Inhaltsverzeichnis

Linux und Schmalband Packet Radio mit Terminal

Versionsgeschichte interaktiv durchsuchen
VisuellWikitext

Version vom 22. April 2022, 18:38 Uhr (Quelltext anzeigen)	**Aktuelle Version vom 24. April 2022, 12:37 Uhr (Quelltext anzeigen)**
Oe1rsa (Diskussion | Beiträge)	Oe1rsa (Diskussion | Beiträge)
(Neu angelegt, in Arbeit.)	(wieder ein Stück weiter ...)
Markierung: Visuelle Bearbeitung	Markierung: Visuelle Bearbeitung

(Eine dazwischenliegende Version desselben Benutzers wird nicht angezeigt)

Zeile 1:	**Zeile 1:**
[[Kategorie:Packet-Radio und I-Gate]]	[[Kategorie:Packet-Radio und I-Gate]]
[[Datei:Funkpaketpost.png|alternativtext=Funkpaketpost bzw. Packet Radio|zentriert|rahmenlos|529x529px]]	[[Datei:Funkpaketpost.png|alternativtext=Funkpaketpost bzw. Packet Radio|zentriert|rahmenlos|529x529px]]
	+ **==Linux und Schmalband Packet Radio mit Terminal==**
− In diesem Artikel versuche ich zu zeigen, wie man mit Linux **''Bordmitteln''** eine Verbindung zu einem Packet Radio Knoten aufbauen kann. Wir kommen damit zum Sendebetrieb. Die Beschreibung eines geeigneten Adapterkabels und den Empfang von Signalen habe ich im Artikel **[[**Linux und Amateur Packet Radio**]]** beschrieben. Ich werde in diesem Artikel auf Software für die man den '''WINE''' Emulator ([https://wwwwinehw.org 1]) benötigt bewusst verzichten. Nicht, dass mich jemand falsch versteht: '''WINE''' ist ein großartiges Projekt und es ist absolut nichts falsch daran es zu verwenden, speziell dann wenn es keine andere Lösung gibt. Wir wollen es eben mal ohne ''Alkohol'' versuchen, hi.	+ In diesem Artikel versuche ich zu zeigen, wie man mit Linux **“Bordmitteln”** eine Verbindung zu einem Packet Radio Knoten aufbauen kann. Wir kommen damit zum Sendebetrieb. Die Beschreibung eines geeigneten Adapterkabels und den Empfang von Signalen habe ich im Artikel **''** Linux und Amateur Packet Radio**''** beschrieben. Ich werde in diesem Artikel auf Software für die man den ''WINE'' Emulator ([https://wwwwinehw.org 1]) benötigt bewusst verzichten. Nicht, dass mich jemand falsch versteht: ''WINE'' ist ein großartiges Projekt und es ist absolut nichts falsch daran es zu verwenden, speziell dann wenn es keine andere Lösung gibt. Wir wollen es eben mal ohne ''Alkohol'' versuchen, hi.

	Obwohl auch auf Kurzwelle nicht ganz unmöglich, so wird Schmalband Packet Radio normalerweise im 2m und 70cm Band über Transceiver gemacht, die nur FM Modulation beherrschen. Darauf konzentrieren wir uns auch.		Obwohl auch auf Kurzwelle nicht ganz unmöglich, so wird Schmalband Packet Radio normalerweise im 2m und 70cm Band über Transceiver gemacht, die nur FM Modulation beherrschen. Darauf konzentrieren wir uns auch.
−	Sehen wir uns an, was wir benötigen, so stoßen wir darauf, dass das ein so genannter ''TNC'', ein '''T'''erminal '''N'''ode '''C'''ontroller ist. Die Aufgabe eines **TNC's** ist es Daten, die wir in paketierter Form übergeben, unter Kontrolle eines Protokolles, in unserem Fall ''AX.25'', an eine Gegenstelle zu übermitteln.	+	Sehen wir uns an, was wir benötigen, so stoßen wir darauf, dass das ein so genannter ''TNC'', ein '''T'''erminal '''N'''ode '''C'''ontroller ist. Die Aufgabe eines **TNC’s** ist es Daten, die wir in paketierter Form übergeben, unter Kontrolle eines Protokolles, in unserem Fall ''AX.25'', an eine Gegenstelle zu übermitteln.
−	Solche **TNC's** waren ursprünglich, in den 80ern des 20. Jahrhunderts, reine Hardwarelösungen und deshalb auch recht unflexibel. Für die Abwicklung von Protokollen sind Computer prädestiniert und seitdem sie ausreichend hohe Geschwindigkeit haben können sie auch die Aufgaben der analogen Signalverarbeitung übernehmen, die in so einem TNC anfallen. Wer schon mit digitalen Übertragungsverfahren im Amateurfunk zu tun hatte weiß, dass oft eine Soundkarte ausreicht. So ist es auch in unserem Fall.	+	Solche **TNC’s** waren ursprünglich, in den 80ern des 20. Jahrhunderts, reine Hardwarelösungen und deshalb auch recht unflexibel. Für die Abwicklung von Protokollen sind Computer prädestiniert und seitdem sie ausreichend hohe Geschwindigkeit haben können sie auch die Aufgaben der analogen Signalverarbeitung übernehmen, die in so einem TNC anfallen. Wer schon mit digitalen Übertragungsverfahren im Amateurfunk zu tun hatte weiß, dass oft eine Soundkarte ausreicht. So ist es auch in unserem Fall.
	Trotzdem geht es nicht ganz ohne Hardware. Eine Verbindung zwischen Rig, also unserem Transceiver, und dem Computer muss her. Diese Verbindung besteht aus zwei Teilen:		Trotzdem geht es nicht ganz ohne Hardware. Eine Verbindung zwischen Rig, also unserem Transceiver, und dem Computer muss her. Diese Verbindung besteht aus zwei Teilen:
Zeile 15:		**Zeile 16:**	
	#eine digitale Schnittstelle für die ''PTT'' Steuerung, also die Sendertastung.		#eine digitale Schnittstelle für die ''PTT'' Steuerung, also die Sendertastung.

Die Beschreibung eines geeigneten Kabels findet sich, wie bereits gesagt, in dem Artikel **[[**Linux und Amateur Packet Radio**]]** . Wir werden in diesem Abschnitt nun zunächst den Sendefall vorbereiten. Es ist hilfreich wenn euch zusätzlich zum Transceiver ein Kontrollempfänger zur Verfügung steht. Ein Empfänger der FM demodulieren kann ist ausreichend. Wer einen Empfänger hat der auch SSB fähig ist auf 2m oder 70cm, der kann darüberhinaus den exakten Hub seiner Aussendung einstellen. Das Verfahren dazu heisst *Besselnull* und ich werde es eventuell an anderer Stelle im Wiki beschreiben. Wie gesagt, das ist aber nicht unbedingt nötig.	Die Beschreibung eines geeigneten Kabels findet sich, wie bereits gesagt, in dem Artikel ''Linux und Amateur Packet Radio''. Wir werden in diesem Abschnitt nun zunächst den Sendefall vorbereiten. Es ist hilfreich wenn euch zusätzlich zum Transceiver ein Kontrollempfänger zur Verfügung steht. Ein Empfänger der FM demodulieren kann ist ausreichend. Wer einen Empfänger hat der auch SSB fähig ist auf 2m oder 70cm, der kann darüberhinaus den exakten Hub seiner Aussendung einstellen. Das Verfahren dazu heisst ''Besselnull'' und ich werde es eventuell an anderer Stelle im Wiki beschreiben. Wie gesagt, das ist aber nicht unbedingt nötig.
Wir installieren die '''hamlib''' und '''sox'''. Dabei handelt es sich einerseits um eine Bibliothek, die es Applikationsprogrammen erlaubt verschiedene Rigs über eine einheitliche Schnittstelle anzusprechen und andererseits um ein vielseitiges Audio Werkzeug.	Wir installieren die '''hamlib''' und '''sox'''. Dabei handelt es sich einerseits um eine Bibliothek, die es Applikationsprogrammen erlaubt verschiedene Rigs über eine einheitliche Schnittstelle anzusprechen und andererseits um ein vielseitiges Audio Werkzeug.
<pre>	<pre>**sudo apt install libhamlib-utils libhamlib-doc sox</pre>**
	Wir könnten den Transceiver damit über die CAT Schnittstelle steuern und insbesonders die PTT Steuerung vornehmen. In meinem Beispiel mache ich aber davon keinen Gebrauch sondern beschränke mich darauf den RTS Pin der seriellen Schnittstelle zu schalten.
	Ein Hinweis scheint angebracht: Falls die Standard Pegellage des RTS Pins bei eurem Transceiver ''ungünstig''

	+ liegt, so kann es sein, dass der Transceiver sofort in den Sendebetrieb geht sobald das Kabel angeschlossen wird. Schließt das Kabel deshalb erst an sobald die Software eingerichtet ist.
	+
	+ Generell ist es kein Fehler zunächst einmal mit einem Multimeter die Pegel zu überprüfen bevor man das Funkgerät einstöpselt.
	+
	+ Nun kann man das Kommando rictl benutzen um manuell die PTT Funktion zu testen. Bei mir lautet das:
	+
	+ <pre>rictl -p /dev/ttyUSB1 -P RTS</pre>
	+ Welches die richtige serielle Schnittstelle ist müsst ihr zuvor herausfinden. Ich empfehle dazu sich einmal anzusehen welche Schnittstellen es im System gibt solange der USB Dongle nicht angeschlossen ist. Das geht mit
	+
	+ <pre>ls /dev/ttyUSB*</pre>
	+ Dann schließt man den Dongle an und gibt das Kommand erneut. Die Schnittstelle die dazugekommen ist muss die Gesuchte sein. Es gibt auch noch andere Methoden die dann zum Einsatz kommen sollten wenn man seine Installation dauerhaft machen will. Dazu mehr bei der Einrichtung des ''soundmodem''.
sudo apt install libhamlib-utils libhamlib-doc sox	Nachdem also das rictl Programm gestartet wurde kann man von dessen Kommandozeile interaktiv PTT

–	+ auf on oder off setzten. '''T 1''' sollte das RTS Signal einschalten und '''T 0''' aus. Wenn man die Funktion mit dem Multimeter validiert hat, so kann man den 6-pol mini DIN Stecker ans Funkgerät anstecken und den Versuch wiederholen. Der Transceiver sollte sich nun auftasten lassen.
– <\pre>	
– Wir könnten den Transceiver damit über die CAT Schnittstelle steuern und insbesonders die PTT Steuerung vornehmen. In meinem Beispiel mache ich aber davon keinen Gebrauch sondern beschränke mich darauf den RTS Pin der seriellen Schnittstelle zu schalten.	+ Immer noch an der Kunstantenne, ihr habt doch bis jetzt alles an der Dummy-Load gemacht, so hoffe ich, schalten wir nun unseren Kontrollempfänger ein und stellen ihn auf die Frequenz unseres Transceivers ein. Dann geben wir das Kommando
	+
	+ <pre>AUDIODEV=plughw:Device,0 play -n synth sine 1000 gain 0</pre>
	+ ein. ''Device'' ist dabei wieder der Name der Soundkarte, den wir schon bei der Inbetriebnahme der Empfangsseite verwendet haben. Der Wert ''1000'' steht für 1000Hz und der gain 0 bedeutet maximale Aussteuerung der Soundkarte. Es ist nun zu empfehlen den Trimmer, wo vorhanden, am Adapterkabel auf einen mittleren Wert zu stellen. Auch keine schlechte Idee ist es zunächst mit einem Kopfhörer zu überprüfen ob die 1000Hz am Ausgang der Soundkarte zu hören sind. Wenn alles klar ist so schließen wir das Kabel an den Transceiver an. Sobald wir die PTT mit Hilfe von ricgctl einschalten sollte das Signal nun auch im Kontrollempfänger zu hören sein. Man kann nun vorsichtig den Trimmer größer drehen, bzw. die Lautstärke am Regler der Soundkarte. Man sollte

dabei so vorgehen, dass das Signal möglichst groß wird, aber **keines Falls zu stark, da dann störende Verzerrungen auftreten. Mein ICOM Receiver macht es mit einfach, da er ganz einfach aufhört zu senden, sobald die Aussteuerung (**der **Hub)** zu **groß wird. Ich gehe bis an diese Grenze heran und drehe den Trimmer dann um einen ''Tick'' kleiner.**

+

+ **Wer will kann an dieser Stelle mit Direwolf weiter machen, oder meinem Weg folgen und das Soundmodem von Thomas Sailer installieren, allerdings in einer von mir überarbeiteten und weitergeführten Version**.

+

+ **'''''...wird fortgesetzt...'''''**

Aktuelle Version vom 24. April 2022, 12:37 Uhr

Linux und Schmalband Packet Radio mit Terminal

In diesem Artikel versuche ich zu zeigen, wie man mit Linux “Bordmitteln” eine Verbindung zu einem Packet Radio Knoten aufbauen kann. Wir kommen damit zum Sendebetrieb. Die Beschreibung eines geeigneten Adapterkabels und den Empfang von Signalen habe ich im Artikel *Linux und Amateur Packet Radio* beschrieben. Ich werde in diesem Artikel auf Software für die man den *WINE* Emulator (1) benötigt bewusst verzichten. Nicht, dass mich jemand falsch versteht: *WINE* ist ein großartiges Projekt und es ist absolut nichts falsch daran es zu verwenden, speziell dann wenn es keine andere Lösung gibt. Wir wollen es eben mal ohne *Alkohol* versuchen, hi.

Obwohl auch auf Kurzwelle nicht ganz unmöglich, so wird Schmalband Packet Radio normalerweise im 2m und 70cm Band über Transceiver gemacht, die nur FM Modulation beherrschen. Darauf konzentrieren wir uns auch.

Sehen wir uns an, was wir benötigen, so stoßen wir darauf, dass das ein so genannter *TNC*, ein **T** erminal **N**ode **C**ontroller ist. Die Aufgabe eines TNC’s ist es Daten, die wir in paketierter Form übergeben, unter Kontrolle eines Protokolles, in unserem Fall *AX.25*, an eine Gegenstelle zu übermitteln.

Solche TNC’s waren ursprünglich, in den 80ern des 20. Jahrhunderts, reine Hardwarelösungen und deshalb auch recht unflexibel. Für die Abwicklung von Protokollen sind Computer prädestiniert und seitdem sie ausreichend hohe Geschwindigkeit haben können sie auch die Aufgaben der analogen Signalverarbeitung übernehmen, die in so einem TNC anfallen. Wer schon mit digitalen Übertragungsverfahren im Amateurfunk zu tun hatte weiß, dass oft eine Soundkarte ausreicht. So ist es auch in unserem Fall.

Trotzdem geht es nicht ganz ohne Hardware. Eine Verbindung zwischen Rig, also unserem Transceiver, und dem Computer muss her. Diese Verbindung besteht aus zwei Teilen:

1. Eine analoge Schnittstelle für Audio Übertragung und
2. eine digitale Schnittstelle für die *PTT* Steuerung, also die Sendertastung.

Die Beschreibung eines geeigneten Kabels findet sich, wie bereits gesagt, in dem Artikel *Linux und Amateur Packet Radio*. Wir werden in diesem Abschnitt nun zunächst den Sendefall vorbereiten. Es ist hilfreich wenn euch zusätzlich zum Transceiver ein Kontrollempfänger zur Verfügung steht. Ein Empfänger der FM demodulieren kann ist ausreichend. Wer einen Empfänger hat der auch SSB fähig ist auf 2m oder 70cm, der kann darüberhinaus den exakten Hub seiner Aussendung einstellen. Das Verfahren dazu heisst *Besselnull* und ich werde es eventuell an anderer Stelle im Wiki beschreiben. Wie gesagt, das ist aber nicht unbedingt nötig.

Wir installieren die **hamlib** und **sox**. Dabei handelt es sich einerseits um eine Bibliothek, die es Applikationsprogrammen erlaubt verschiedene Rigs über eine einheitliche Schnittstelle anzusprechen und andererseits um ein vielseitiges Audio Werkzeug.

```
sudo apt install libhamlib-utils libhamlib-doc sox
```

Wir könnten den Transceiver damit über die CAT Schnittstelle steuern und insbesonders die PTT Steuerung vornehmen. In meinem Beispiel mache ich aber davon keinen Gebrauch sondern beschränke mich darauf den RTS Pin der seriellen Schnittstelle zu schalten.

Ein Hinweis scheint angebracht: Falls die Standard Pegellage des RTS Pins bei eurem Transceiver *ungünstig* liegt, so kann es sein, dass der Transceiver sofort in den Sendebetrieb geht sobald das Kabel angeschlossen wird. Schließt das Kabel deshalb erst an sobald die Software eingerichtet ist.

Generell ist es kein Fehler zunächst einmal mit einem Multimeter die Pegel zu überprüfen bevor man das Funkgerät einstöpselt.

Nun kann man das Kommando rictl benutzen um manuell die PTT Funktion zu testen. Bei mir lautet das:

```
rictl -p /dev/ttyUSB1 -P RTS
```

Welches die richtige serielle Schnittstelle ist müsst ihr zuvor herausfinden. Ich empfehle dazu sich einmal anzusehen welche Schnittstellen es im System gibt solange der USB Dongle nicht angeschlossen ist. Das geht mit

```
ls /dev/ttyUSB*
```

Dann schließt man den Dongle an und gibt das Kommand erneut. Die Schnittstelle die dazugekommen ist muss die Gesuchte sein. Es gibt auch noch andere Methoden die dann zum Einsatz kommen sollten wenn man seine Installation dauerhaft machen will. Dazu mehr bei der Einrichtung des *soundmodem*.

Nachdem also das rictl Programm gestartet wurde kann man von dessen Kommandozeile interaktiv PTT auf on oder off setzten. **T 1** sollte das RTS Signal einschalten und **T 0** aus. Wenn man die Funktion mit dem Multimeter validiert hat, so kann man den 6-pol mini DIN Stecker ans Funkgerät anstecken und den Versuch wiederholen. Der Transceiver sollte sich nun auftasten lassen.

Immer noch an der Kunstantenne, ihr habt doch bis jetzt alles an der Dummy-Load gemacht, so hoffe ich, schalten wir nun unseren Kontrollempfänger ein und stellen ihn auf die Frequenz unseres Transceivers ein. Dann geben wir das Kommando

```
AUDIODEV=plughw:Device,0 play -n synth sine 1000 gain 0
```

ein. *Device* ist dabei wieder der Name der Soundkarte, den wir schon bei der Inbetriebnahme der Empfangsseite verwendet haben. Der Wert *1000* steht für 1000Hz und der gain 0 bedeutet maximale Aussteuerung der Soundkarte. Es ist nun zu empfehlen den Trimmer, wo vorhanden, am Adapterkabel auf einen mittleren Wert zu stellen. Auch keine schlechte Idee ist es zunächst mit einem Kopfhörer zu überprüfen ob die 1000Hz am Ausgang der Soundkarte zu hören sind. Wenn alles klar ist so schließen wir das Kabel an den Transceiver an. Sobald wir die PTT mit Hilfe von ricgctl einschalten sollte das Signal nun auch im Kontrollempfänger zu hören sein. Man kann

nun vorsichtig den Trimmer größer drehen, bzw. die Lautstärke am Regler der Soundkarte. Man sollte dabei so vorgehen, dass das Signal möglichst groß wird, aber keines Falls zu stark, da dann störende Verzerrungen auftreten. Mein ICOM Receiver macht es mit einfach, da er ganz einfach aufhört zu senden, sobald die Aussteuerung (der Hub) zu groß wird. Ich gehe bis an diese Grenze heran und drehe den Trimmer dann um einen *Tick* kleiner.

Wer will kann an dieser Stelle mit Direwolf weiter machen, oder meinem Weg folgen und das Soundmodem von Thomas Sailer installieren, allerdings in einer von mir überarbeiteten und weitergeführten Version.

…wird fortgesetzt…

Linux und Schmalband Packet Radio mit Terminal: Unterschied zwischen den Versionen

Versionsgeschichte interaktiv durchsuchen
VisuellWikitext

Version vom 22. April 2022, 18:38 Uhr (Quelltext anzeigen)	**Aktuelle Version vom 24. April 2022, 12:37 Uhr (Quelltext anzeigen)**
Oe1rsa (Diskussion | Beiträge)	Oe1rsa (Diskussion | Beiträge)
(Neu angelegt, in Arbeit.)	(wieder ein Stück weiter ...)
Markierung: Visuelle Bearbeitung	Markierung: Visuelle Bearbeitung

(Eine dazwischenliegende Version desselben Benutzers wird nicht angezeigt)

Zeile 1:	**Zeile 1:**
[[Kategorie:Packet-Radio und I-Gate]]	[[Kategorie:Packet-Radio und I-Gate]]
[[Datei:Funkpaketpost.png|alternativtext=Funkpaketpost bzw. Packet Radio|zentriert|rahmenlos|529x529px]]	[[Datei:Funkpaketpost.png|alternativtext=Funkpaketpost bzw. Packet Radio|zentriert|rahmenlos|529x529px]]
	+ **==Linux und Schmalband Packet Radio mit Terminal==**
− In diesem Artikel versuche ich zu zeigen, wie man mit Linux **''Bordmitteln''** eine Verbindung zu einem Packet Radio Knoten aufbauen kann. Wir kommen damit zum Sendebetrieb. Die Beschreibung eines geeigneten Adapterkabels und den Empfang von Signalen habe ich im Artikel **[[**Linux und Amateur Packet Radio**]]** beschrieben. Ich werde in diesem Artikel auf Software für die man den '''WINE''' Emulator ([https://wwwwinehw.org 1]) benötigt bewusst verzichten. Nicht, dass mich jemand falsch versteht: '''WINE''' ist ein großartiges Projekt und es ist absolut nichts falsch daran es zu verwenden, speziell dann wenn es keine andere Lösung gibt. Wir wollen es eben mal ohne ''Alkohol'' versuchen, hi.	+ In diesem Artikel versuche ich zu zeigen, wie man mit Linux **“Bordmitteln”** eine Verbindung zu einem Packet Radio Knoten aufbauen kann. Wir kommen damit zum Sendebetrieb. Die Beschreibung eines geeigneten Adapterkabels und den Empfang von Signalen habe ich im Artikel **''** Linux und Amateur Packet Radio**''** beschrieben. Ich werde in diesem Artikel auf Software für die man den ''WINE'' Emulator ([https://wwwwinehw.org 1]) benötigt bewusst verzichten. Nicht, dass mich jemand falsch versteht: ''WINE'' ist ein großartiges Projekt und es ist absolut nichts falsch daran es zu verwenden, speziell dann wenn es keine andere Lösung gibt. Wir wollen es eben mal ohne ''Alkohol'' versuchen, hi.

Obwohl auch auf Kurzwelle nicht ganz unmöglich, so wird Schmalband Packet Radio normalerweise im 2m und 70cm Band über Transceiver gemacht, die nur FM Modulation beherrschen. Darauf konzentrieren wir uns auch.	Obwohl auch auf Kurzwelle nicht ganz unmöglich, so wird Schmalband Packet Radio normalerweise im 2m und 70cm Band über Transceiver gemacht, die nur FM Modulation beherrschen. Darauf konzentrieren wir uns auch.
− Sehen wir uns an, was wir benötigen, so stoßen wir darauf, dass das ein so genannter ''TNC'', ein '''T'''erminal '''N'''ode '''C'''ontroller ist. Die Aufgabe eines **TNC's** ist es Daten, die wir in paketierter Form übergeben, unter Kontrolle eines Protokolles, in unserem Fall ''AX.25'', an eine Gegenstelle zu übermitteln.	+ Sehen wir uns an, was wir benötigen, so stoßen wir darauf, dass das ein so genannter ''TNC'', ein '''T'''erminal '''N'''ode '''C'''ontroller ist. Die Aufgabe eines **TNC’s** ist es Daten, die wir in paketierter Form übergeben, unter Kontrolle eines Protokolles, in unserem Fall ''AX.25'', an eine Gegenstelle zu übermitteln.
− Solche **TNC's** waren ursprünglich, in den 80ern des 20. Jahrhunderts, reine Hardwarelösungen und deshalb auch recht unflexibel. Für die Abwicklung von Protokollen sind Computer prädestiniert und seitdem sie ausreichend hohe Geschwindigkeit haben können sie auch die Aufgaben der analogen Signalverarbeitung übernehmen, die in so einem TNC anfallen. Wer schon mit digitalen Übertragungsverfahren im Amateurfunk zu tun hatte weiß, dass oft eine Soundkarte ausreicht. So ist es auch in unserem Fall.	+ Solche **TNC’s** waren ursprünglich, in den 80ern des 20. Jahrhunderts, reine Hardwarelösungen und deshalb auch recht unflexibel. Für die Abwicklung von Protokollen sind Computer prädestiniert und seitdem sie ausreichend hohe Geschwindigkeit haben können sie auch die Aufgaben der analogen Signalverarbeitung übernehmen, die in so einem TNC anfallen. Wer schon mit digitalen Übertragungsverfahren im Amateurfunk zu tun hatte weiß, dass oft eine Soundkarte ausreicht. So ist es auch in unserem Fall.
Trotzdem geht es nicht ganz ohne Hardware. Eine Verbindung zwischen Rig, also unserem Transceiver, und dem Computer muss her. Diese Verbindung besteht aus zwei Teilen:	Trotzdem geht es nicht ganz ohne Hardware. Eine Verbindung zwischen Rig, also unserem Transceiver, und dem Computer muss her. Diese Verbindung besteht aus zwei Teilen:
Zeile 15:	**Zeile 16:**
#eine digitale Schnittstelle für die ''PTT'' Steuerung, also die Sendertastung.	#eine digitale Schnittstelle für die ''PTT'' Steuerung, also die Sendertastung.

Die Beschreibung eines geeigneten Kabels findet sich, wie bereits gesagt, in dem Artikel **[[**Linux und Amateur Packet Radio**]]** . Wir werden in diesem Abschnitt nun zunächst den Sendefall vorbereiten. Es ist hilfreich wenn euch zusätzlich zum Transceiver ein Kontrollempfänger zur Verfügung steht. Ein Empfänger der FM demodulieren kann ist ausreichend. Wer einen Empfänger hat der auch SSB fähig ist auf 2m oder 70cm, der kann darüberhinaus den exakten Hub seiner Aussendung einstellen. Das Verfahren dazu heisst *Besselnull* und ich werde es eventuell an anderer Stelle im Wiki beschreiben. Wie gesagt, das ist aber nicht unbedingt nötig.	Die Beschreibung eines geeigneten Kabels findet sich, wie bereits gesagt, in dem Artikel ''Linux und Amateur Packet Radio''. Wir werden in diesem Abschnitt nun zunächst den Sendefall vorbereiten. Es ist hilfreich wenn euch zusätzlich zum Transceiver ein Kontrollempfänger zur Verfügung steht. Ein Empfänger der FM demodulieren kann ist ausreichend. Wer einen Empfänger hat der auch SSB fähig ist auf 2m oder 70cm, der kann darüberhinaus den exakten Hub seiner Aussendung einstellen. Das Verfahren dazu heisst ''Besselnull'' und ich werde es eventuell an anderer Stelle im Wiki beschreiben. Wie gesagt, das ist aber nicht unbedingt nötig.
Wir installieren die '''hamlib''' und '''sox'''. Dabei handelt es sich einerseits um eine Bibliothek, die es Applikationsprogrammen erlaubt verschiedene Rigs über eine einheitliche Schnittstelle anzusprechen und andererseits um ein vielseitiges Audio Werkzeug.	Wir installieren die '''hamlib''' und '''sox'''. Dabei handelt es sich einerseits um eine Bibliothek, die es Applikationsprogrammen erlaubt verschiedene Rigs über eine einheitliche Schnittstelle anzusprechen und andererseits um ein vielseitiges Audio Werkzeug.
<pre>	<pre>**sudo apt install libhamlib-utils libhamlib-doc sox</pre>**
	Wir könnten den Transceiver damit über die CAT Schnittstelle steuern und insbesonders die PTT Steuerung vornehmen. In meinem Beispiel mache ich aber davon keinen Gebrauch sondern beschränke mich darauf den RTS Pin der seriellen Schnittstelle zu schalten.
	Ein Hinweis scheint angebracht: Falls die Standard Pegellage des RTS Pins bei eurem Transceiver ''ungünstig''

	+ liegt, so kann es sein, dass der Transceiver sofort in den Sendebetrieb geht sobald das Kabel angeschlossen wird. Schließt das Kabel deshalb erst an sobald die Software eingerichtet ist.
	+
	+ Generell ist es kein Fehler zunächst einmal mit einem Multimeter die Pegel zu überprüfen bevor man das Funkgerät einstöpselt.
	+
	+ Nun kann man das Kommando rictl benutzen um manuell die PTT Funktion zu testen. Bei mir lautet das:
	+
	+ <pre>rictl -p /dev/ttyUSB1 -P RTS< /pre>
	+ Welches die richtige serielle Schnittstelle ist müsst ihr zuvor herausfinden. Ich empfehle dazu sich einmal anzusehen welche Schnittstellen es im System gibt solange der USB Dongle nicht angeschlossen ist. Das geht mit
	+
	+ <pre>ls /dev/ttyUSB*</pre>
	+ Dann schließt man den Dongle an und gibt das Kommand erneut. Die Schnittstelle die dazugekommen ist muss die Gesuchte sein. Es gibt auch noch andere Methoden die dann zum Einsatz kommen sollten wenn man seine Installation dauerhaft machen will. Dazu mehr bei der Einrichtung des ''soundmodem''.
sudo apt install libhamlib-utils libhamlib-doc sox	Nachdem also das rictl Programm gestartet wurde kann man von dessen Kommandozeile interaktiv PTT

–	+ auf on oder off setzten. '''T 1''' sollte das RTS Signal einschalten und '''T 0''' aus. Wenn man die Funktion mit dem Multimeter validiert hat, so kann man den 6-pol mini DIN Stecker ans Funkgerät anstecken und den Versuch wiederholen. Der Transceiver sollte sich nun auftasten lassen.
– <\pre>	
– Wir könnten den Transceiver damit über die CAT Schnittstelle steuern und insbesonders die PTT Steuerung vornehmen. In meinem Beispiel mache ich aber davon keinen Gebrauch sondern beschränke mich darauf den RTS Pin der seriellen Schnittstelle zu schalten.	+ Immer noch an der Kunstantenne, ihr habt doch bis jetzt alles an der Dummy-Load gemacht, so hoffe ich, schalten wir nun unseren Kontrollempfänger ein und stellen ihn auf die Frequenz unseres Transceivers ein. Dann geben wir das Kommando
	+
	+ <pre>AUDIODEV=plughw:Device,0 play -n synth sine 1000 gain 0</pre>
	+ ein. ''Device'' ist dabei wieder der Name der Soundkarte, den wir schon bei der Inbetriebnahme der Empfangsseite verwendet haben. Der Wert ''1000'' steht für 1000Hz und der gain 0 bedeutet maximale Aussteuerung der Soundkarte. Es ist nun zu empfehlen den Trimmer, wo vorhanden, am Adapterkabel auf einen mittleren Wert zu stellen. Auch keine schlechte Idee ist es zunächst mit einem Kopfhörer zu überprüfen ob die 1000Hz am Ausgang der Soundkarte zu hören sind. Wenn alles klar ist so schließen wir das Kabel an den Transceiver an. Sobald wir die PTT mit Hilfe von ricgctl einschalten sollte das Signal nun auch im Kontrollempfänger zu hören sein. Man kann nun vorsichtig den Trimmer größer drehen, bzw. die Lautstärke am Regler der Soundkarte. Man sollte

dabei so vorgehen, dass das Signal möglichst groß wird, aber **keines Falls zu stark, da dann störende Verzerrungen auftreten. Mein ICOM Receiver macht es mit einfach, da er ganz einfach aufhört zu senden, sobald die Aussteuerung (**der **Hub)** zu **groß wird. Ich gehe bis an diese Grenze heran und drehe den Trimmer dann um einen ''Tick'' kleiner.**

+

+ **Wer will kann an dieser Stelle mit Direwolf weiter machen, oder meinem Weg folgen und das Soundmodem von Thomas Sailer installieren, allerdings in einer von mir überarbeiteten und weitergeführten Version**.

+

+ **'''''...wird fortgesetzt...'''''**

Aktuelle Version vom 24. April 2022, 12:37 Uhr

Linux und Schmalband Packet Radio mit Terminal

In diesem Artikel versuche ich zu zeigen, wie man mit Linux “Bordmitteln” eine Verbindung zu einem Packet Radio Knoten aufbauen kann. Wir kommen damit zum Sendebetrieb. Die Beschreibung eines geeigneten Adapterkabels und den Empfang von Signalen habe ich im Artikel *Linux und Amateur Packet Radio* beschrieben. Ich werde in diesem Artikel auf Software für die man den *WINE* Emulator (1) benötigt bewusst verzichten. Nicht, dass mich jemand falsch versteht: *WINE* ist ein großartiges Projekt und es ist absolut nichts falsch daran es zu verwenden, speziell dann wenn es keine andere Lösung gibt. Wir wollen es eben mal ohne *Alkohol* versuchen, hi.

Obwohl auch auf Kurzwelle nicht ganz unmöglich, so wird Schmalband Packet Radio normalerweise im 2m und 70cm Band über Transceiver gemacht, die nur FM Modulation beherrschen. Darauf konzentrieren wir uns auch.

Sehen wir uns an, was wir benötigen, so stoßen wir darauf, dass das ein so genannter *TNC*, ein **T** erminal **N**ode **C**ontroller ist. Die Aufgabe eines TNC’s ist es Daten, die wir in paketierter Form übergeben, unter Kontrolle eines Protokolles, in unserem Fall *AX.25*, an eine Gegenstelle zu übermitteln.

Solche TNC’s waren ursprünglich, in den 80ern des 20. Jahrhunderts, reine Hardwarelösungen und deshalb auch recht unflexibel. Für die Abwicklung von Protokollen sind Computer prädestiniert und seitdem sie ausreichend hohe Geschwindigkeit haben können sie auch die Aufgaben der analogen Signalverarbeitung übernehmen, die in so einem TNC anfallen. Wer schon mit digitalen Übertragungsverfahren im Amateurfunk zu tun hatte weiß, dass oft eine Soundkarte ausreicht. So ist es auch in unserem Fall.

Trotzdem geht es nicht ganz ohne Hardware. Eine Verbindung zwischen Rig, also unserem Transceiver, und dem Computer muss her. Diese Verbindung besteht aus zwei Teilen:

1. Eine analoge Schnittstelle für Audio Übertragung und
2. eine digitale Schnittstelle für die *PTT* Steuerung, also die Sendertastung.

Die Beschreibung eines geeigneten Kabels findet sich, wie bereits gesagt, in dem Artikel *Linux und Amateur Packet Radio*. Wir werden in diesem Abschnitt nun zunächst den Sendefall vorbereiten. Es ist hilfreich wenn euch zusätzlich zum Transceiver ein Kontrollempfänger zur Verfügung steht. Ein Empfänger der FM demodulieren kann ist ausreichend. Wer einen Empfänger hat der auch SSB fähig ist auf 2m oder 70cm, der kann darüberhinaus den exakten Hub seiner Aussendung einstellen. Das Verfahren dazu heisst *Besselnull* und ich werde es eventuell an anderer Stelle im Wiki beschreiben. Wie gesagt, das ist aber nicht unbedingt nötig.

Wir installieren die **hamlib** und **sox**. Dabei handelt es sich einerseits um eine Bibliothek, die es Applikationsprogrammen erlaubt verschiedene Rigs über eine einheitliche Schnittstelle anzusprechen und andererseits um ein vielseitiges Audio Werkzeug.

```
sudo apt install libhamlib-utils libhamlib-doc sox
```

Wir könnten den Transceiver damit über die CAT Schnittstelle steuern und insbesonders die PTT Steuerung vornehmen. In meinem Beispiel mache ich aber davon keinen Gebrauch sondern beschränke mich darauf den RTS Pin der seriellen Schnittstelle zu schalten.

Ein Hinweis scheint angebracht: Falls die Standard Pegellage des RTS Pins bei eurem Transceiver *ungünstig* liegt, so kann es sein, dass der Transceiver sofort in den Sendebetrieb geht sobald das Kabel angeschlossen wird. Schließt das Kabel deshalb erst an sobald die Software eingerichtet ist.

Generell ist es kein Fehler zunächst einmal mit einem Multimeter die Pegel zu überprüfen bevor man das Funkgerät einstöpselt.

Nun kann man das Kommando rictl benutzen um manuell die PTT Funktion zu testen. Bei mir lautet das:

```
rictl -p /dev/ttyUSB1 -P RTS
```

Welches die richtige serielle Schnittstelle ist müsst ihr zuvor herausfinden. Ich empfehle dazu sich einmal anzusehen welche Schnittstellen es im System gibt solange der USB Dongle nicht angeschlossen ist. Das geht mit

```
ls /dev/ttyUSB*
```

Dann schließt man den Dongle an und gibt das Kommand erneut. Die Schnittstelle die dazugekommen ist muss die Gesuchte sein. Es gibt auch noch andere Methoden die dann zum Einsatz kommen sollten wenn man seine Installation dauerhaft machen will. Dazu mehr bei der Einrichtung des *soundmodem*.

Nachdem also das rictl Programm gestartet wurde kann man von dessen Kommandozeile interaktiv PTT auf on oder off setzten. **T 1** sollte das RTS Signal einschalten und **T 0** aus. Wenn man die Funktion mit dem Multimeter validiert hat, so kann man den 6-pol mini DIN Stecker ans Funkgerät anstecken und den Versuch wiederholen. Der Transceiver sollte sich nun auftasten lassen.

Immer noch an der Kunstantenne, ihr habt doch bis jetzt alles an der Dummy-Load gemacht, so hoffe ich, schalten wir nun unseren Kontrollempfänger ein und stellen ihn auf die Frequenz unseres Transceivers ein. Dann geben wir das Kommando

```
AUDIODEV=plughw:Device,0 play -n synth sine 1000 gain 0
```

ein. *Device* ist dabei wieder der Name der Soundkarte, den wir schon bei der Inbetriebnahme der Empfangsseite verwendet haben. Der Wert *1000* steht für 1000Hz und der gain 0 bedeutet maximale Aussteuerung der Soundkarte. Es ist nun zu empfehlen den Trimmer, wo vorhanden, am Adapterkabel auf einen mittleren Wert zu stellen. Auch keine schlechte Idee ist es zunächst mit einem Kopfhörer zu überprüfen ob die 1000Hz am Ausgang der Soundkarte zu hören sind. Wenn alles klar ist so schließen wir das Kabel an den Transceiver an. Sobald wir die PTT mit Hilfe von ricgctl einschalten sollte das Signal nun auch im Kontrollempfänger zu hören sein. Man kann

nun vorsichtig den Trimmer größer drehen, bzw. die Lautstärke am Regler der Soundkarte. Man sollte dabei so vorgehen, dass das Signal möglichst groß wird, aber keines Falls zu stark, da dann störende Verzerrungen auftreten. Mein ICOM Receiver macht es mit einfach, da er ganz einfach aufhört zu senden, sobald die Aussteuerung (der Hub) zu groß wird. Ich gehe bis an diese Grenze heran und drehe den Trimmer dann um einen *Tick* kleiner.

Wer will kann an dieser Stelle mit Direwolf weiter machen, oder meinem Weg folgen und das Soundmodem von Thomas Sailer installieren, allerdings in einer von mir überarbeiteten und weitergeführten Version.

…wird fortgesetzt…

Linux und Schmalband Packet Radio mit Terminal: Unterschied zwischen den Versionen

Versionsgeschichte interaktiv durchsuchen
VisuellWikitext

Version vom 22. April 2022, 18:38 Uhr (Quelltext anzeigen)	**Aktuelle Version vom 24. April 2022, 12:37 Uhr (Quelltext anzeigen)**
Oe1rsa (Diskussion | Beiträge)	Oe1rsa (Diskussion | Beiträge)
(Neu angelegt, in Arbeit.)	(wieder ein Stück weiter ...)
Markierung: Visuelle Bearbeitung	Markierung: Visuelle Bearbeitung

(Eine dazwischenliegende Version desselben Benutzers wird nicht angezeigt)

Zeile 1:	**Zeile 1:**
[[Kategorie:Packet-Radio und I-Gate]]	[[Kategorie:Packet-Radio und I-Gate]]
[[Datei:Funkpaketpost.png|alternativtext=Funkpaketpost bzw. Packet Radio|zentriert|rahmenlos|529x529px]]	[[Datei:Funkpaketpost.png|alternativtext=Funkpaketpost bzw. Packet Radio|zentriert|rahmenlos|529x529px]]
	+ **==Linux und Schmalband Packet Radio mit Terminal==**
− In diesem Artikel versuche ich zu zeigen, wie man mit Linux **''Bordmitteln''** eine Verbindung zu einem Packet Radio Knoten aufbauen kann. Wir kommen damit zum Sendebetrieb. Die Beschreibung eines geeigneten Adapterkabels und den Empfang von Signalen habe ich im Artikel **[[**Linux und Amateur Packet Radio**]]** beschrieben. Ich werde in diesem Artikel auf Software für die man den '''WINE''' Emulator ([https://wwwwinehw.org 1]) benötigt bewusst verzichten. Nicht, dass mich jemand falsch versteht: '''WINE''' ist ein großartiges Projekt und es ist absolut nichts falsch daran es zu verwenden, speziell dann wenn es keine andere Lösung gibt. Wir wollen es eben mal ohne ''Alkohol'' versuchen, hi.	+ In diesem Artikel versuche ich zu zeigen, wie man mit Linux **“Bordmitteln”** eine Verbindung zu einem Packet Radio Knoten aufbauen kann. Wir kommen damit zum Sendebetrieb. Die Beschreibung eines geeigneten Adapterkabels und den Empfang von Signalen habe ich im Artikel **''** Linux und Amateur Packet Radio**''** beschrieben. Ich werde in diesem Artikel auf Software für die man den ''WINE'' Emulator ([https://wwwwinehw.org 1]) benötigt bewusst verzichten. Nicht, dass mich jemand falsch versteht: ''WINE'' ist ein großartiges Projekt und es ist absolut nichts falsch daran es zu verwenden, speziell dann wenn es keine andere Lösung gibt. Wir wollen es eben mal ohne ''Alkohol'' versuchen, hi.

Obwohl auch auf Kurzwelle nicht ganz unmöglich, so wird Schmalband Packet Radio normalerweise im 2m und 70cm Band über Transceiver gemacht, die nur FM Modulation beherrschen. Darauf konzentrieren wir uns auch.	Obwohl auch auf Kurzwelle nicht ganz unmöglich, so wird Schmalband Packet Radio normalerweise im 2m und 70cm Band über Transceiver gemacht, die nur FM Modulation beherrschen. Darauf konzentrieren wir uns auch.
− Sehen wir uns an, was wir benötigen, so stoßen wir darauf, dass das ein so genannter ''TNC'', ein '''T'''erminal '''N'''ode '''C'''ontroller ist. Die Aufgabe eines **TNC's** ist es Daten, die wir in paketierter Form übergeben, unter Kontrolle eines Protokolles, in unserem Fall ''AX.25'', an eine Gegenstelle zu übermitteln.	+ Sehen wir uns an, was wir benötigen, so stoßen wir darauf, dass das ein so genannter ''TNC'', ein '''T'''erminal '''N'''ode '''C'''ontroller ist. Die Aufgabe eines **TNC’s** ist es Daten, die wir in paketierter Form übergeben, unter Kontrolle eines Protokolles, in unserem Fall ''AX.25'', an eine Gegenstelle zu übermitteln.
− Solche **TNC's** waren ursprünglich, in den 80ern des 20. Jahrhunderts, reine Hardwarelösungen und deshalb auch recht unflexibel. Für die Abwicklung von Protokollen sind Computer prädestiniert und seitdem sie ausreichend hohe Geschwindigkeit haben können sie auch die Aufgaben der analogen Signalverarbeitung übernehmen, die in so einem TNC anfallen. Wer schon mit digitalen Übertragungsverfahren im Amateurfunk zu tun hatte weiß, dass oft eine Soundkarte ausreicht. So ist es auch in unserem Fall.	+ Solche **TNC’s** waren ursprünglich, in den 80ern des 20. Jahrhunderts, reine Hardwarelösungen und deshalb auch recht unflexibel. Für die Abwicklung von Protokollen sind Computer prädestiniert und seitdem sie ausreichend hohe Geschwindigkeit haben können sie auch die Aufgaben der analogen Signalverarbeitung übernehmen, die in so einem TNC anfallen. Wer schon mit digitalen Übertragungsverfahren im Amateurfunk zu tun hatte weiß, dass oft eine Soundkarte ausreicht. So ist es auch in unserem Fall.
Trotzdem geht es nicht ganz ohne Hardware. Eine Verbindung zwischen Rig, also unserem Transceiver, und dem Computer muss her. Diese Verbindung besteht aus zwei Teilen:	Trotzdem geht es nicht ganz ohne Hardware. Eine Verbindung zwischen Rig, also unserem Transceiver, und dem Computer muss her. Diese Verbindung besteht aus zwei Teilen:
Zeile 15:	**Zeile 16:**
#eine digitale Schnittstelle für die ''PTT'' Steuerung, also die Sendertastung.	#eine digitale Schnittstelle für die ''PTT'' Steuerung, also die Sendertastung.

Die Beschreibung eines geeigneten Kabels findet sich, wie bereits gesagt, in dem Artikel **[[**Linux und Amateur Packet Radio**]]** . Wir werden in diesem Abschnitt nun zunächst den Sendefall vorbereiten. Es ist hilfreich wenn euch zusätzlich zum Transceiver ein Kontrollempfänger zur Verfügung steht. Ein Empfänger der FM demodulieren kann ist ausreichend. Wer einen Empfänger hat der auch SSB fähig ist auf 2m oder 70cm, der kann darüberhinaus den exakten Hub seiner Aussendung einstellen. Das Verfahren dazu heisst *Besselnull* und ich werde es eventuell an anderer Stelle im Wiki beschreiben. Wie gesagt, das ist aber nicht unbedingt nötig.	Die Beschreibung eines geeigneten Kabels findet sich, wie bereits gesagt, in dem Artikel ''Linux und Amateur Packet Radio''. Wir werden in diesem Abschnitt nun zunächst den Sendefall vorbereiten. Es ist hilfreich wenn euch zusätzlich zum Transceiver ein Kontrollempfänger zur Verfügung steht. Ein Empfänger der FM demodulieren kann ist ausreichend. Wer einen Empfänger hat der auch SSB fähig ist auf 2m oder 70cm, der kann darüberhinaus den exakten Hub seiner Aussendung einstellen. Das Verfahren dazu heisst ''Besselnull'' und ich werde es eventuell an anderer Stelle im Wiki beschreiben. Wie gesagt, das ist aber nicht unbedingt nötig.
Wir installieren die '''hamlib''' und '''sox'''. Dabei handelt es sich einerseits um eine Bibliothek, die es Applikationsprogrammen erlaubt verschiedene Rigs über eine einheitliche Schnittstelle anzusprechen und andererseits um ein vielseitiges Audio Werkzeug.	Wir installieren die '''hamlib''' und '''sox'''. Dabei handelt es sich einerseits um eine Bibliothek, die es Applikationsprogrammen erlaubt verschiedene Rigs über eine einheitliche Schnittstelle anzusprechen und andererseits um ein vielseitiges Audio Werkzeug.
<pre>	<pre>**sudo apt install libhamlib-utils libhamlib-doc sox</pre>**
	Wir könnten den Transceiver damit über die CAT Schnittstelle steuern und insbesonders die PTT Steuerung vornehmen. In meinem Beispiel mache ich aber davon keinen Gebrauch sondern beschränke mich darauf den RTS Pin der seriellen Schnittstelle zu schalten.
	Ein Hinweis scheint angebracht: Falls die Standard Pegellage des RTS Pins bei eurem Transceiver ''ungünstig''

	+ liegt, so kann es sein, dass der Transceiver sofort in den Sendebetrieb geht sobald das Kabel angeschlossen wird. Schließt das Kabel deshalb erst an sobald die Software eingerichtet ist.
	+
	+ Generell ist es kein Fehler zunächst einmal mit einem Multimeter die Pegel zu überprüfen bevor man das Funkgerät einstöpselt.
	+
	+ Nun kann man das Kommando rictl benutzen um manuell die PTT Funktion zu testen. Bei mir lautet das:
	+
	+ <pre>rictl -p /dev/ttyUSB1 -P RTS</pre>
	+ Welches die richtige serielle Schnittstelle ist müsst ihr zuvor herausfinden. Ich empfehle dazu sich einmal anzusehen welche Schnittstellen es im System gibt solange der USB Dongle nicht angeschlossen ist. Das geht mit
	+
	+ <pre>ls /dev/ttyUSB*</pre>
	+ Dann schließt man den Dongle an und gibt das Kommand erneut. Die Schnittstelle die dazugekommen ist muss die Gesuchte sein. Es gibt auch noch andere Methoden die dann zum Einsatz kommen sollten wenn man seine Installation dauerhaft machen will. Dazu mehr bei der Einrichtung des ''soundmodem''.
sudo apt install libhamlib-utils libhamlib-doc sox	Nachdem also das rictl Programm gestartet wurde kann man von dessen Kommandozeile interaktiv PTT

–	+ auf on oder off setzten. '''T 1''' sollte das RTS Signal einschalten und '''T 0''' aus. Wenn man die Funktion mit dem Multimeter validiert hat, so kann man den 6-pol mini DIN Stecker ans Funkgerät anstecken und den Versuch wiederholen. Der Transceiver sollte sich nun auftasten lassen.
– <\pre>	
– Wir könnten den Transceiver damit über die CAT Schnittstelle steuern und insbesonders die PTT Steuerung vornehmen. In meinem Beispiel mache ich aber davon keinen Gebrauch sondern beschränke mich darauf den RTS Pin der seriellen Schnittstelle zu schalten.	+ Immer noch an der Kunstantenne, ihr habt doch bis jetzt alles an der Dummy-Load gemacht, so hoffe ich, schalten wir nun unseren Kontrollempfänger ein und stellen ihn auf die Frequenz unseres Transceivers ein. Dann geben wir das Kommando
	+
	+ <pre>AUDIODEV=plughw:Device,0 play -n synth sine 1000 gain 0</pre>
	+ ein. ''Device'' ist dabei wieder der Name der Soundkarte, den wir schon bei der Inbetriebnahme der Empfangsseite verwendet haben. Der Wert ''1000'' steht für 1000Hz und der gain 0 bedeutet maximale Aussteuerung der Soundkarte. Es ist nun zu empfehlen den Trimmer, wo vorhanden, am Adapterkabel auf einen mittleren Wert zu stellen. Auch keine schlechte Idee ist es zunächst mit einem Kopfhörer zu überprüfen ob die 1000Hz am Ausgang der Soundkarte zu hören sind. Wenn alles klar ist so schließen wir das Kabel an den Transceiver an. Sobald wir die PTT mit Hilfe von ricgctl einschalten sollte das Signal nun auch im Kontrollempfänger zu hören sein. Man kann nun vorsichtig den Trimmer größer drehen, bzw. die Lautstärke am Regler der Soundkarte. Man sollte

dabei so vorgehen, dass das Signal möglichst groß wird, aber **keines Falls zu stark, da dann störende Verzerrungen auftreten. Mein ICOM Receiver macht es mit einfach, da er ganz einfach aufhört zu senden, sobald die Aussteuerung (**der **Hub)** zu **groß wird. Ich gehe bis an diese Grenze heran und drehe den Trimmer dann um einen ''Tick'' kleiner.**

+

+ **Wer will kann an dieser Stelle mit Direwolf weiter machen, oder meinem Weg folgen und das Soundmodem von Thomas Sailer installieren, allerdings in einer von mir überarbeiteten und weitergeführten Version**.

+

+ **'''''...wird fortgesetzt...'''''**

Aktuelle Version vom 24. April 2022, 12:37 Uhr

Linux und Schmalband Packet Radio mit Terminal

In diesem Artikel versuche ich zu zeigen, wie man mit Linux “Bordmitteln” eine Verbindung zu einem Packet Radio Knoten aufbauen kann. Wir kommen damit zum Sendebetrieb. Die Beschreibung eines geeigneten Adapterkabels und den Empfang von Signalen habe ich im Artikel *Linux und Amateur Packet Radio* beschrieben. Ich werde in diesem Artikel auf Software für die man den *WINE* Emulator (1) benötigt bewusst verzichten. Nicht, dass mich jemand falsch versteht: *WINE* ist ein großartiges Projekt und es ist absolut nichts falsch daran es zu verwenden, speziell dann wenn es keine andere Lösung gibt. Wir wollen es eben mal ohne *Alkohol* versuchen, hi.

Obwohl auch auf Kurzwelle nicht ganz unmöglich, so wird Schmalband Packet Radio normalerweise im 2m und 70cm Band über Transceiver gemacht, die nur FM Modulation beherrschen. Darauf konzentrieren wir uns auch.

Sehen wir uns an, was wir benötigen, so stoßen wir darauf, dass das ein so genannter *TNC*, ein **T** erminal **N**ode **C**ontroller ist. Die Aufgabe eines TNC’s ist es Daten, die wir in paketierter Form übergeben, unter Kontrolle eines Protokolles, in unserem Fall *AX.25*, an eine Gegenstelle zu übermitteln.

Solche TNC’s waren ursprünglich, in den 80ern des 20. Jahrhunderts, reine Hardwarelösungen und deshalb auch recht unflexibel. Für die Abwicklung von Protokollen sind Computer prädestiniert und seitdem sie ausreichend hohe Geschwindigkeit haben können sie auch die Aufgaben der analogen Signalverarbeitung übernehmen, die in so einem TNC anfallen. Wer schon mit digitalen Übertragungsverfahren im Amateurfunk zu tun hatte weiß, dass oft eine Soundkarte ausreicht. So ist es auch in unserem Fall.

Trotzdem geht es nicht ganz ohne Hardware. Eine Verbindung zwischen Rig, also unserem Transceiver, und dem Computer muss her. Diese Verbindung besteht aus zwei Teilen:

1. Eine analoge Schnittstelle für Audio Übertragung und
2. eine digitale Schnittstelle für die *PTT* Steuerung, also die Sendertastung.

Die Beschreibung eines geeigneten Kabels findet sich, wie bereits gesagt, in dem Artikel *Linux und Amateur Packet Radio*. Wir werden in diesem Abschnitt nun zunächst den Sendefall vorbereiten. Es ist hilfreich wenn euch zusätzlich zum Transceiver ein Kontrollempfänger zur Verfügung steht. Ein Empfänger der FM demodulieren kann ist ausreichend. Wer einen Empfänger hat der auch SSB fähig ist auf 2m oder 70cm, der kann darüberhinaus den exakten Hub seiner Aussendung einstellen. Das Verfahren dazu heisst *Besselnull* und ich werde es eventuell an anderer Stelle im Wiki beschreiben. Wie gesagt, das ist aber nicht unbedingt nötig.

Wir installieren die **hamlib** und **sox**. Dabei handelt es sich einerseits um eine Bibliothek, die es Applikationsprogrammen erlaubt verschiedene Rigs über eine einheitliche Schnittstelle anzusprechen und andererseits um ein vielseitiges Audio Werkzeug.

```
sudo apt install libhamlib-utils libhamlib-doc sox
```

Wir könnten den Transceiver damit über die CAT Schnittstelle steuern und insbesonders die PTT Steuerung vornehmen. In meinem Beispiel mache ich aber davon keinen Gebrauch sondern beschränke mich darauf den RTS Pin der seriellen Schnittstelle zu schalten.

Ein Hinweis scheint angebracht: Falls die Standard Pegellage des RTS Pins bei eurem Transceiver *ungünstig* liegt, so kann es sein, dass der Transceiver sofort in den Sendebetrieb geht sobald das Kabel angeschlossen wird. Schließt das Kabel deshalb erst an sobald die Software eingerichtet ist.

Generell ist es kein Fehler zunächst einmal mit einem Multimeter die Pegel zu überprüfen bevor man das Funkgerät einstöpselt.

Nun kann man das Kommando rictl benutzen um manuell die PTT Funktion zu testen. Bei mir lautet das:

```
rictl -p /dev/ttyUSB1 -P RTS
```

Welches die richtige serielle Schnittstelle ist müsst ihr zuvor herausfinden. Ich empfehle dazu sich einmal anzusehen welche Schnittstellen es im System gibt solange der USB Dongle nicht angeschlossen ist. Das geht mit

```
ls /dev/ttyUSB*
```

Dann schließt man den Dongle an und gibt das Kommand erneut. Die Schnittstelle die dazugekommen ist muss die Gesuchte sein. Es gibt auch noch andere Methoden die dann zum Einsatz kommen sollten wenn man seine Installation dauerhaft machen will. Dazu mehr bei der Einrichtung des *soundmodem*.

Nachdem also das rictl Programm gestartet wurde kann man von dessen Kommandozeile interaktiv PTT auf on oder off setzten. **T 1** sollte das RTS Signal einschalten und **T 0** aus. Wenn man die Funktion mit dem Multimeter validiert hat, so kann man den 6-pol mini DIN Stecker ans Funkgerät anstecken und den Versuch wiederholen. Der Transceiver sollte sich nun auftasten lassen.

Immer noch an der Kunstantenne, ihr habt doch bis jetzt alles an der Dummy-Load gemacht, so hoffe ich, schalten wir nun unseren Kontrollempfänger ein und stellen ihn auf die Frequenz unseres Transceivers ein. Dann geben wir das Kommando

```
AUDIODEV=plughw:Device,0 play -n synth sine 1000 gain 0
```

ein. *Device* ist dabei wieder der Name der Soundkarte, den wir schon bei der Inbetriebnahme der Empfangsseite verwendet haben. Der Wert *1000* steht für 1000Hz und der gain 0 bedeutet maximale Aussteuerung der Soundkarte. Es ist nun zu empfehlen den Trimmer, wo vorhanden, am Adapterkabel auf einen mittleren Wert zu stellen. Auch keine schlechte Idee ist es zunächst mit einem Kopfhörer zu überprüfen ob die 1000Hz am Ausgang der Soundkarte zu hören sind. Wenn alles klar ist so schließen wir das Kabel an den Transceiver an. Sobald wir die PTT mit Hilfe von ricgctl einschalten sollte das Signal nun auch im Kontrollempfänger zu hören sein. Man kann

nun vorsichtig den Trimmer größer drehen, bzw. die Lautstärke am Regler der Soundkarte. Man sollte dabei so vorgehen, dass das Signal möglichst groß wird, aber keines Falls zu stark, da dann störende Verzerrungen auftreten. Mein ICOM Receiver macht es mit einfach, da er ganz einfach aufhört zu senden, sobald die Aussteuerung (der Hub) zu groß wird. Ich gehe bis an diese Grenze heran und drehe den Trimmer dann um einen *Tick* kleiner.

Wer will kann an dieser Stelle mit Direwolf weiter machen, oder meinem Weg folgen und das Soundmodem von Thomas Sailer installieren, allerdings in einer von mir überarbeiteten und weitergeführten Version.

…wird fortgesetzt…