET**

+ **attach ip0 axudp 1 1 l9300 d9210**  
**127.0.0.1**

+ **attach ip1 axudp 2 1 l9301 d9211**  
**127.0.0.1**

+ **attach ip2 axudp 3 1 l9302 d9212**  
**127.0.0.1**

- + **po 1 baud 1200**
- + **po 2 baud 1200**
- + **po 3 baud 9600**
- +
- + **3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)<br>**
- + **sendet alle UI zu aprsd auf 192.168.1.1:9000<br>**
- + **sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1 <br>**
- + **./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\**
- + **-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\**
- + **-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\**
- + **-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v**
- +
- + **4. udpbox als aprs-digi (Beispiel)<br>**
- + **-R empfangen axudp auf Port 9000**
- +
- + **-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt**
- +
- + **-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg**
- + **wie z.b. gar nicht aprs-frames, thirdparty-messages, Status-Meldungen**
- + **weq. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten**

- + (siehe aprs Protokollbeschreibung [APRS101.pdf](#))
- +
- + -x filtere Frames mit TCPIP oder NOCALL weg
- +
- + -p sende nur (soweit vom Absender richtig adressierte) Frames die
- + nach direkt gehört aussehen (first hop digi), sende aber den Rest
- + vom Pfad nach Protokoll modifiziert mit für weitere Hops
- + füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein
- + andernfalls bliebe der digi unsichtbar.
- + Es werden alle Adressierungsarten akzeptiert einschliesslich der
- + effizientesten mit Destination-SSID. Damit kann bei (insbesondere von
- + Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa
- + 30% eingespart werden.
- +
- + -t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg,
- + lasse aber (retryende) User Messages nach 28s durch
- +
- + -b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls
- + gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt
- + wurde, alle 30min

- +
- + `-k` Filtere alles (ausser User-msq) ausserhalb Umkreis koordinate(grad) /radius(km)
- +
- + `-c` sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt
- + mit Linefeed an Rechner 192.168.1.24: 2000
- +
- + `-r` sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100
- +
- + `-v` sagt was es tut und warum auf dem standard output
- +
- + `./udpbox -v -u OE0AAA-12:/tmp /msg12.txt -R 0.0.0.0:9000\`
- + `-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\`
- + `-p 5,6,7,8,9 -t 1680,28 -b 300: aprsbeacon.txt -k 48.2/-13.1/40\`
- + `-c 192.168.1.24:2000 -e -r 127.0.0.1: 9100<br>`
- +
- + Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format
- + an zb. aprsd für lgate senden (-m 127.0.0.1:9304)
- + Weitere parameter siehe -h
- +
- + Bakentext File Beispiel:
- + (Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen

- + aber Call und Koordinaten ausbessern)
- + Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit
- + zb. von einem Messwert Programm modifiziert werden.
- +
- + OE0AAA-11>TEST,TRACE2-2:!9000.00 N/18000.00E#PHG3750Test Digi
- +
- + dazu auf 266MHz Geode CPU mit billiq-USB-Sound"karte" optimierter Modemstart
- + (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)
- +
- + aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\
- + -l 128 -b 1 -M 0 -U 127.0.0.1:9000 /9100 -m 0 -e 50 &
- +
- + Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund
- +
- +
- + I-Gate mit udpgate
- +
- + Rx-Igate kompatibel nach:
- + [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)
- +
- + ./udpgate -R 0.0.0.0:9000 -t 14580 -s OE0AAA-10 -n 10:netbeacon.txt\



+ **-q www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501**

+

+ **-t TCP Port für Connects mit Aprs-Gattern wie xastir oder weitere igates**

+

+ **-s Call des Servers**

+

+ **-n alle 10 Min Netzbake mit Server Position**

+ **File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!)**

+ **grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol**

+

+ **!8959.00N/01300.20E&lgate Nordpol**

+

+ **-q Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)**

+

+ **-p passwort oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist**

+

+ **-f Filterparameter werden zum Server gesendet**

+

+ **-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel**

+ **(das File wird nach jeder Zeile geschlossen und kann gekürzt /gelöscht werden)**

The diagram illustrates the structure of a project page. On the left, a table of contents lists sections with their corresponding line numbers. On the right, the main content area shows the actual HTML structure of the page, including a header, a list of sections, and a footer.

Linie	Inhalt
1	== Das Projekt ==
2	Dieser (USB) Soundkartentreiber befindet sich in der Entwicklung und soll es ermöglichen mit 2 Kanälen mehrere Modems zugleich unter Linux zu initialisieren.
3	Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei verwendung eines getrennten Senders und Empfängers.
4	Weiterer Vorteil ist die Möglichkeit des "Multibaud" Digi, also mehrere Geschwindigkeiten auf einer Frequenz, FSK und AFSK gemischt.
5	In Stereo kann theoretisch ein multibaud FSK AFSK KISS als auch AXUDP AX.25 Modem betrieben werden.

The main content area on the right shows the following structure:

- Header: + [ ]
- Section 1: + [-w www Port [ ]]
- Section 2: + [ ]
- Section 3: + [ ]
- Section 4: + [[Bild:udpqate-html.gif|center|www-Statistik-Beispiel]] [ ]
- Section 5: + [ ]
- Section 6: + <br> [ ]
- Footer: Dieses Projekt ist Open Source - Haftung, Verantwortung und Spaß übernimmt jeder selbst.

**Aktuelle Version vom 22. November 2019, 18:48 Uhr**

# Inhaltsverzeichnis

1 Das Projekt ..... 84

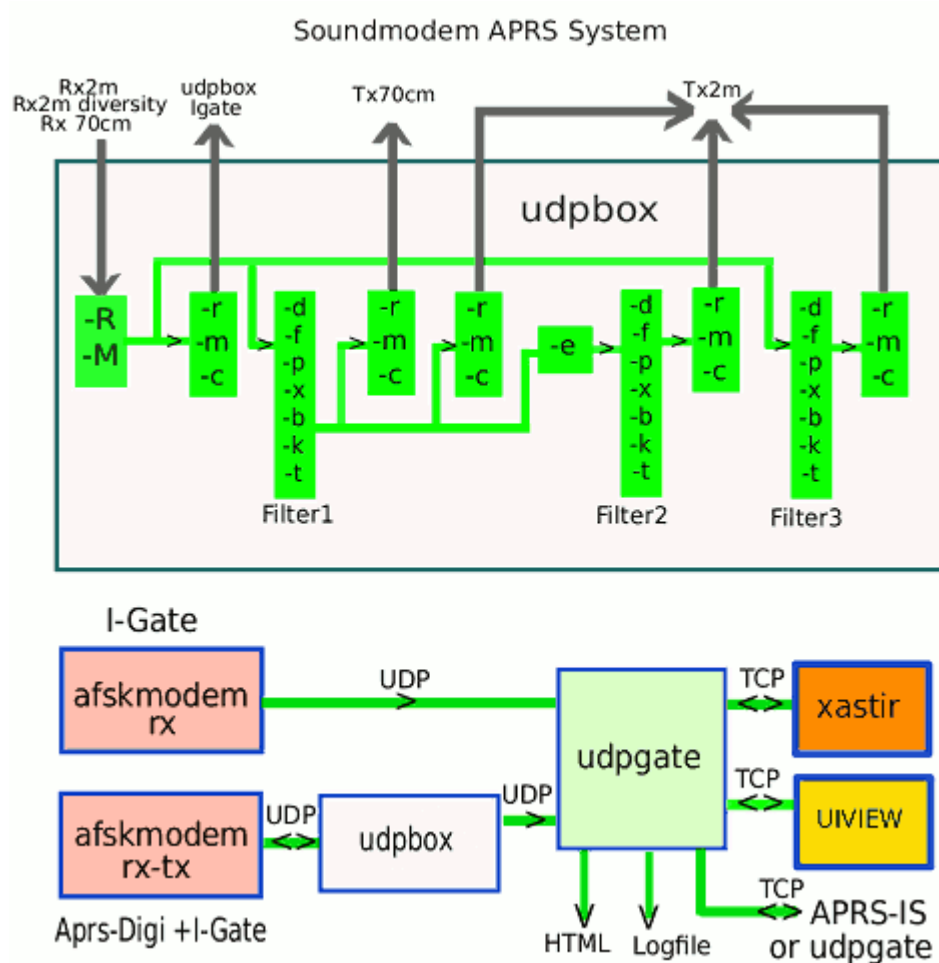
2 Der Source Code .....	84
3 Der kompilierte Treiber .....	84
4 Starten bzw. Aufrufen des Treibers .....	85

## Das Projekt

Dieser (USB) Soundkartentreiber von OE5DXL soll es ermöglichen mit 2 Kanälen (L und R der Soundkarte) mehrere Modems zugleich unter Linux zu initialisieren.

Als KISS Treiber sind bis zu 16 Modems von 1baud bis 28kbaud möglich. Der Equalizer ermöglicht einen Vollduplexbetrieb bei Verwendung eines getrennten Senders und Empfängers.

Weiterer Vorteil ist die Möglichkeit des **"Multibaud"** Digi, also mehrere Geschwindigkeiten FSK AFSK gemischt (bspw. 1k2 2k4 4k8 9k6 auf einer QRG). In Stereo kann so theoretisch ein multibaud FSK AFSK KISS, als auch AXUDP AX.25 Modem betrieben werden.



## Der Source Code

[1]<https://github.com/oe5hpm/dxIAPRS>

[2]<http://gitlab.oe5xbl.ampr.org/oe5hpm/dxIAPRS/>

## Der kompilierte Treiber

**Soundmodem-bin** - Der fertig kompilierte Soundmodem Treiber

**udpbox-bin** - UDP Filter und RAW-Monitor Konverter und (neu) mit aprs-digi, Bake, User-Message-Receiver

**udpgate-bin** - I-Gate, APRS-IS, APRS-Server mit Rangefilter, HTML-Statistik, Log

**udphub-bin** - axudp Hub zum HAMNET-PR-Login ohne IP Beschränkung

**udpflex-bin** - Interface com-port (/dev/ttySxx) mit KISS oder RMNC bidirektional auf axudp

## Starten bzw. Aufrufen des Treibers

mit oss testen 1200 + 9600 baud monitor (ohne kiss oder udp)

```
./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

mit alsa:

```
aoss ./afskmodem -f 32000 -M 0 -c 0 -b 1200 -M 1 -c 0 -b 9600 -a -g
```

APRS mit Xastir KISS-Interface, PTT auf ttyS0:

```
aoss ./afskmodem -i /tmp/soundmodem -t /dev/ttyS0 -f 32000 -M 0 -i
```

Xastir

```
"interface" > "interface control" > "add" "serial kiss tnc"  
"add" "tnc port" /tmp/soundmodem  
"interface control" "start"
```

2-Frequenz-halbduplex-Digi mit 1200 / 1200+9600Bd xnet mit UDP und LPT PTT:

Bei UDP ist die Startreihenfolge egal, die Programme können auch auf verschiedenen Rechnern laufen

```
sudo nice -n -19 aoss ./afskmodem \  
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \  
-C 0 -b 1 -r 300 -C 1 -b 2 \  
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 0 \  
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 0 \  
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 0
```

linuxsnet (XNET) AUTOEXEC.NET

```
attach ip0 axudp 1 1 l9200 d9210 127.0.0.1
attach ip1 axudp 2 1 l9201 d9211 127.0.0.1
attach ip2 axudp 3 1 l9202 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

XNET mit KISS und TTY ptt (XNET nach dem Modem starten!)

```
aoss ./afskmodem \
-t /dev/ttyS0 -f 24000 -i /tmp/soundmodem \
-c 2 -s 9 -l 256 -b 6 -e 7 -C 0 -r 300 \
-M 0 -c 1 -b 1200 -q 200 -m 0 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -m 0 \
-M 2 -c 0 -b 9600 -a -g -q 200 -m 0
```

linuxsnet AUTOEXEC.NET

```
attach sdev0 kiss 1 3 38400 /tmp/soundmodem
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

144.800MHz 1200Bd, 70cm 1200+9600Bd xnet, aprsdigi, aprsd und udpbox: APRS hört auf allen Userzugängen und sendet zum IGATE.

Senden auf 144.800 nur APRS Messages.

Auf dem 1200Bd 70cm Zugang normales PR + APRS.

1. Modem sendet alle Ports zu udpbox Port 920x und hört auf Port 921x:

(auf langsamen Rechnern oder bei hoher CPU last hilft Priotität mit nice oder renice erhöhen)

```
sudo nice -n -19 aoss ./afskmodem \
-p /dev/parport0 -f 44100 -c 2 -s 9 -l 256 -b 6 -e 7 \
-C 0 -b 1 -r 300 -C 1 -b 2 \
-M 0 -c 1 -b 1200 -q 200 -U 127.0.0.1:9200/9210 -m 2 \
-M 1 -c 0 -b 1200 -H 40 -q 200 -U 127.0.0.1:9201/9211 -m 2 \
-M 2 -c 0 -b 9600 -a -g -q 200 -U 127.0.0.1:9202/9212 -m 2
```

2. XNET empfängt von udpbox und sendet direkt zum Modem:

linuxsnet AUTOEXEC.NET

```
attach ip0 axudp 1 1 l9300 d9210 127.0.0.1
attach ip1 axudp 2 1 l9301 d9211 127.0.0.1
attach ip2 axudp 3 1 l9302 d9212 127.0.0.1
po 1 baud 1200
po 2 baud 1200
po 3 baud 9600
```

### 3. udpbox empfängt vom Modem (Port 920x) in AXUDP (9401)

sendet alle UI zu aprsd auf 192.168.1.1:9000

sendet nur "APRS Messages" (-f p58) zu Modem Funkport 1

```
./udpbox -R 0.0.0.0:9200 -m 192.168.1.1:9000 -r 127.0.0.1:93:9300\  
-R 0.0.0.0:9201 -m 192.168.1.1:9000 -r 127.0.0.1:93:9301\  
-R 0.0.0.0:9202 -m 192.168.1.1:9000 -r 127.0.0.1:93:9302\  
-M 0.0.0.0:9401 -r 127.0.0.1:9211 -f p58 -r 127.0.0.1:9210 -v
```

### 4. udpbox als aprs-digi (Beispiel)

-R empfangen axudp auf Port 9000

-u bestätige und speichere User Messages an OE0AAA-12 in File /tmp/msg12.txt

-f filtere für die überlastete 144.800 je nach Geschmack Data-Typen weg wie z.B. garnicht aprs-frames, thirdparty-messages, Status-Meldungen weg. Die Zahlen sind der Dezimalwert des 1. Bytes der Nutzdaten (siehe aprs Protokollbeschreibung APRS101.pdf)

-x filtere Frames mit TCPIP oder NOCALL weg

-p sende nur (soweit vom Absender richtig adressierte) Frames die nach direkt gehört aussehen (first hop digi), sende aber den Rest vom Pfad nach Protokoll modifiziert mit für weitere Hops füge das digicall OE0AAA-11 zur korrekten Pfad aufzeichnung ein anderfalls bliebe der digi unsichtbar. Es werden alle Adressierungsarten akzeptiert einschliesslich der effizientesten mit Destination-SSID. Damit kann bei (insbesondere von Mobilstationen gesendeten) Frames 14 byte "WIDE1-1,WIDE2-2..." oder etwa 30% eingespart werden.

-t filtere gleichbleibende Texte (sprich nervige Baken) 27min weg, lasse aber (retryende) User Messages nach 28s durch

-b sende Bake aus dem File aprsbeacon.txt alle 300s, aber ebenfalls gefiltert, also wenn der Text nicht zb. durch neue Wetterdaten ersetzt wurde, alle 30min

-k Filtere alles (ausser User-msg) ausserhalb Umkreis koordinate(grad)/radius(km)

-c sende zu Monitorzwecken (nc -l -u -p 2000) den sendefertigen Inhalt mit Linefeed an Rechner 192.168.1.24:2000

-r sende das gleiche(-e) zum Modem als axudp 127.0.0.1:9100

-v sagt was es tut und warum auf dem standard output

```
./udpbox -v -u OE0AAA-12:/tmp/msg12.txt -R 0.0.0.0:9000\  
-f d59,60,125,65-83,85-90,97-122 -x TCPIP,NOCALL -d OE0AAA-11\  
-p 5,6,7,8,9 -t 1680,28 -b 300:aprsbeacon.txt -k 48.2/-13.1/40\  
-c 192.168.1.24:2000 -e -r 127.0.0.1:9100
```

Man kann noch eine Kopie der ungefilterten rx Daten im monitor-format an zb. aprsd für lgate senden (-m 127.0.0.1:9304) Weitere parameter siehe -h

Bakentext File Beispiel: (Hier sollte man vorsichtig sein um keine Alarmsymbole zu erwischen aber Call und Koordinaten ausbessern) Es wird nur die 1. Zeile des Files gesendet, das File kann aber jederzeit zb. von einem Messwert Programm modifiziert werden.

```
0E0AAA-11>TEST,TRACE2-2:!9000.00N/18000.00E#PHG3750Test Digi
```

dazu auf 266MHz Geode CPU mit billig-USB-Sound"karte" optimierter Modemstart (-e 50 leichtes rx Hochpassfilter wegen dumpfem nf-Ausgang beim Rx)

```
aoss /home/tc/afskmodem -f 24000 -e 8 -t /dev/ttyS0\  
-l 128 -b 1 -M 0 -U 127.0.0.1:9000/9100 -m 0 -e 50 &
```

Startreihenfolge egal, mit & am Ende der Kommandozeile laufen die Programme im Hintergrund

I-Gate mit udpgate

Rx-Igate kompatibel nach: [http://wiki.ham.fi/APRS\\_iGate\\_properties#APRS-IS\\_connection\\_2](http://wiki.ham.fi/APRS_iGate_properties#APRS-IS_connection_2)

```
./udpgate -R 0.0.0.0:9000 -t 14580 -s 0E0AAA-10 -n 10:netbeacon.txt\  
-g www.db0anf.de 14580 -p /etc/pass.txt -f "m/30" -l 6:udp.log -w 14501
```

-t TCP Port für Connects mit Aprs-Gaffern wie xastir oder weitere igates

-s Call des Servers

-n alle 10 Min Netzbake mit Server Position File Inhalt (bitte richtige Koordinaten an den gleichen Spalten eingeben!) grad minuten.minutenkommas, "/" und "&" ist das Aprs-Symbol

```
!8959.00N/01300.20E&Igate Nordpol
```

-g Connect zum APRS-IS Netz oder anderem udpgate (Befehl wiederholen dann werden die Server bei Linkausfall der Reihe nach mit 30s Pause versucht)

-p password oder Filename mit Passwort damits in der Kommandozeile unsichtbar ist

-f Filterparameter werden zum Server gesendet

-l Loggt Connects, Frames (gute, gefilterte, duplikate) je nach loglevel (das File wird nach jeder Zeile geschlossen und kann gekürzt/gelöscht werden)

-w www Port

www-Statistik-Beispiel

Dieses Projekt ist Open Source - Haftung, Verantwortung und Spaß übernimmt jeder selbst.