

Anleitung zur Installation von Zusatzfunktionen für SvxDink

Stand 07.01.2023
Adi Bier / DL1HRC

1. Einleitung
2. Vorbereitungen
 - 2.1. Hardware
 - 2.2. Software
 - 2.2.1. Installation der Entwicklungsumgebung
 - 2.2.2. Installation des pjSip-Frameworks
 - 2.2.3. Installation einer Branch-Version von SvxDink
 - 2.2.4. Kompilierung und Installation eines Branches oder experimentellen Erweiterung
 - 2.2.5. Installation der Sprachumgebung
 - 2.2.6. Konfiguration von SvxDink
 - 2.2.7. Aktualisierung eines Branches
3. SIP-Erweiterungen
 - 3.1. Experimentelle Erweiterung pjSipLogic
 - 3.2. Asterisk als zentraler Telefonieserver für SvxDink und Hamradio
 - 3.3. Sicherheitshinweise
 - 3.4. Konfiguration einer Fritz!Box als Netzzugang zum HAMRADIO-Sip-Netzwerk
 - 3.5. Konfiguration von Volp-Clients
 - 3.5.1. Konfiguration ZoiPer
 - 3.5.2. Konfiguration Twinkle
 - 3.5.3. Konfiguration Sipnatic
 - 3.6. SipLogic über Alsa-Loopback
 - 3.7. Anbindung eines Hytera-Repeater RDxxx an das SIP-Netzwerk
 - 3.8. Einrichtung eines Userzuganges
4. USRP-Erweiterung (Branch svxdink-usrp)
 - 4.1. Installation der MMDVM-Umgebung
 - 4.1.1. Analog_Bridge
 - 4.1.2. MMDVM_Bridge
 - 4.1.3. AMBEServer
 - 4.2. Konfiguration der MMDVM-Umgebung
 - 4.2.1. Analog_Bridge
 - 4.2.1.1. ThumbDV 3000-USB-Stick
 - 4.2.1.2. md380-emu
 - 4.2.1.3. AMBEServer und ThumbDV 3000-USB-Stick
 - 4.2.2. MMDVM_Bridge
 - 4.3. SvxDink-Abschnitt [UsrpLogic]
5. Tetra-Erweiterungen (Branch tetra-contrib)
 - 5.1. Vorbereitungen
 - 5.2. Konfiguration
 - 5.2.1. PEI-Hardware
 - 5.2.2. Konfiguration der PEI in der CPS
 - 5.2.3. Test der PEI-Kommunikation
 - 5.3. SvxDink-Abschnitt [TetraLogic]
 - 5.4. Konfiguration der Tetra-Mobilgeräte
 - 5.4.1. ISSI
 - 5.4.2. LocationInfo

- 5.4.2.1. Positionsmeldung periodisch senden
 - 5.4.2.2. Positionsmeldung senden bei DMO=on
 - 5.4.2.3. Positionsmeldung bei Low Battery
 - 5.4.2.4. Positionsmeldung bei Gefahr
 - 5.4.2.5. Positionsmeldung beim Senden einer Status-SDS
 - 5.4.2.6. Positionsmeldung beim Ausschalten des HRT/MRT's
- 5.5. Branch tetra-sip
- 6. Installation und Konfiguration weiterer Branche
 - 6.1. hotspot-nmea-Branch
 - 6.2. AnnounceLogic-Branch
- 7. Installation von Zusatzmodulen
 - 7.1. PropagationMonitor
 - 7.2. KatWarn
 - 7.3. Unwetterwarnungen vom Deutschen Wetterdienst
 - 7.4. Announcement
- 8. Installation SvxBPortal 2.5 (beta)
 - 8.1. Systemanforderungen
 - 8.2. Installation des MySQL-Datenbankservers
 - 8.3. Installation des Apache2-Webservers
 - 8.4. Erstellen der MySQL-Datenbank
 - 8.5. Erste Einstellungen im SvxBPortal
 - 8.6. Administrator Interface
 - 8.7. Sprachen
 - 8.8. Tips
- 9. Sonstiges
 - 9.1. Hinweise für Entwickler
 - 9.2. Hilfe erhalten
 - 9.3. Danksagung
- 10. Quellenangaben

1. Einleitung

SvxLink bietet von Hause aus einen sehr großen Funktionsumfang. Neben der Standardinstallation, die von Tobias/SM0SVX angeboten wird, habe ich eine ganze Reihe von Erweiterungen programmiert, die für den ein oder anderen Funkamateurliebhaber eventuell interessant sein könnten. Ich stelle meine Erweiterung unter github.com bzw. gitlab.com als Repo **ohne jegliche Garantie** zum Download zur Verfügung. Die Installation und der Betrieb erfolgt auf eigenes Risiko. Bei Problemen und Nachfragen kann ich im Rahmen meiner Zeit helfen, Voraussetzung ist, dass Logfiles und Konfigurationsfiles zur Verfügung gestellt werden.

Bitte bedenken, dass die Installation der Erweiterungen einige Linuxkenntnisse voraussetzt, Linux-Neulingen sei davon abgeraten oder sie sollten sich um entsprechende Unterstützung bemühen.

Experimentelle Erweiterung „pjSipLogic“

Die experimentelle pjSipLogic-Erweiterung ermöglicht das Verbinden des eigenen Nodes mit einem Sip-Server (Fritz!Box, Asterisk,...) und damit das „Anrufen“ eines Relais bzw. Links per Telefon. Diese Erweiterung ist seit Ende 2022 auch im sm0svx-Master verfügbar. Daher ist es prinzipiell bei jeder Branch-Variante möglich eine Sip-Unterstützung zu nutzen. Um die Sip-Funktionalität einzubinden muß zunächst das pjSip-Framework installiert werden und die Sip-Funktion explizit als cmake-Option aktiviert werden, siehe hierzu Abschnitt 2.2.2 in diesem Manual.

Die folgenden SvxFramework-Erweiterungen habe ich in den letzten Jahren erstellt:

Branch „svxlink-usrp“

Mit dem svxlink-usrp-Branch ist es möglich SvxFramework mit der Analog_Bridge des MMDVM-Frameworks zu verbinden. Mit etwas Konfigurationsaufwand ist es möglich eine Verbindung Analoges Relaisnetzwerk ↔ Digitales Relaisnetzwerk herzustellen.

Branch „tetra-contrib“

Der Branch tetra-contrib ermöglicht die Ansteuerung eines Tetra-Endgerätes über die PEI und damit die Verknüpfung verschiedener Tetra-Nodes per SvxFrameworkReflector.

Branch „tetra-usrp“

Dieser Branch ist eine Kombination aus tetra-contrib und svxlink-usrp.

Branch „tetra-sip“

Der tetra-sip Branch beinhaltet die pjSipLogic und tetra-contrib, unterstützt also beide Versionen. Sie können also ein Tetra-Endgerät über SvxFramework mit einem Sipserver verbinden.

Branch „tetra-sip-usrp“

Dieser Branch ist eine Kombination aus tetra-contrib, pjSipLogic und svxlink-usrp.

Die beiden letzten Branches (tetra-sip und tetra-sip-usrp) sind seit der Einbindung von pjSipLogic überflüssig geworden und werden demnächst gelöscht.

Branch „hotspot-nmea“

Dieser Branch unterstützt ein externes Gerät, das NMEA-Daten über eine serielle Schnittstelle bereitstellt. Im Allgemeinen kann hierfür eine Standard-GPS-Maus genutzt werden. Ein weiteres Feature ist die Unterstützung des gpsd-Protokolls.

Branch „AnnounceLogic“

Die AnnounceLogic kann für das Streamen eines Rundspruches in eine RepeaterLogic oder einen Relaisverbund verwendet werden.

Modul „KatWarn“

Nimmt Gefahrenmeldungen (Großbrände, Hochwasser, Orkane, größere Unfälle, terroristische Gefahrenlagen, ...) vom Katwarn-Dienst per eMail entgegen und gibt diese zeitnah per generierter Sprachansage auf dem Relais/Link aus.

Modul „Unwetterwarnungen“

Gibt wetterbedingte Gefahrenmeldungen als Sprachansage aus. Die Meldungen werden nach Registrierung beim Deutschen Wetterdienst als Service an die angegebene eMail-Adresse gesendet. SvxLink erstellt daraus per TTS eine Sprachansage und gibt diese zeitnah per Ansage auf dem Relais/Link aus.

Modul „Announcement“

Empfängt eine Text-eMail und gibt den Inhalt als Sprachausgabe auf dem Relais/Link aus.

2. Vorbereitungen

2.1. Hardware

Mit Ausnahme der Modulerweiterungen ist für die Installation der Branches eine Entwicklungsumgebung auf dem Node notwendig. Das Kompillieren von Software benötigt relativ viel RAM und CPU-Ressourcen. Fertige Images kann ich leider nicht anbieten, es gibt allerdings für den tetra-contrib-Branch ein Image von Lawrence/DL1FLW [14].

Der Betrieb von SvxLink ist mit einigen Abstrichen zwar theoretisch auf einem OrangePi mit 256MB RAM und 2GByte SD-Card möglich aber definitiv nicht zu empfehlen. Insbesondere während des Kompilierens wird es auf Grund mangelnder Ressourcen zu Problemen kommen. Auch während des normalen Betriebs steht man Problemen gegenüber. Auf Grund der eingeschränkten Ressourcen müssen z.B. Logfiles öfter rotiert oder gelöscht werden als auf „normalen“ Systemen. Die Audiodatenverarbeitung erfordert auch Ressourcen und ein SvxLink kann auf einem RaspberryPi 3 schon mal permanent 25% CPU-Last erzeugen.

Hardware-Anforderungen:

- RaspberryPi Modell 3 oder besser oder normaler Desktop-PC
- 1GByte RAM oder mehr
- 16GByte SD-Card
- entsprechender Adapter für die Verbindung TRX ↔ Raspi/PC

Als Interface bzw. Adapter zwischen Funkgerät und Raspi/PC eignen sich mit Ausnahme der TetraLogic nahezu alle im Handel erhältlichen EchoLink-Adapter. Beachten Sie aber, dass hier adapterspezifische Konfigurationen notwendig sein werden. Im Internet finden sich auch eine ganze Reihe von Anleitungen für Selbstbau-Adapter oder auch fertige SvxHats, die einfach auf z.B. den Raspi aufgesteckt werden können und die erforderlichen Anschlüsse für das Funkgerät (normalerweise Audio-IN, Audio-OUT, PTT und Squelch) bereitstellen.

2.2. Software

Im Gegensatz zu den zusätzlichen Modulen (Beschrieben in Abschnitt 7) ist für die Logik-Erweiterungen (pjSipLogic, Tetra, USRP) auf dem Gerät eine Entwicklungsumgebung notwendig. Fertige Images stelle ich dafür leider nicht bereit. Wenn Sie diese Logik-Erweiterungen nicht installieren möchten und Sie bereits ein lauffähiges SvxLink in Betrieb haben so gehen Sie bitte direkt zu Abschnitt 7 (Installation der Zusatzmodule).

Für die SvxLink-Installation wird ausdrücklich die Installation eines Betriebssystems **ohne graphische Oberfläche** insbesondere ohne pulseaudio empfohlen.

Die folgenden Anleitungen beziehen sich auf das Vorhandensein eines Debian oder Ubuntu (Serverversion). Für die Installation eines Betriebssystems auf einem Raspi gibt es genügend Anleitungen, z.B. [15], es soll daher nicht Bestandteil dieser Anleitung sein.

Tip: Falls Sie sich für eine der Logik-Erweiterungen interessieren, so besteht die Möglichkeit ein Installationsskript herunterzuladen und zu starten. Dieses installiert halbautomatisch die benötigten Pakete für genau die spezielle Variante (und nur für diese) auf Ihrem System. Eine Konfiguration findet nicht statt! Siehe hierzu [8].

2.2.1. Installation der Entwicklungsumgebung

Ich empfehle sich zunächst für eine Installationsvariante zu entscheiden und die Installation auf einem „nackten“ Linux-System zu starten. Reste früherer SvxLink-Installationen könnten möglicherweise zu Problemen führen.

Für ubuntu- bzw. debianbasierte Systeme installieren Sie die folgenden Pakete mittels apt als root bzw. mittels sudo:

```
sudo apt update
sudo apt install gcc g++ wget make cmake libgcrypt20-dev libgsm1-dev libsigc++-
2.0-dev tcl-dev git libspeex-dev libasound2-dev libpopt-dev libssl-dev libopus-
dev groff libcurl4-openssl-dev libjsoncpp-dev gpiod libgpiod-dev
```

Auf OpenSUSE gibt es dafür den zypper:

```
zypper up
zypper install gcc gcc-c++ wget make cmake libgcrypt20-devel libgsm-devel
alsa-devel libsigc++2-devel tcl-devel git speex-devel libopus-devel
libasound2 popt-devel openssl-devel libjsoncpp-dev gpiod libgpiod-dev
```

Beim späteren Betrieb sind weitere Pakete sinnvoll, am besten Sie installieren diese zeitnah bevor Sie mit der weiteren Installation fortfahren.

Für die Unterstützung eines Rtl-SDR-Sticks benötigen Sie die libusb-dev-Bibliothek:

```
sudo apt-get install libusb-1.0.0-dev
```

Für die Einstellung der TX- und RX-NF-Pegels steht der Alsa-mixer zur Verfügung. Dieser ist ein verhältnismäßig wichtiges Werkzeug und sollte auf keinem System fehlen.

```
sudo apt-get install alsa-base alsa-utils
```

Ein weiteres meiner Meinung nach wichtiges Tool ist der Midnight-Commander (vielleicht kennt noch jemand den Norton-Commander) um Dateien zu kopieren und Konfigurationsfiles zu bearbeiten.

```
sudo apt-get install mc
```

Übersicht und kurze Beschreibung der von SvxLink genutzten Softwarepakete:

gcc (erforderlich)

Der GNU C-Compiler zum Kompilieren der SvxLink-Quellen welche in C geschrieben sind

g++ bzw. gcc-c++ (erforderlich)

Der GNU C++-Compiler zum Kompilieren der SvxLink-Quellen welche in C++ geschrieben sind

make (erforderlich)

Make ist ein Tool, mit dem man Abhängigkeiten eines Buildprozesses auflösen kann.

cmake (erforderlich)

Cmake ist ein Programmierwerkzeug, welches als Build-System genutzt wird.

libgcrypt-devel (erforderlich)

Kryptobibliothek die Funktionen wie AES, SHA-x, MDx usw. bereitstellen. In SvxLink wird diese Bibliothek zur Passwortauthentifizierung verwendet

libgsm-devel

Bibliotheken für den GSM-Codec (Sprachkompression). In SvxLink wird diese Bibliothek für EchoLink-Verbindungen und zwischen SvxReflector und den SvxLink-Nodes verwendet.

Alsa-devel (erforderlich)

Bereitstellung eines Audio/MIDI-Frameworks für die Linux Sound Architecture (ALSA).

libsigc++2-devel (erforderlich)

Diese Bibliothek beinhaltet ein vollständiges Callback-System für die Benutzung in Widget-Bibliotheken. Sie ist notwendig für die ereignisgesteuerte Abarbeitung innerhalb von SvxLink.

tcl-devel (erforderlich)

Eine um 1988 entwickelte Programmiersprache, die in SvxLink als Userinterface zur Steuerung von Sprachausgaben verwendet wird.

speex-devel (empfohlen)

Ressourcen für den Speex-Sprachcodec, dieser kann beim Remotetrx und svxreflector verwendet werden sowie bei EchoLink-Verbindungen zwischen zwei SvxLink-Nodes

libopus-devel (erforderlich)

Ressourcen für den Opus-Sprachcodec, dieser kann beim remotetrx und svxreflector verwendet werden sowie bei EchoLink-Verbindungen zwischen zwei SvxLink-Nodes

libasound2 (erforderlich)

Die ALSA-Bibliothek und ihre Standarderweiterungen sowie die erforderlichen Konfigurationsdateien

popt-devel (erforderlich)

Bibliotheken zum Parsen von Kommandozeileingaben.

openssl-devel (erforderlich)

Software für Transport Layer Security. Sie beinhaltet verschiedene Netzwerkprotokolle und Verschlüsselungen.

libjsoncpp-dev (erforderlich)

Behandlung von json-Konfigurationsfiles zum einlesen von Nutzerdaten und Node-Beschreibungen

gpod (optional, für Raspi erforderlich)

Verschiedene Tools zum Interagieren mit Linux GPIO character-devices.

libgpod-dev (optional, für Raspi erforderlich)

Header und Entwicklungsumgebung für gpod

wget (optional)

Mit diesem Programm kann man direkt aus einem Terminal Dateien von FTP-, HTTP- oder HTTPS-Servern herunterladen.

git (empfohlen)

Versionsmanagement-Framework um den SvxFramework-Sourcecode von github.com herunterzuladen.

mc (optional)

Der Midnight-Commander ist ein Tool zum Kopieren und Bearbeiten von Dateien im Dateisystem. Funktionalität wie der Norton-Commander

2.2.2. Installation des pjSip-Frameworks

Wenn Sie sich für die Sip-Erweiterung (Abschnitt 3) interessieren, so müssen Sie zunächst mit der Installation des pjSip-Framework auf Ihrem System fortfahren. Eine detailliertere Darstellung der Installationsschritte ist in Bild 3 (Abschnitt 3.1) zu sehen.

Inzwischen konnte die experimentelle pjSipLogic-Erweiterung auch mit neueren Versionen des pjsip-Frameworks kompiliert werden. Getestet wurde mit den Versionen 2.10, 2.11 und 2.12.1. Die Unterstützung für libwebrtc und video ist nicht erforderlich und sollte deaktiviert werden.

```
wget https://github.com/pjsip/pjproject/archive/2.12.1.tar.gz
tar -xvzf 2.12.1.tar.gz
cd pjproject-2.12.1
./configure --disable-libwebrtc --disable-video
make dep
make
sudo make install
```

ACHTUNG: Auf einigen Plattformen muß pjSip mit dem Compilerflag -fPIC kompiliert werden, anderenfalls kommt es beim Linken oder beim SvxFramework-Start zu diversen Fehlermeldungen wie z.B. SipLogic.so: unexpected reloc type 0x03

In diesem Fall muß pjSip wie folgt konfiguriert bzw. installiert werden:

```
./configure --disable-libwebrtc --disable-video CPPFLAGS=-fPIC CXXFLAGS=-fPIC
CFLAGS=-fPIC
```

Achten Sie auf eventuelle Fehlermeldungen und setzen Sie mit Abschnitt 2.2.3. erst fort, wenn keine Fehlermeldungen mehr aufgetreten sind.

Damit steht das Sip-Framework zur Verfügung das für die Sip-Unterstützung im SvxFramework benötigt wird. Die Installation ist optional, für einen „normalen“ SvxFramework-Betrieb ohne Telefoneinwahl ist Sip nicht erforderlich.

2.2.3. Installation einer Branch-Version SvxFramework

Falls noch nicht geschehen legen Sie die Gruppe svxfw und den Benutzer svxfw an:

```
groupadd svxfw
useradd -g svxfw -G svxfw -c "SvxFramework Admin" svxfw
```

Laden Sie jetzt bitte das DL1HRC-Repository auf Ihr System herunter:

```
git clone http://github.com/dl1hrc/svxlink.git
```

Checken Sie jetzt den für Sie interessanten Branch aus:

```
cd svxlink
git checkout xxxxxxx
```

wobei xxxxxxx dem Name des interessierenden Branches entspricht. xxxxxxx steht dabei für

- svxlink-usrp
- tetra-contrib
- ~~pjSipLogic~~ (ist jetzt Bestandteil des Hauptentwicklungszweiges)
- tetra-usrp
- ~~tetra-sip~~ (nicht mehr notwendig)
- ~~tetra-sip-usrp~~ (nicht mehr notwendig)
- hotspot-nmea
- AnnounceLogic

Achtung: Wenn das Verzeichnis schon durch frühere Versuche existiert, dann funktioniert das so natürlich nicht. Hier müssen Sie mit den git-Werkzeugen weiterarbeiten oder das Verzeichnis löschen. Eine Anleitung zum Werkzeug *git* ist hier zu finden [16].

2.2.4. Kompilierung und Installation eines Branches oder einer experimentellen Erweiterung

Wechseln Sie in das Verzeichnis *svxlink* und erstellen Sie das *build*-Verzeichnis

```
cd svxlink/src
mkdir build
cd build
```

Für den Fall, dass das *build*-Verzeichnis schon von früheren Installationsversuchen existieren sollte und nicht leer ist empfiehlt sich zunächst ein

```
make clean
```

auszuführen. Starten Sie *cmake* wie folgt:

```
cmake -DUSE_QT=OFF -DCMAKE_INSTALL_PREFIX=/usr -DSYSCONF_INSTALL_DIR=/etc -
DLOCAL_STATE_DIR=/var -DWITH_SYSTEMD=ON -DCMAKE_BUILD_TYPE=Release ..
```

Um eine experimentelle Erweiterung wie die *pjSipLogic* zu installieren müssen Sie den Schalter *-DWITH_CONTRIB_SIP_LOGIC=ON* setzen:

```
cmake -DUSE_QT=OFF -DCMAKE_INSTALL_PREFIX=/usr -DSYSCONF_INSTALL_DIR=/etc -
DLOCAL_STATE_DIR=/var -DCMAKE_BUILD_TYPE=Release -DWITH_CONTRIB_SIP_LOGIC=ON -
DWITH_SYSTEMD=ON ..
```

Wollen Sie die Funktionalitäten Sip, Tetra und Usrp installieren, so checken Sie zunächst den entsprechenden tetra-usrp-Branch aus (siehe Abschnitt 2.2.3) und starten danach *cmake* mit den Schaltern

```
cmake -DUSE_QT=OFF -DCMAKE_INSTALL_PREFIX=/usr -DSYSCONF_INSTALL_DIR=/etc -
DLOCAL_STATE_DIR=/var -DCMAKE_BUILD_TYPE=Release -DWITH_CONTRIB_SIP_LOGIC=ON -
DWITH_CONTRIB_TETRA_LOGIC=ON -DWITH_SYSTEMD=ON -DWITH_CONTRIB_USRP_LOGIC=ON ..
```

Starten Sie nun *make* und installieren Sie SvxFink:

```
make
sudo make install
sudo ldconfig
```

Für den Betrieb von neueren SvxFink-Versionen hat sich die Art des Zugriffs auf die GPIO's bzw. seriellen Schnittstellen geändert. Der Zugriff auf die Pins bzw. Ports erfolgen jetzt nicht mehr auf direktem Weg sondern über den gpio-Daemon. Dafür zuständig ist die Datei */etc/svxfink/devices.conf*

```
#####
#
# Configuration file for the SvxFink server Device Pins
#
#####
# The purpose of this file is to define a list of device paths (either hidraw
# or serial) that SvxFink will need access to. Devices defined here will have
# their ownership and permissions set upon SvxFink startup so that they may be
# properly accessed by the SvxFink service.
# Space separated list of device paths. Typically hidraw and serial devices.
# Examples: /dev/hidraw0 /dev/ttyUSB0

DEV_LIST="/dev/ttyS0"
#DEV_LIST="/dev/hidraw0 /dev/hidraw1"

# User that should own the devices
DEV_USER="svxfink"

# Group for the devices
DEV_GROUP="daemon"
# File access mode for the devices
DEV_MODE="0664"
```

Konfigurieren Sie die entsprechende Schnittstelle in dieser Datei (hier im Beispiel */dev/ttyS0*), damit SvxFink darauf zugreifen kann.

Desweiteren müssen in der */etc/groups* (falls noch nicht geschehen) die Berechtigungen für den User SvxFink festlegen, damit SvxFink nicht mit root-Rechten laufen muß. Der User „svxfink“ sollte dabei den Gruppen

adm, tty, dialout, audio, plugdev, users, input, netdev, pi, spi, i2c, gpio
angehören.

2.2.5 Installation der Sprachumgebung

Für die Installation der Sprachumgebung stehen Pakete in verschiedenen Sprachen (Englisch [39], Deutsch [25], Italienisch [38], Spanisch, Französisch [43], Schwedisch [40]) zur Verfügung. Die englischen und italienischen Pakete sind für die Verwendung auf allen Nodes freigegeben.

Beachten Sie bitte, dass für die Verwendung der deutschen Linguatex-Sprachansagen der Kauf einer Homeversion des Voice-Readers erforderlich ist [24]!

Um die Sprachumgebung zu installieren, entpacken Sie das File *svxfink-sounds-de_DE-petra.tar.gz* nach */usr/share/svxfink/sounds/de_DE*

so dass sich folgende Struktur ergibt:

Links	Datei	Befehl	Optionen	Rechts
Verzeichnisbaum		← ...share/svmlink/sounds/de_DE → [↓]		
	sgml-data		↙ Name	Größe Modifikations
	snmp		/..	ÜBERVZ. 19. Apr 2020
	software-properties		/.git	4096 19. Apr 2020
	solid		/Core	4096 19. Apr 2020
	sounds		/Default	4096 09. Mai 2020
	speech-dispatcher		/DtmfRepeater	4096 19. Apr 2020
	ssl-cert		/EchoLink	4096 19. Apr 2020
	svmlink		/Help	4096 19. Apr 2020
	events.d		/MetarInfo	12288 19. Apr 2020
	modules.d		/Parrot	4096 19. Apr 2020
	sounds		/PhoneLogic	4096 19. Apr 2020
	de DE		/PropagationMonitor	4096 19. Apr 2020
	.git		/SelCallEnc	4096 19. Apr 2020
	Core		/SipLogic	4096 19. Apr 2020
	Default		/TclVoiceMail	4096 19. Apr 2020
	DtmfRepeater		/TrafficInfo	4096 19. Apr 2020
	EchoLink		/WeatherInfo	4096 19. Apr 2020
	Help		README.md	294 19. Apr 2020
	MetarInfo			
	Parrot			
	PhoneLogic			
	PropagationMonitor			
	SelCallEnc			
	SipLogic			
	TclVoiceMail			
	TrafficInfo			
	WeatherInfo			
	system-config-printer			
/usr/share/svmlink/sounds/de_DE		ÜBERVZ. 48G/124G (38%)		

Bild 1: Verzeichnisstruktur der Soundfiles für eine deutsche Sprachumgebung

Da SvxFLink mehrere Sprachen unterstützt muß der svxFLink.conf-Parameter *DEFAULT_LANG=de_DE* dem Verzeichnisabschnitt in */usr/share/svxFLink/de_DE* entsprechen, sonst werden die Soundausgaben nicht gefunden. Vergessen Sie nicht, die Berechtigungen zu setzen (Gruppe svxFLink, user svxFLink).

```
Sudo chown -R svxFLink:svxFLink /usr/share/svxFLink/sounds
```

Damit ist die Grundinstallation abgeschlossen.

2.2.6. Konfiguration von SvxFLink

Beim probeweisen Starten von SvxFLink kann es zu Fehlermeldungen kommen wie z.B. das eine Soundkarte nicht gefunden wurde oder der EchoLink-Zugang nicht funktioniert. Um diese Fehler zu beseitigen sind entsprechende Parameter in der */etc/svxFLink/svxFLink.conf* zu konfigurieren und müssen an Ihr System entsprechend der vorhandenen Hardware angepasst werden.

Für eine Beschreibung der Konfigurationsparameter gibt es hier eine Dokumentation [17]. Auch die etwas in die Jahre gekommene SvxFLink.de-Webseite informiert zumindest über die meisten Parameter. Konfigurieren Sie zunächst eine einfache Simplex- oder RepeaterLogic so dass die Grundfunktionalität gewährleistet ist. Es macht keinen Sinn weiterzumachen, wenn nicht wenigstens eine SimplexLogic lauffähig ist und aus diesem Grund SvxFLink gar nicht erst startet.

Alsamixer: Für den ersten Versuch starten Sie den alsamixer mit dem Index der verwendeten Soundkarte als Option. Wenn Sie z.B. an einem RaspberryPi eine USB-Soundkarte verwenden, so wird diese vermutlich (ohne weitere Konfiguration am Linux-System) den Index 1 besitzen. In der svxlink.conf würde also folgendes einzutragen sein:

```
[Rx1]
AUDIO_DEV=alsa:plughw:1
AUDIO_CHANNEL=0
```

```
[Tx1]
AUDIO_DEV=alsa:plughw:1
AUDIO_CHANNEL=0
```

In dieser Konfiguration wird die Soundkarte mit dem Index 1 und der Kanal 0 verwendet. Wenn unter [GLOBAL] der Parameter CARD_CHANNELS=1 konfiguriert wurde, ist der konfigurierte AUDIO_CHANNEL egal, da sowohl der linke als auch der rechte Kanal identische Audiostreams enthalten/auswerten.

Rufen sie den alsamixer hierbei mit

```
alsamixer -c1
```

auf wobei „-c1“ die Soundkarte mit dem Index 1 anspricht. Nach dem Start zeigt sich eine simple textbasierte Umgebung in der Sie die Audiopegel für den Sende- und Empfangszweig einstellen können.

Um die veränderten Audio-Pegeleinstellungen permanent zu halten (auch nach einem Neustart des Systems) speichern Sie diese mit

```
alsactl store
```

ab. Um die abgespeicherten Audio-Pegeleinstellungen wieder herzustellen rufen Sie

```
alsactl restore
```

auf.

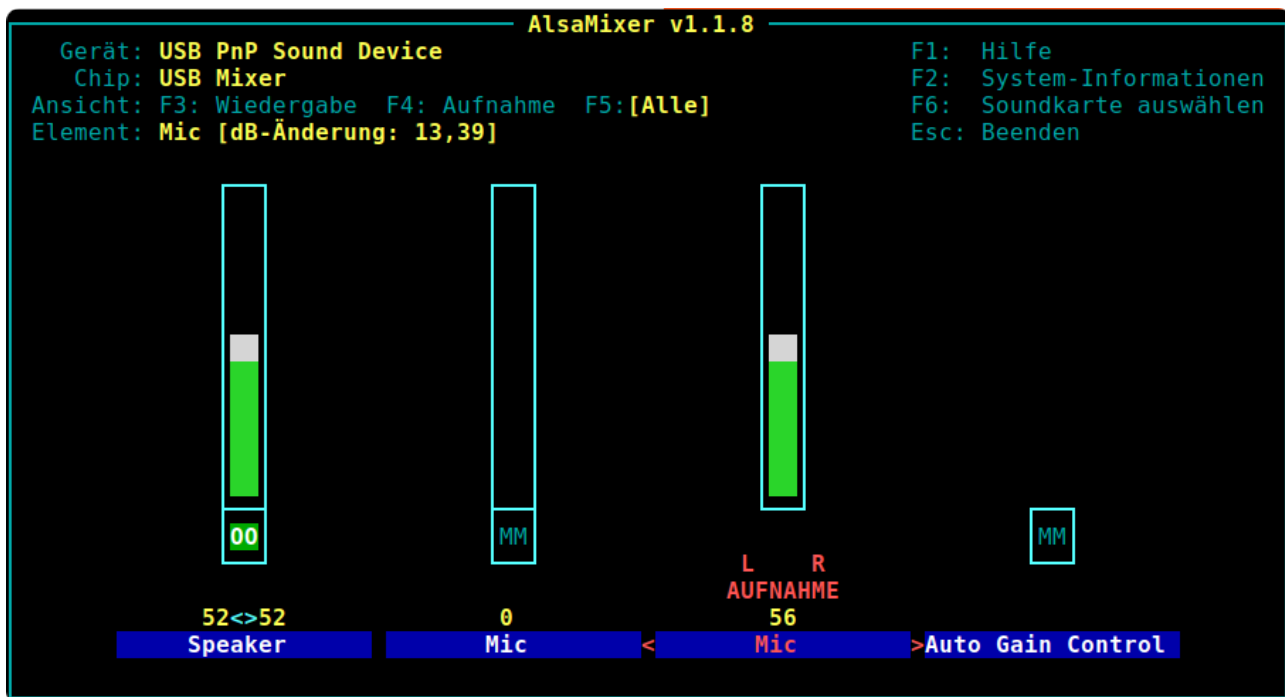


Bild 2: Typische Einstellungen im alsamixer für eine Standard-USB-Soundkarte

Für den Anfang stellen Sie die Regler für Speaker/PCM und Capture auf ca. 50%. Sollten beim späteren Betrieb Reglereinstellungen nahe 100% oder 0% erforderlich sein, damit die Audio funktioniert so ist das verdächtig. Hier muß hardwaremäßig nachgebessert werden. Verwenden Sie den PREAMP-Parameter nur im absoluten Ausnahmefall. Der bei USB-Soundkarten häufig anzutreffende „Auto Gain Control“-Schalter sollte deaktiviert werden. Der Eingang Mic ist insbesondere bei Relaissteuerungen stumm zuschalten, das eingehende Audio wird über den Capture-Eingang verarbeitet.

Wichtiger Hinweis:

Werden Nodes in einem Verbund (über den SvxFlector) betrieben, so muß das Audio der einzelnen Nodes aufeinander angepasst werden. Nicht alles was lokal relativ gut „klingt“ wird auf der jeweils anderen Seite genauso wahrgenommen. In vielen Fällen wird der Rx-Pegel viel zu niedrig (oder zu hoch) eingeregelt und dann, damit es lokal gut klingt, entsprechend der Tx-Regler angepasst. Für den Audiopegel, der IN den Netzverbund gesendet wird sind am lokalen Node aber der Capture und Line-In bzw. Mic-Regler zuständig und diese bestimmen damit mehr oder weniger direkt die Lautstärke auf den jeweils anderen Nodes. In jedem Fall sollte eine Kalibrierung auf jedem einzelnen Node erfolgen, d.h. ein definierter Rx-Eingangspegel, der aber abhängig von der jeweils verwendeten Soundkarte ist!

Die Datei `/etc/svmlink/node_info.json` wird verwendet, wenn nodebezogene Informationen zum SvxFlector übertragen um dort z.B. in einem Dashboard visualisiert werden sollen.

Parametrieren Sie diese Datei entsprechend Ihrer örtlichen Verhältnisse und beachten Sie, dass die json-Datei „wohlgeformt“ sein muß. Wenn Sie diese Grundkonfiguration erfolgreich abgeschlossen haben setzen Sie mit dem Abschnitt fort, der für Sie interessant ist.

2.2.7. Aktualisierung eines Branches

Von Zeit zu Zeit gibt es Updates, wenn Sie diese installieren wollen so wechseln Sie in das svmlink-Verzeichnis (normalerweise in Ihrem *home*-Verzeichnis) und starten Sie das Update wie folgt:

```
cd $HOME/svmlink  
git pull
```

Der Befehl *git pull* „zieht“ die aktuelle Version vom github-Repository auf Ihren Rechner. Führen Sie danach:

```
cd src/build  
make  
sudo make install
```

aus, damit sind Sie auf dem aktuellen Stand. Um herauszufinden, welchen Branch Sie gerade nutzen geben Sie ein

```
cd $HOME/svmlink  
git status
```

Eine mögliche Ausgabe wäre z.B.

```
adi@adi-PORTEGE-Z930:~$ cd svmlink  
adi@adi-PORTEGE-Z930:~/svmlink$ git status  
Auf Branch master  
Ihr Branch ist auf demselben Stand wie 'origin/master'.  
  
nichts zu committen, Arbeitsverzeichnis unverändert  
adi@adi-PORTEGE-Z930:~/svmlink$
```

Sie befinden sich also im „master“, dem Hauptentwicklungszweig. Um den Branch zu wechseln (z.B. zum „tetra-contrib“) führen Sie aus:

```
adi@adi-PORTEGE-Z930:~/svmlink$ git checkout tetra-contrib  
Zu Zweig »tetra-contrib« gewechselt  
Ihr Branch ist 69 Commits hinter 'origin/tetra-contrib', und kann vorgespuelt werden.  
(benutzen Sie "git pull", um Ihren lokalen Branch zu aktualisieren)  
adi@adi-PORTEGE-Z930:~/svmlink$
```

In diesem Fall sollten Sie ein *git pull* ausführen, damit Ihr lokaler Branch auf den aktuellen Stand gezogen wird.

Für weiterführende Informationen lesen Sie bitte unter [16] nach.

3. SIP-Erweiterungen

Die pjSip-Erweiterung steht seit Ende 2022 im Hauptentwicklungszweiges zur Verfügung. Infolge dessen ändert sich die Vorgehensweise leicht. Bitte die älteren Versionen dieser Anleitung nicht mehr als Referenz verwenden.

3.1. Experimentelle Erweiterung pjSipLogic

Die folgenden Informationen beziehen sich vorrangig auf den Branch pjSipLogic im DL1HRC-Repo. Die SipLogic ist eine neue Logic die als experimentelle Komponente hinzu-kompiliert werden kann. Sie ist in der Lage sich als Sip-Client mit einem Telefonnetz zu verbinden um eine Überleitung Telefon → Relais/Link zu ermöglichen. Sie können also Ihr Relais oder Link je nach Konfiguration vom Festnetz oder HAMNET aus anrufen. Entsprechende Funktionalitäten gibt es schon länger beim Allstar-Projekt (Asterisk mit Relaissteuerung) [18]. An dieser Stelle herzlichen Dank an Peter/SA2BLV für das Finden und Fixen von Fehlern in meiner Logic, die zu Abstürzen führten.

Zur Installation nutzen Sie am besten das Skript von meiner Webseite [8]. Falls Sie die Installation „zu Fuß“, also manuell durchführen möchten so hilft Bild 3, welches die Abfolge der notwendigen Installationsschritte darstellt.

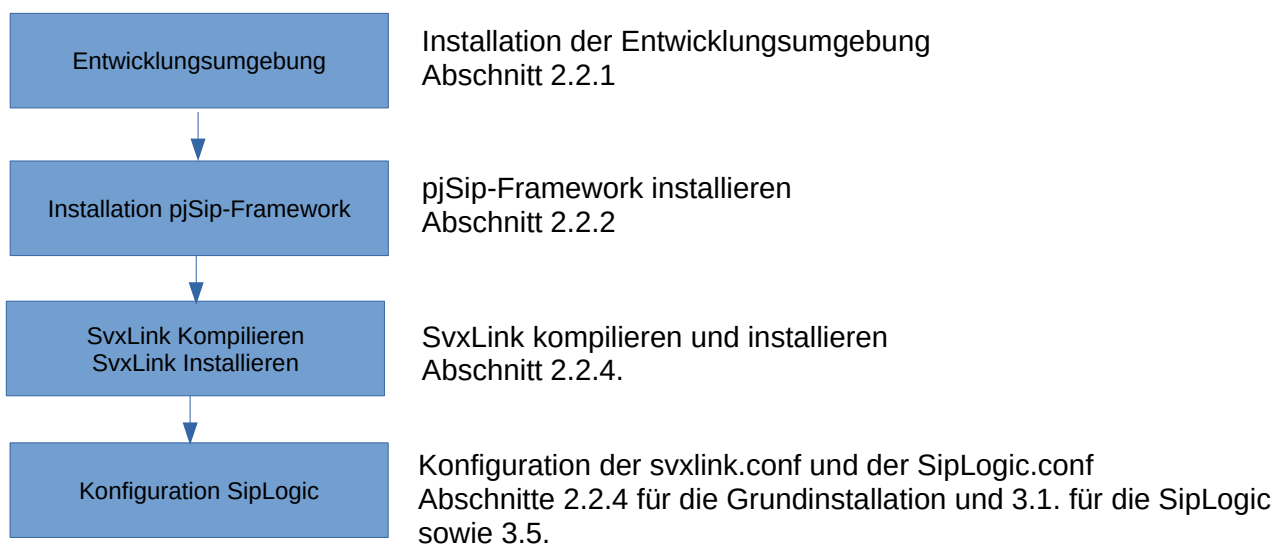


Bild 3: Schrittfolge zur Installation der experimentellen Sip-Unterstützung

Falls Sie bei der manuellen Installation auf Probleme oder Fehlermeldungen stoßen sollten, so hilft Abschnitt 9.2. (Hilfe erhalten) möglicherweise weiter.

Nachfolgend ein Beispiel für eine [SipLogic] Konfiguration innerhalb der svxlink.conf. Es sind nur die wichtigsten Konfigurationsparameter dargestellt, gegebenenfalls sind weitere Parameter für ein ordnungsgemäßes Funktionieren notwendig. Die eigentliche Sip-Konfiguration erfolgt in der Datei SipLogic.conf im Verzeichnis `/etc/svxlink/svxlink.d/`

Diese Datei wird automatisch gelesen, sobald man die experimentelle pjSipLogic-Erweiterung enabled hat. Hier zunächst die svxlink.conf-Einstellungen:

/etc/svxlink/svxlink.conf

```
[GLOBAL]
LOGICS=SipLogic, RepeaterLogic
CFG_DIR=svxlink.d
TIMESTAMP_FORMAT="%c"
CARD_SAMPLE_RATE=48000
CARD_CHANNELS=1
LINKS=SipLink

# Dieser Abschnitt (SipLink) ist erforderlich um beide
# Logiken miteinander zu verbinden
[SipLink]
CONNECT_LOGICS=RepeaterLogic, SipLogic
DEFAULT_ACTIVE=1

[RepeaterLogic]
# hier eine Standard-Konfiguration für einen Repeater einfügen
...
```

In der SipLogic.conf erfolgt die eigentliche Sip-Konfiguration wie Account- und Logindaten:

/etc/svxlink/svxlink.d/SipLogic.conf

```
[SipLogic]
TYPE=Sip
LANGUAGE=de_DE

# Serverkonfiguration
SIPSERVER=my.sipserver.org:6099
SIPREGISTRAR=my.sipserver.org:6099
SIPPORT=6099

# Logindaten
CALLERNAME=DL1ABC <2621234>
USERNAME=2621234
PASSWORD=secret

# pseudo-tty-Device zur Steuerung der SipLogic
SIP_CTRL_PTY=/tmp/sipctrl

# automatischen Abheben bei einem Anruf aus dem Handy-/Festnetz
AUTOANSWER=1

# bei SEMI_DUPLEX=1 wird der Anruf über Sip permanent durchgeschaltet
# das darf nur dann aktiviert werden, wenn sich keine Simplex-Nodes
# im gesamten Verbund befinden.
# Um sicher zu gehen sollten Sie diesen Parameter erst einmal
# deaktivieren
SEMI_DUPLEX=0

# Squelch-Einstellungen für SEMI_DUPLEX=0
VOX_THRESH=1000
VOX_FILTER_DEPTH=20
SQL_START_DELAY=0
SQL_DELAY=0
SQL_HANGTIME=1500

# Eventhandler
EVENT_HANDLER=/usr/share/svxlink/events.tcl
```

```

# Nutzungsbeschränkung nach Regex, für weitere Informationen siehe [26]
#REJECT_INCOMING=^()$
#ACCEPT_INCOMING=^(.*)$
#REJECT_OUTGOING=^()$
#ACCEPT_OUTGOING=^(.*)$

# sonstige Parameter:
DEFAULT_LANG=de_DE
SIPSCHEMA=digest
SIPEXTENSION=default

# maximale Zeit bis ein Anruf entgegengenommen wird in Sekunden
CALL_TIMEOUT=45

# SIP_LOGLEVEL: 0-wenige Ausgaben, 4-viele Ausgaben
SIP_LOGLEVEL=0

# Intervall der Registrierung am Sip-Server in Sekunden
REG_TIMEOUT=300
#JITTER_BUFFER_DELAY=300

# Vorverstärker für den Netzwerk-Sip-Audiostrom, falls dieser zu leise
# sein sollte - bitte nicht zu weit „aufdrehen“
SIP_PREAMP=3
SIP_LIMITER_THRESH=-1.0

# Eingehende Sip-Calls werden in konfigurierte TG's weitergeleitet
PHONENUMBERS_TO_TG=0049:262,0034:232,0041:222,017612345678:9999

```

Achten Sie darauf, dass der Zugang zur HF nur für lizenzierte Funkamateure möglich sein darf.

Bei der Verwendung von Fritz!Box-en (und vermutlich allen anderen Routern, die Telefonie per Sip bereitstellen) im eigenen Netzwerk ist darauf zu achten, dass diese standardmäßig den SIP-Port 5060 nutzt, eine Port-Weiterleitung vom DSL in das eigene Netzwerk an einen Asterisk ist für den Port 5060 also nicht möglich. Daher ist für die eigenen HAMRADIO-Anwendungen ein anderer Port als 5060 zu wählen.

Beachten Sie, dass der Parameter SEMI_DUPLEX=1 nur gesetzt werden darf, wenn alle im Netzwerk befindlichen Nodes duplexfähig (also nur Repeater oder SIP-Clients) sind. Ein SimplexNode würde bei einer SIP-Verbindung sonst niemals auf Empfang schalten. Ist dieser Parameter auskommentiert oder als SEMI_DUPLEX=0 konfiguriert so wird für die SIP-Verbindung (und nur dann) eine VOX-Steuerung aktiviert. Das Verhalten dieser ist über die Parameter VOX* bzw. SQL* steuerbar. Der eingehende Audiostream wird damit einem VOX-Kriterium unterworfen. Die Übertragung auf den Funk wird also vom Level des eingehenden SIP-Audiostreams abhängig gemacht. Hier wird also für den OM am Telefon eine gewisse funktechnische Disziplin vorausgesetzt und es muß eine mögliche gute audioseitige Störunterdrückung gegeben sein, damit ein QSO durchgeführt werden kann. Der Abfall der SIP-Squelch wird dabei genau wie am Funk durch einen Roger-Beep signalisiert. Das kann natürlich konfiguriert werden.

Momentan besteht noch das Problem, dass die Reflektoren nur Simplex unterstützen und damit SEMI_DUPLEX=1 nur am Node funktioniert, der per Sip angewählt wird. An einer Lösung wird gerade gearbeitet.

Eine SipLogik unterstützt derzeit noch keine Module. Das ist möglicherweise für spätere Versionen vorgesehen.

a) Anwahl eines SIP-Teilnehmers von der Linux-Konsole aus

Voraussetzung ist die Konfiguration des Parameters SIP_CTRL_PTY=/tmp/sip_ctrl

Mit dem Kommandozeilenbefehl:

```
echo "C2620055#">/tmp/sip_ctrl
```

wird die Telefonnummer 2620055 angewählt. Der Nummer muß dabei das Digit „C“ vorangestellt (für „Call“) und am Ende eine Raute angefügt werden.

b) Anwahl eines Teilnehmers vom Funk ins SIP-Netz

Für die Anwahl einer Telefonnummer aus dem Funknetz heraus (per DTMF) muß die TCL-Prozedur „Simplex.tcl“ nach dem locale-Konzept erweitert werden. Erstellen Sie dazu im Verzeichnis */usr/share/svxlink/events.d/local* die Datei Simplex.tcl mit folgendem Inhalt (ohne Anspruch auf optimale Umsetzung):

```
#
# This is the namespace in which all functions and variables below will exist.
#
namespace eval SimplexLogic {

variable stored_digits "";

#
# Executed when a DTMF digit has been received
# digit      - The detected DTMF digit
# duration   - The duration, in milliseconds, of the digit
#
# Return 1 to hide the digit from further processing in SvxFunk or
# return 0 to make SvxFunk continue processing as normal.
#
proc dtmf_digit_received {digit duration} {
    variable port;
    variable stored_digits;
    puts "DTMF digit \"$digit\" detected with duration $duration ms";

    set len [string length $stored_digits];
    if {$len > 20} {
        set stored_digits "";
    }

    if {$digit == "#"} {
        if {$len > 5}
            set port [open "/tmp/sip_ctrl" w+];
            puts $port $stored_digits;
            close $port;
            set stored_digits "";
        } else {
            set stored_digits "$stored_digits$digit";
        }
        return 0;
    }
}

# end of namespace
}
```

Die über den Funkkanal und SimplexLogic empfangenen DTMF-Digits werden dabei auf das pty-Device der SipLogic weitergeleitet und dort ausgewertet. Es muß nicht extra darauf hingewiesen werden wie problematisch eine komplette Freigabe von Rufnummern sein könnte.

c) Aufnahme eines eingehenden Anrufes per Funk bei nicht automatischer Anrufannahme

Ein eingehender Anruf wird durch einen Klingelton signalisiert. Um diesen anzunehmen gibt man per Funk die DTMF-Kombination „CA“ ein (für „Call Answer“).

d) Anwahl einer EchoLink-Verbindung über die SipLogic

Michael/DL8MVB hatte die Anfrage ob man eine Lösung realisieren kann, die es erlaubt vom heimischen DECT-Handy oder normalem VoIP-Telefon einen SvxFLink-Node mit SipLogic und EchoLink-Anbindung anzuwählen um von dort eine EchoLink-Verbindung zum Lieblings-Repeater aufzubauen.

Da momentan die SipLogic noch keine Module unterstützt, muß der Umweg über die per LogicLinking verbundenen Logiken SipLogic ↔ RepeaterLogic (mit EchoLink-Modul) gegangen werden. Die Verknüpfung zwischen SipLogic und RepeaterLogic erfolgt über das per Parameter DTMF_CTRL_PTY konfigurierte Pseudo-tty.

Beispiel */etc/svxfLink/svxfLink.conf*:

```
[GLOBAL]
LOGICs=RepeaterLogic,SipLogic
LINKS=SipLink

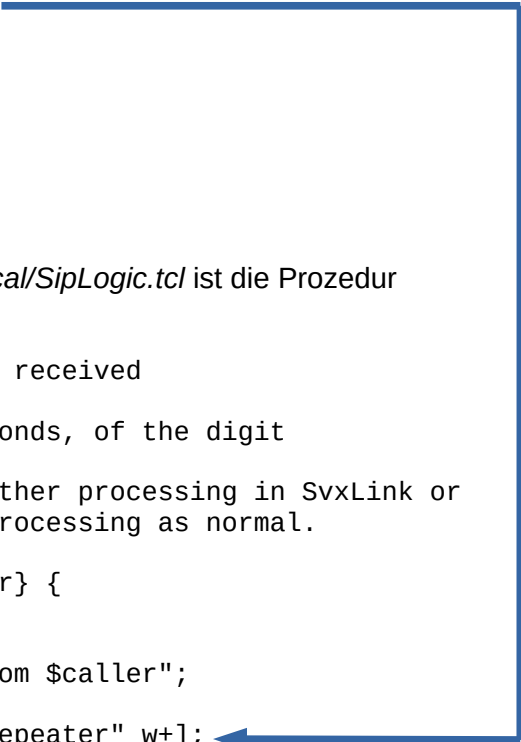
[SipLogic]
...

[RepeaterLogic]
...
DTMF_CTRL_PTY=/var/spool/dtmf_repeater
...

[SipLink]
CONNECT_LOGICs=RepeaterLogic,SipLogic
DEFAULT_ACTIVE=1
```

In der tcl-Datei */usr/share/svxfLink/events.d/local/SipLogic.tcl* ist die Prozedur `dtmf_digit_received` wie folgt anzupassen:

```
# Executed when a DTMF digit has been received
# digit - The detected DTMF digit
# code - The duration, in milliseconds, of the digit
#
# Return 1 to hide the digit from further processing in SvxFLink or
# return 0 to make SvxFLink continue processing as normal.
#
proc dtmf_digit_received {digit caller} {
    variable port;
    variable stored_digits;
    puts "Dtmf digit received $digit from $caller";
    if {$digit == "#"} {
        set port [open "/var/spool/dtmf_repeater" w+];
        puts $port $stored_digits;
        puts $port "#";
    }
}
```



```

    close $port;
    set stored_digits "";
} else {
    set stored_digits "$stored_digits$digit";
}
return 0;
}

```

Nach dem svxlink-Neustart wählen Sie Ihren Node mit der SipLogic per Telefon an, danach Starten Sie das Modul EchoLink (normalerweise per 2#) und danach die EchoLink-Nodenummer.

Im Bild 4 ist ein möglicher Beispielaufbau eines kleinen SIP-Netzwerkes als Anregung für eigene Experimente dargestellt. Das Netzwerk beinhaltet sowohl Sip-Clients als auch einen Asterisk-Server dessen Konfiguration im nächsten Abschnitt beschrieben ist. Einen Asteriskserver vollständig zu konfigurieren und zu betreiben ist für Newcomer nicht so ganz einfach und setzt einiges an Erfahrung voraus. Anfänger sollten zunächst mit einer pjSipLogic und einem Sip-Client beginnen.

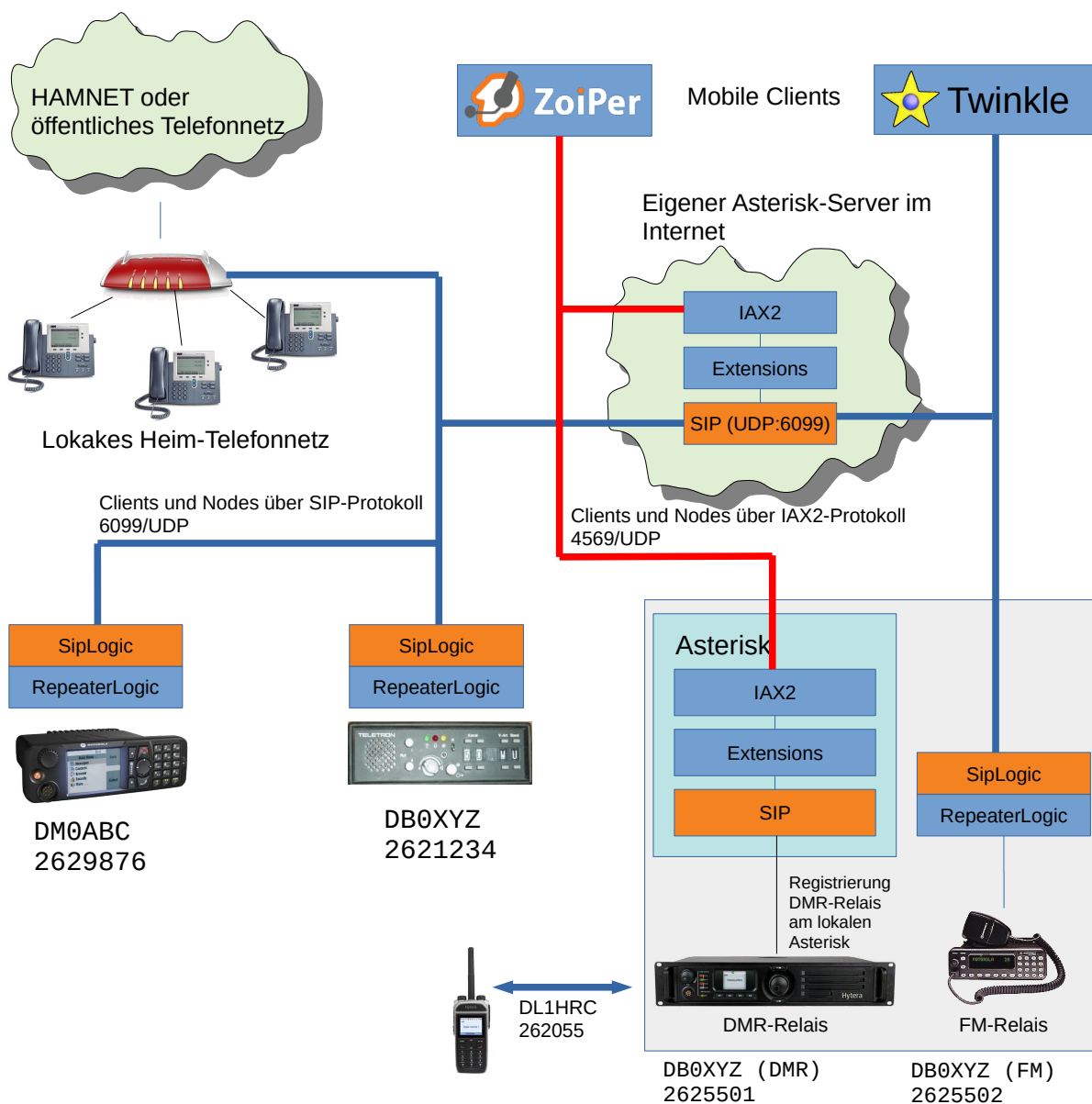


Bild 4: Beispiel für ein kleines Hamradio-SIP-Netzwerkes als Beispiel für eigene Experimente

3.2. Konfiguration von Asterisk als zentraler Telefonieserver

Ein eigener Asterisk kann verschiedene SipLogiken von verschiedenen Svxmlink-Nodes zusammenführen, so dass man in der Lage ist von einem Relais zum anderen zu „telefonieren“. Weiterhin können an diese Asterisk-Instanz weitere Svxmlink-Nodes oder Sip-Clients wie z.B. Zoiper oder Twinkle angeschlossen werden.

Die vollständige Konfiguration von Asterisk soll aber nicht Bestandteil dieser Anleitung sein. Schauen Sie bitte unter [18] für weiterführende Informationen nach. Hier wird nur auf die wichtigsten Konfigurationsabschnitte eingegangen.

Die Installation von Asterisk ist recht einfach, unter debian- oder ubuntu-basierten Systemen können Sie das relativ schnell per

```
sudo apt install asterisk
```

durchführen. Die für den Betrieb wichtigsten Konfigurationsfiles liegen normalerweise unter */etc/asterisk*:

<i>sip.conf</i>	Sip-Konfiguration für die angeschlossenen Clients
<i>iax.conf</i>	IAX2-Konfiguration für die Kommunikation zwischen zwei Asterisk-Instanzen
<i>extensions.conf</i>	Konfiguration des Anruf- und Wahlverhaltens (Dialplan)

Weiterführende Dokumentationen zur Asterisk Sip-Konfiguration finden Sie unter [19].

/etc/asterisk/sip.conf:

```
[general]
transport=udp

; bei Problemen kann man hier das Sip-Debugging einschalten
sipdebug=no

; Sprachumgebung für Ansagen
language=de
nat=force_rport,comedia

; der Netzwerk-Port über den die Kommunikation läuft
bindport=6099
srvlookup=yes

; Sicherheitsfeatures bitte in jedem Fall so einstellen:
allowguest=no
alwaysauthreject=yes

qualify=yes
rtptimeout=60
dtmfmode=rfc2833
rtpholdtimeout=300
trustpid=yes
insecure=port,invite
qualify=yes

; Registrierung an der lokalen Fritz!Box
register=>asterisk625:passwort@192.168.178.1/asterisk625

; 625 Fritzbox internes Netzwerk
[asterisk625]
type=friend
```

```

; Verweis auf den Abschnitt in der extensions.conf, der für das Callhandling
; zuständig ist
context=default

; IP der Fritz!Box
host=192.168.178.1

; muß bei einer Fritz!Box so angegeben werden
fromdomain=fritz.box
fromuser=asterisk625
defaultuser=asterisk625
authuser=asterisk625

; das Passwort muß bei neueren Fritz!Boxen mindesten 8 Zeichen lang sein
secret=geheim
callerid="Asterisk-Fritz-625"
directmedia=no
mailbox=625

; DL1HRC/2620055 via SIP
[2620055]
type=friend

; wenn der Host keine feste IP-Adresse besitzt
host=dynamic

; Name, der bei der Gegenstelle angezeigt wird
callerid="Adi/DL1HRC" <2620055>

; Login und Passwort für den Zugang zum Sip-Server
defaultuser=2620055
secret=ganz_geheim

; weitere Sicherheisteinstellung
insecure=invite,port
directmedia=no

; Abschnitt in der extensions.conf, der den Anruf „bearbeitet“
context=internal-sip

```

Nachfolgend ein Konfigurationsbeispiel für den Fall, dass Sie zwei Asterisk-Instanzen per IAX2-Protokoll verbinden möchten (/etc/asterisk/iax.conf):

```

[general]
language=de
bandwidth=high
allow=all
authdebug=yes
autokill=yes
maxcallnumbers=16386
auth=md5
qualify=yes
requirecalltoken=no
nat=yes

; Beispieleintrag für ein Relais, hier DM0KWD
[262002]
type=friend
trunk=yes

```

```
; IP-Adresse für eine VPN-Verbindung zwischen den einzelnen Nodes
host=10.8.0.28

; DMR-ID als Username
username=262002
secret=secret

; Context „IAX“ innerhalb der extensions.conf, der das Callhandling übernimmt
context=IAX
host=dynamic
```

In den Extensions (*/etc/asterisk/extensions.conf*) wird die Rufbehandlung, der sogenannte Dialplan konfiguriert [12] und [13].

```
[general]
static=yes
autofallthrough=yes
writeprotect=no
clearglobalvars=no
autofallthrough=yes

[globals]
CONSOLE=Console/default
RINGTIME=40
PAGING_HEADER=Intercom

;*****
;* incoming call from IAX-connections
;*****
[IAX]

; Anwahl 2629 -> DM0KWD -> Conference(123)
exten => 2629,1,Log(NOTICE,${EXTEN} Incoming call via IAX (2629) from $
${CALLERID(num)})
exten => 2629,2,Dial(IAX2/262009/${EXTEN},,tT)
exten => 2629,n,Hangup()

; Anwahl z.B. 2621234 -> Weiterleitung an SIP und IAX -> Conference(123)
exten => _2[46][023]XXX.,1,Log(NOTICE,${EXTEN} Incoming call via [IAX]
(_26[23]XXX.) from ${CALLERID(num)})
exten => _2[46][023]XXX.,2,Dial(SIP/${EXTEN}/${EXTEN}&IAX2/${EXTEN}/${
${EXTEN}},90,tT)
exten => _2[46][023]XXX.,n,Hangup()
```

Der folgende Abschnitt behandelt einen eingehenden Anruf aus dem Festnetz. Wird eine berechnete Telefonnummer erkannt so erfolgt ein direktes Durchschalten auf den Funkkanal (Relais, Node). Der Abschnitt [checknumber] enthält eine Liste von berechtigten Telefonnummern, so wie diese vom Festnetz aus übertragen wird. Im Allgemeinen beginnen diese z.B. mit 0176.... und nicht mit +49. Soweit meine Erfahrungen, updates sind willkommen!

Wird die gesamte Liste der berechtigten Rufnummern durchlaufen ohne Return(), dann erfolgt als letzte Handlung die Abfrage einer Pin-Nummer. Dazu ist eine Pinnummer (mindestens 4stellig) in der Zeile *exten => _0X.,3,Authenticate(Pin-Nummer)* einzutragen.

```
[default]
```

```

exten => asterisk625,1,Log(NOTICE,Incoming call exten [default] asterisk625 from
${CALLERID(all)} ${CALLERID(name)})
exten => asterisk625,2,Set(foo=${SIP_HEADER(From)})
exten => asterisk625,3,Set(foo=${CUT(foo,@,1)})
exten => asterisk625,4,Set(foo=${CUT(foo,:,2)})
exten => asterisk625,5,Log(NOTICE,Incoming call from ${foo} to ${EXTEN})
exten => asterisk625,6,Answer()
exten => asterisk625,7,Gosub(checknumber,${foo},1)
exten => asterisk625,8,Playback(access-granted)
exten => asterisk625,9,ConfBridge(123)
exten => asterisk625,n,Hangup()

[checknumber]
exten => _X.,1,Log(NOTICE,checking number ${EXTEN})
exten => 0555123123,1,Return()
exten => 0173qqqqqq,1,Return()
exten => 0174xxxxxx,1,Return()
exten => 0175yyyyyy,1,Return()
exten => _0X.,2,Log(NOTICE,Authenticate?)
exten => _0X.,3,Authenticate(Pin-Nummer)
exten => _0X.,n,Return()

```

Hier die berechtigten Rufnummern eintragen für die automatische Rufannahme

3.3. Sicherheitshinweise

Achten Sie darauf, dass Ihr Asterisk-Server und auch Ihre SipLogic nicht von unberechtigten Personen genutzt werden können. Sie werden feststellen, dass bereits nach sehr kurzer Zeit Anmelde- und Angriffsversuche stattfinden.

Hier die wichtigsten Hinweise ohne Anspruch auf Vollständigkeit:

- Installieren und konfigurieren Sie fail2ban, siehe [20]
- konfigurieren Sie allowguest=no, allowoverlap=no, allowtransfer=no, allowsubscribe=no und alwaysauthreject=yes in der /etc/asterisk/sip.conf
- Konfigurieren Sie sichere Passworte für die SIP-User (abc123 ist KEIN sicheres Passwort)
- Lassen Sie die interne HAMRADIO-Kommunikation idealerweise über ein eigenes VPN laufen, soweit möglich
- kontrollieren Sie von Zeit zu Zeit die Logfiles (normalerweise in /var/log/asterisk/)

3.4. Konfiguration einer Fritz!Box als Netzzugang zum HAMRADIO-Sip-Netzwerk

Die eigene Fritz!Box kann als Zugang zum Hamradio-Netzwerk verwendet werden um eine Verbindung Repeater ↔ eigenes Telefonnetz herzustellen. Es muß nur sichergestellt sein, dass beim Anruf eine Person mit Amateurfunklizenz den Anruf annimmt oder in das Amateurfunknetzwerk ausführt.

1) Eigene DMR-ID als Telefongerät anlegen und einem internen Telefon zuweisen

Fritz!Box 7360 adimedia

Angemeldet | [FRITZ!Box](#) | [FRITZ!NAS](#) | [MyFRITZ!](#) | [?](#)

Übersicht
Internet
Telefonie
 Anrufe
 Anrufbeantworter
 Telefonbuch
 Weckruf
 Fax
 Rufbehandlung
 Telefoniegeräte
Eigene Rufnummern
 Heimnetz
 WLAN
 DECT
 Diagnose
 System

Eigene Rufnummern

Rufnummern | Anschlüsseinstellungen | Sprachübertragung

Auf dieser Seite können Sie Ihre eigenen Rufnummern einrichten und bearbeiten.

Status	Rufnummer	Anschluss	Anbieter	Vorauswahl		
●	0346226110	Internet	Telekom	*121#		
●	0346226111	Internet	Telekom	*122#		
●	0346226112	Internet	Telekom	*123#		
●	1010	Internet	FRITZ!Box 7360 (Internet-Telefonie)	*127#		

Liste drucken | Neue Rufnummer

Ansicht: Erweitert | Inhalt | Handbuch | Tipps&Tricks | Newsletter | avm.de

Bild 5: Konfiguration einer neuen Rufnummer: „Neue Rufnummer“ anklicken

2) Internet-Rufnummer anlegen und bei svxlink.ham-radio-op.net anmelden (mit den zugewiesenen Zugangsdaten)

Fritz!Box 7360 adimedia

Angemeldet | [FRITZ!Box](#) | [FRITZ!NAS](#) | [MyFRITZ!](#) | [?](#)

Internetrufnummer eintragen

Wählen Sie Ihren Internettelefonie-Anbieter aus und tragen Sie die Ihnen mitgeteilten Anmeldedaten ein.

Telefonie-Anbieter: anderer Anbieter

Rufnummer für die Anmeldung*: Interne Rufnummer in der Fritz!Box*:

Weitere Rufnummer

***Rufnummer für die Anmeldung**
 Geben Sie in dieser Spalte bitte die Rufnummer für die Anmeldung ein. Diese Rufnummer haben Sie von Ihrem Anbieter bekommen. Sie kann je nach Anbieter unterschiedlich benannt sein. Bitte geben Sie die Rufnummer genau so ein, wie vom Anbieter vorgegeben, einschließlich eventuell enthaltener Sonderzeichen.

***Interne Rufnummer in der Fritz!Box**
 Geben Sie nun bitte Ihre Rufnummer ohne Vorwahl und ohne Sonderzeichen ein.

Weitere Rufnummer
 Über "Weitere Rufnummer" können Sie hier weitere Rufnummern anlegen, wenn diese dieselben Zugangsdaten (Benutzername und Kennwort) wie die erste Rufnummer haben. Rufnummern mit abweichenden Zugangsdaten können Sie später unter "Eigene Rufnummern" über die Schaltfläche "Neue Rufnummer" einrichten.

Zugangsdaten

Benutzername:
 Kennwort:
 Registrar:
 Proxy-Server:

Leistungsmerkmale

DTMF-Übertragung: Automatisch

☒ Anmeldung immer über eine Internetverbindung
 Falls Ihr Internetanbieter die separate Internettelefonie-Verbindung für eigene Rufnummern reserviert, aktivieren Sie diese Option, wenn es sich um eine Rufnummer eines anderen Anbieters handelt.

☒ Ortsvorwahl für ausgehende Gespräche einfügen

Weiter | Abbrechen

Ansicht: Erweitert | Inhalt | Handbuch | Tipps&Tricks | Newsletter | avm.de

Bild 6: Eingabe der eigenen DMR-ID als Benutzername, das Passwort und der Servername in der Form „Servername:Port“

3) Die erhaltenen Zugangsdaten eintragen, dann auf „Weiter“ klicken

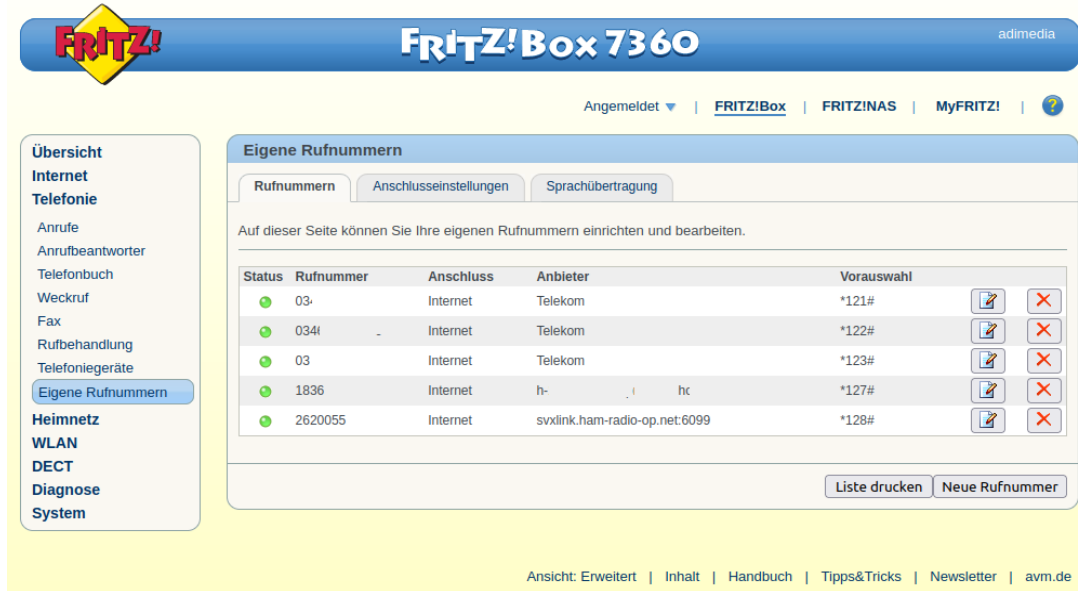


Bild 7: Erfolgreiche Anmeldung der Fritz!Box beim SIP-Server

Die grüne „LED“ links in der Liste zeigt an, dass sich die Fritz!Box erfolgreich mit dem SIP-Server verbunden hat.

3.5. Konfiguration von SIP-Clients

Installieren Sie ZoiPer auf Ihrem Handy (Android, Iphone). ZoiPer ist für einen einfachen Sip-Zugang kostenlos. Für weitere Accounts (Sip oder IAX) ist ein einmaliger Betrag fällig.

3.5.1. ZoiPer konfigurieren

Einstellungen → Konten → Konto hinzufügen

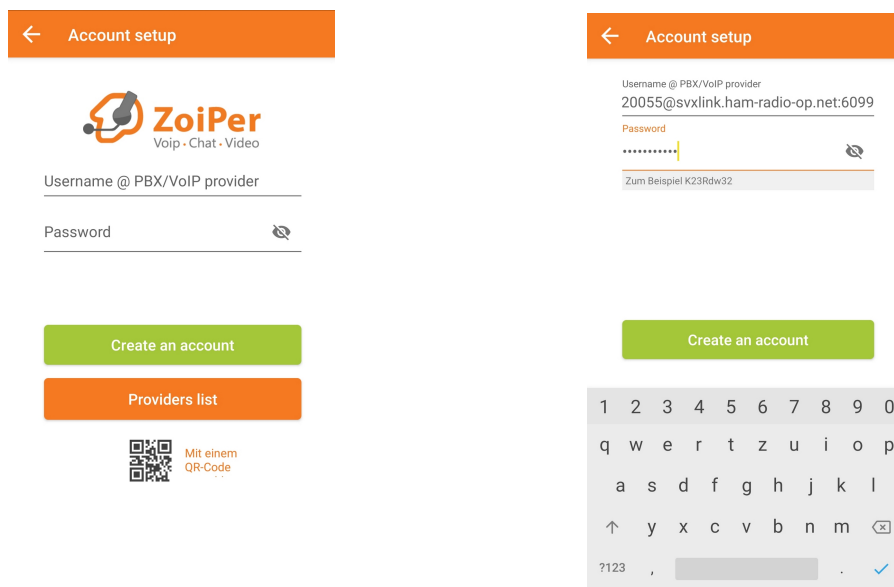


Bild 8: Zugangsdaten/SIP-Server in der Form Username@SIP-server:Port eingeben. Wichtig ist der abweichende UDP-Port 6099!

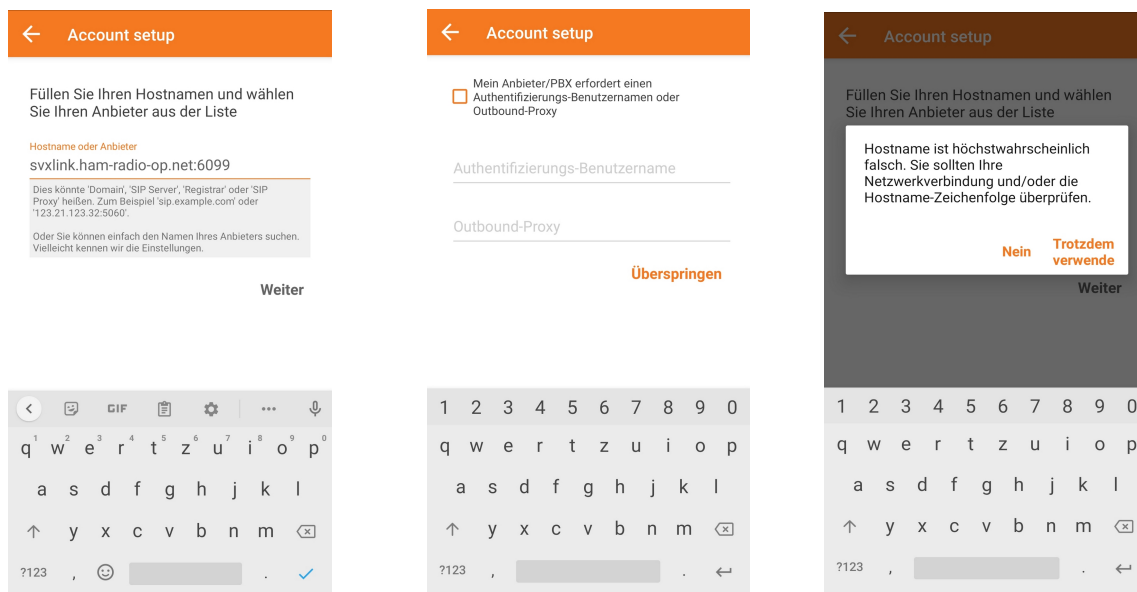


Bild 9: Weiterführende Konfigurationen („Überspringen“ bzw. „Trotzdem verwenden“)

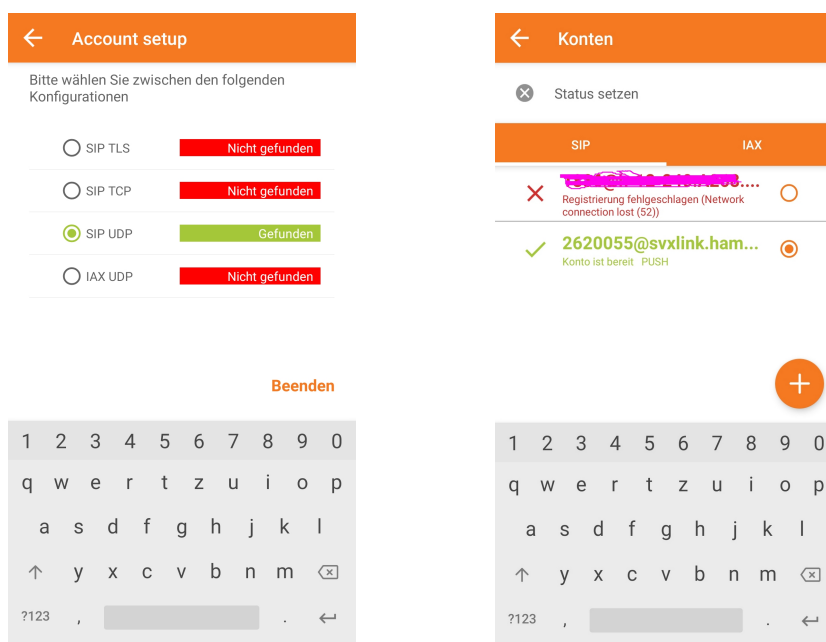


Bild 10: Erfolgreiche Anmeldung am SIP-Server (hier grün dargestellt)

Der Zoiper ist auch in der Lage sich per IAX-Protokoll direkt mit einem Asterisk zu verbinden, hierzu ist aber ein bestimmter Betrag für die App zu zahlen.

3.5.2. Einrichtung von Twinkle auf einem PC/Laptop

Die Installation von Twinkle unter Ubuntu ist unter [9] beschrieben. Beim erstmaligen Start wird man durch die Einrichtung geführt.

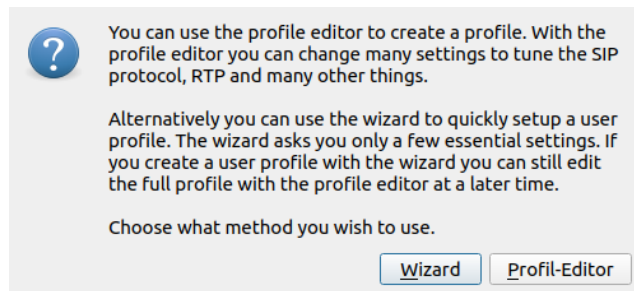


Bild 11: Auswahl zwischen Wizard oder Profileditor

Geben Sie einen Profilnamen ein, HAMRADIO ist sicher passend für diesen Anwendungsfall.

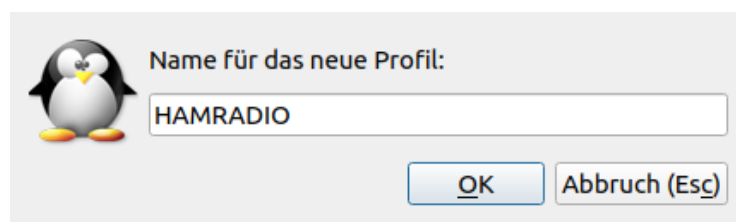


Bild 12: Einen sprechenden Namen für das Profil wählen

Im zweiten Schritt geben Sie bitte die Zugangsdaten ein, die Sie auf Anfrage gern erhalten.

SIP Service Provider (Umschalt-F1 für Hilfe):	Anderer
Ihr Absendername:	Adi / DL1HRC
Nutzername *:	26200551
Domain*:	svxlink.ham-radio-op.net:6099
Anmeldename:	26200551
Passwort:	
SIP-Proxy:	
STUN-Server:	
OK Abbruch (Esc)	

Bild 13: Login, Passwort und Servereinstellungen

Das nächste Fenster einfach per OK bestätigen.

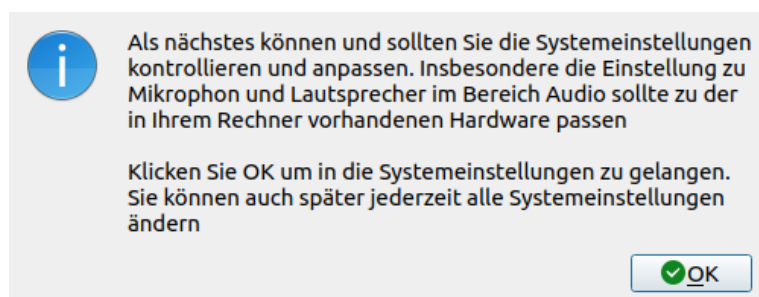


Bild 14: Diesen Hinweis mit OK bestätigen

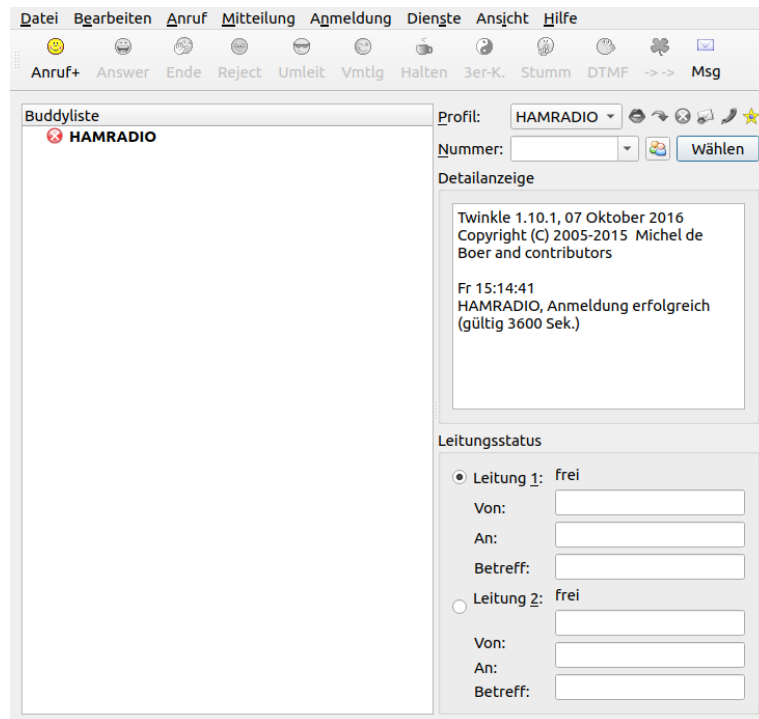


Bild 15: Twinkle mit erfolgreicher Anmeldung am SIP-Server

Die erfolgreiche Anmeldung wird im rechten Fenster (Bild 15) angezeigt: „Anmeldung erfolgreich“

3.5.3. Konfiguration von Sipnatic

Die Sipnatic-App steht für Android und Iphone zur Verfügung. Informationen findet man hier [35]. Weitere Informationen folgen in einer der nächsten Versionen dieser Beschreibung.

3.6. AsteriskLogic über Alsa-Loopback und Asterisk-Konsole

Es gibt eine weitere Möglichkeit mit einigen Abstrichen seinen SvxFLink-Node an ein SIP-Telefonnetz anzudocken. Genutzt wird dabei das sogenannte Alsa-Loopback-Device und die Asterisk-Konsole mit einem call-File als Rufaufbau. Asterisk und SvxFLink müssen dabei zwingend auf einem Gerät installiert sein. Die Kopplung des Audios erfolgt dabei über die Asterisk-Konsolenfunktion + Loopback + SvxFLink-AsteriskLogic. Die Asterisk-Logic ist dabei eigentlich eine RepeaterLogic, bei der Rx und Tx auf die Alsa-Loopback's umgeleitet werden. Das Linux-Modul snd-aloop muß per modprobe geladen werden (besser fester Eintrag in der `/etc/modules-load.d/modules.conf`). Weiterhin empfiehlt sich die Soundkarten-Nummerierung z.B. über die `/etc/modprobe/alsa-base.conf` festzulegen, so dass diese nach dem Neustart immer die selben Indexe besitzen.

Vorteile gegenüber der Verwendung des pjSipLogic-Banches:

- eine neuere Standard-SvxFLink-Version ist ausreichend
- kein Kompillieren erforderlich

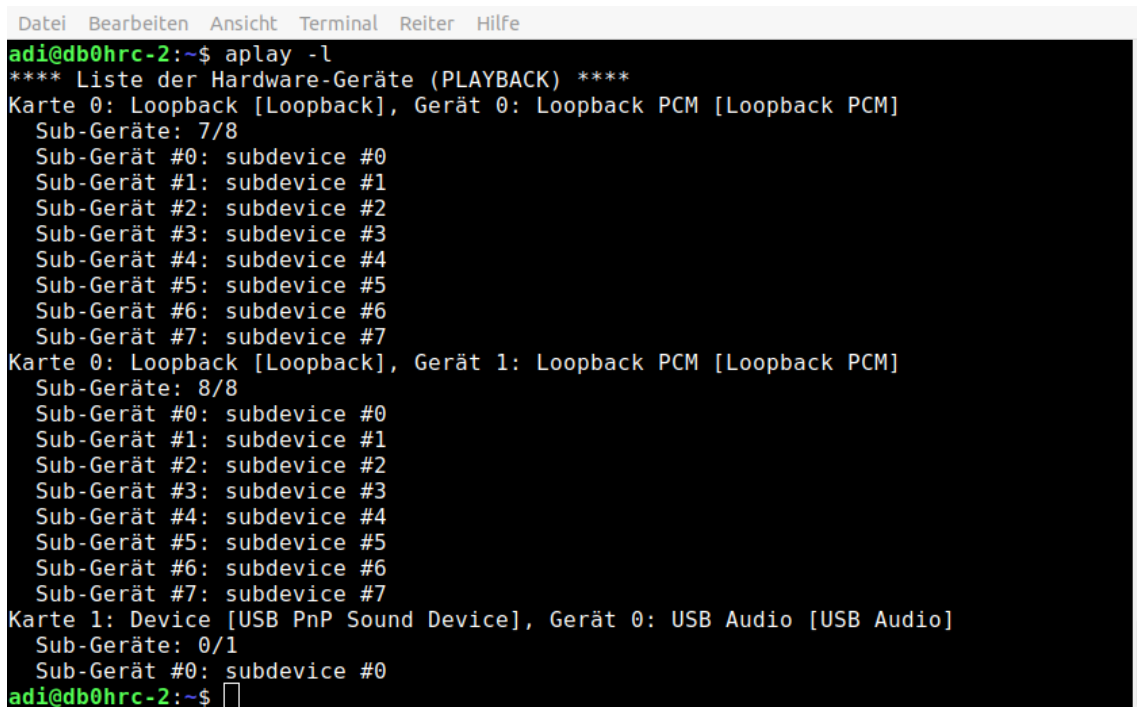
Nachteile:

- nur VOX-Funktion für den SIP-Audiostream möglich
- Asterisk-Instanz auf der selben Hardware notwendig
- das Handling ist insgesamt etwas umständlich

Um die ALSA-loopback devices zu erzeugen, müssen Sie das Kernel-Modul snd-aloop laden.

```
$ sudo modprobe snd-aloop
```

Dieses Modul erzeugt zwei Geräte (0 und 1) und Untergeräte (subdevices 0-7) als Aufnahmegerät (capture device) und Wiedergabegerät (playback device). Mit dem Befehl `aplay -l` werden diese neuen loopback-devices aufgelistet.



```
adi@db0hrc-2:~$ aplay -l
**** Liste der Hardware-Geräte (PLAYBACK) ****
Karte 0: Loopback [Loopback], Gerät 0: Loopback PCM [Loopback PCM]
  Sub-Geräte: 7/8
    Sub-Gerät #0: subdevice #0
    Sub-Gerät #1: subdevice #1
    Sub-Gerät #2: subdevice #2
    Sub-Gerät #3: subdevice #3
    Sub-Gerät #4: subdevice #4
    Sub-Gerät #5: subdevice #5
    Sub-Gerät #6: subdevice #6
    Sub-Gerät #7: subdevice #7
Karte 0: Loopback [Loopback], Gerät 1: Loopback PCM [Loopback PCM]
  Sub-Geräte: 8/8
    Sub-Gerät #0: subdevice #0
    Sub-Gerät #1: subdevice #1
    Sub-Gerät #2: subdevice #2
    Sub-Gerät #3: subdevice #3
    Sub-Gerät #4: subdevice #4
    Sub-Gerät #5: subdevice #5
    Sub-Gerät #6: subdevice #6
    Sub-Gerät #7: subdevice #7
Karte 1: Device [USB PnP Sound Device], Gerät 0: USB Audio [USB Audio]
  Sub-Geräte: 0/1
    Sub-Gerät #0: subdevice #0
adi@db0hrc-2:~$
```

Bild 16: Ausgabe von „aplay -l“

Angesprochen werden die Geräte von SvxLink und Asterisk über die Device-Parameter

`alsa:plughw:0,1,1` bzw. `alsa:plughw:0,1,0`

und deren korrespondierenden Parameter

`hw:0,0,1` bzw. `hw:0,0,0`

Hier ein Beispiel für eine `svxlink.conf` (nur die wichtigsten Abschnitte und Parameter sind aufgeführt):

```
[GLOBAL]
LOGICS=RepeaterLogic,AsteriskLogic
LINKS=AsteriskLink

[AsteriskLink]
NAME=Asterisk
CONNECT_LOGICS=AsteriskLogic,RepeaterLogic
DEFAULT_ACTIVE=1
```

```

[RepeaterLogic]
...

[AsteriskLogic]
TYPE=Repeater
RX=RxAsterisk
TX=TxAsterisk
OPEN_ON_SQL=40
OPEN_ON_DTMF=*
DEFAULT_LANG=de_DE
OPEN_SQL_FLANK=OPEN
SHORT_IDENT_INTERVAL=30
LONG_IDENT_INTERVAL=60
RGR_SOUND_DELAY=0
IDLE_TIMEOUT=6

# NO_REPEAT=1 muss konfiguriert sein, sonst hört sich der SIP-Anrufer
# selbst
NO_REPEAT=1
CALLSIGN=DB0ABC
EVENT_HANDLER=/usr/share/svxlink/events.tcl
DEFAULT_LANG=de_DE
STATE_PTY=/tmp/state_pty_asterisk
DTMF_CTRL_PTY=/tmp/dtmf_AsteriskLogic

[RxAsterisk]
RX_ID=3
TYPE=Local
AUDIO_DEV=alsa:plughw:0,1,1
AUDIO_CHANNEL=0
SQL_DET=VOX
SQL_START_DELAY=0
SQL_DELAY=0

# für die SQL_HANGTIME haben sich Werte von 1400-2000 (ms) bewährt
SQL_HANGTIME=2000
VOX_FILTER_DEPTH=20
VOX_THRESH=1000
DEEMPHASIS=0
DTMF_DEC_TYPE=INTERNAL
DTMF_MUTING=1
DTMF_HANGTIME=100
DTMF_MAX_FWD_TWIST=18
DTMF_MAX_REV_TWIST=12

[TxAsterisk]
TYPE=Local
AUDIO_DEV=alsa:plughw:0,1,0
AUDIO_CHANNEL=0
PTT_TYPE=NONE
TIMEOUT=9000
TX_DELAY=0
PREEMPHASIS=0

Im Asterisk muß nun das Alsa-Modul aktiviert und eine entsprechende Konfiguration
ausgeführt werden. Für weitere Erklärungen siehe [27].

/etc/asterisk/alsa.conf:

[general]

```

```

autoanswer=yes
context=default
extension=alsa
language=de
input_device=hw:0,0,0
output_device=hw:0,0,1

```

Die beiden Parameter `input_device` und `output_device` korrespondieren damit mit den in der `svxlink.conf` (Abschnitt `[AsteriskLogic]`) konfigurierten Alsa-Devices.

<u>/etc/svxlink/svxlink.conf</u>		<u>/etc/asterisk/alsa.conf</u>
<code>[RxAsterisk]</code> <code>AUDIO_DEV=alsa:plughw:0,1,1</code>	←	<code>[general]</code> <code>output_device=hw:0,0,1</code>
<code>[TxAsterisk]</code> <code>AUDIO_DEV=alsa:plughw:0,1,0</code>	→	<code>input_device=hw:0,0,0</code>

Das ein- und ausgehende Audio wird also sozusagen über die Alsa-Loopback Devices „durchgereicht“.

Weiterhin ist die Asterisk-Konsole erforderlich deren Konfiguration über die Datei `/etc/asterisk/console.conf` erfolgt:

```

[general]

[default]
input_device = default
output_device = default

; autoanswer muss = yes sein, damit der „Ruf“ aus der SvxLink-Welt automatisch
; angenommen wird
autoanswer = yes
context = default
extension = alsa
callerid = "Radio Console DB0YYY" <700>
language = de
overridecontext = no
active = yes

```

Es muß jetzt noch eine call-Datei angelegt werden, welche einen Rufaufbau zur Konsole realisiert, diese muß in das Verzeichnis `/var/spool/asterisk/outgoing` kopiert werden. Sie ist sozusagen der Start des Rufaufbaus:

/var/spool/asterisk/console.call:

```

Channel: Console/default
CallerID: "Console" <700>
Extension: s
Context: dconsole
MaxRetries: 2
RetryTime: 60
WaitTime: 30
Priority: 2

```

Nach der Beendigung von Asterisk (Neustart, Update etc.) muß dieser wieder neu aufgebaut werden. Kopieren Sie per `crond` diese Datei regelmäßig in das o.g. Verzeichnis:

/var/spool/cron/crontabs/asterisk:

```
*/1 * * * * /home/asterisk/make_call.sh 1>/dev/null 2>/dev/null
```

Die Datei *make_call.sh* befindet sich im Asterisk-Home-Verzeichnis und wird dort ausgeführt, alle Dateien müssen dem User *asterisk* gehören. Hier zeigt sich auch ein kleines Problem dieser Lösung: die Verbindung HF ↔ SIP wird erst dann hergestellt, wenn die call-Datei in das outgoing-Verzeichnis kopiert wird.

/home/asterisk/make_call.sh:

```
#!/bin/bash
if [ ! -f /var/spool/asterisk/outgoing/console.call ]; then
  cp /var/spool/asterisk/console.call /var/spool/asterisk/outgoing/
fi
```

Wenn Asterisk eine *.call-Datei im outgoing-Verzeichnis erkennt, so erfolgt ein automatisches Anwählen entsprechend der darin konfigurierten Parameter.

Jetzt fehlt noch der Dialplan, der in der */etc/asterisk/extensions.conf* definiert wird:

```
[globals]
CONSOLE=console/default

[dconsole]
exten => s,1,Log(NOTICE,Connect console to radio via loopback,extend ${EXTEN})
exten => s,2,Answer
exten => s,n,ConfBridge(123)
```

Es wird hierbei also automatisiert mit der extension „dconsole“ verbunden, damit befindet sich der Repeater/Node im Konferenzraum 123 des Asterisk-Servers. Was man damit macht ist jedem OM selbst überlassen.

3.7. Anbindung eines Hytera-Repeaters RDxxx an das SIP-Netzwerk

Die Hytera-Repeater RDxxx unterstützen von Hause aus die Anbindung an ein SIP-Netzwerk. Allerdings ist meiner Meinung nach die Implementierung nicht sehr gut.

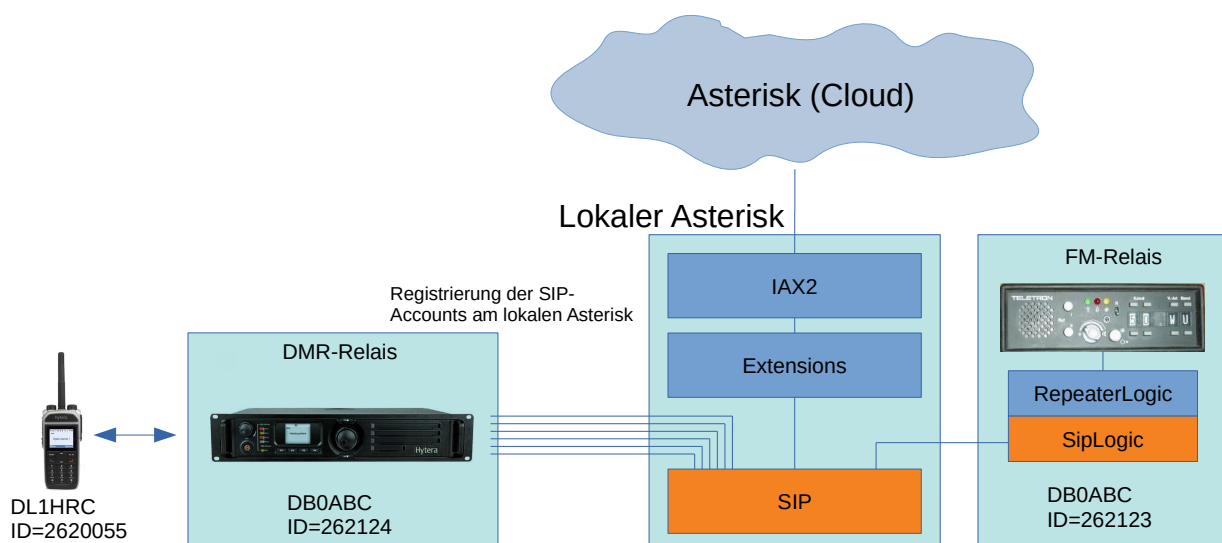


Bild 17: Prinzipaufbau Netzwerk aus DMR-Relais ↔ Asterisk ↔ FM-Relais

Auch ist es mir bisher trotz umfangreicher Tests nicht gelungen einen mobilen Teilnehmer vom SIP per Funk anzurufen. Im Codeplug muß für jeden SIP-Teilnehmer ein eigener Eintrag vorhanden sein.

Bild 17 zeigt den prinzipiellen Aufbau einen kleinen SIP-Netzwerkes, dass ein DMR-Relais über einen eigenen Asterisk-Server mit einem FM-Relais (SvxLink mit pjSipLogic-Branch) verbindet. Für die ordnungsgemäße Funktionalität sind auf allen Seiten einige Vorarbeiten erforderlich.

a) Konfiguration des Hytera-Repeater:

RD985 → Conventional → Phone → Phone System → Phone System 1:

IP-Adresse Ihres Asterisk-Servers

Passwort mit dem sich jeder registrierter User am Asterisk anmeldet

Roger-Beep wird generiert wenn der Funkuser die PTT losläßt

Kommandos für Anwahl und Abwahl der Sip-Verbindung

Bild 18: Parameter für SIP

	4	2620000	2620000	Slot2	Private Call
	5	2620000	2620000	Slot2	Private Call
	6	2620000	2620000	Slot2	Private Call
	7	2620000	2620000	Slot2	Private Call
	8	2620004	2620004	Slot2	Private Call
	9	2620002	2620002	Slot2	Group Call
	10	2620	2620	Slot2	Group Call

Bild 19: Konfigurierte Sip-Accounts/Users im Codeplug des Hytera-Repeater

Der Anmeldename am Asterisk-Server ergibt sich dabei aus der Kombination folgender Parameter:

call type	- Art des Rufes	1 = privater Ruf oder 2 = Gruppenruf
timeslot	- Zeitschlitz	1 = Zeitschlitz 1 oder 2 = Zeitschlitz 2
DMR ID	- die DMR-ID des jeweiligen Users. z.B. 2620055	

Beispiel: 122620055

2620055 → DMR-ID

2 → TS 2

1 → private call

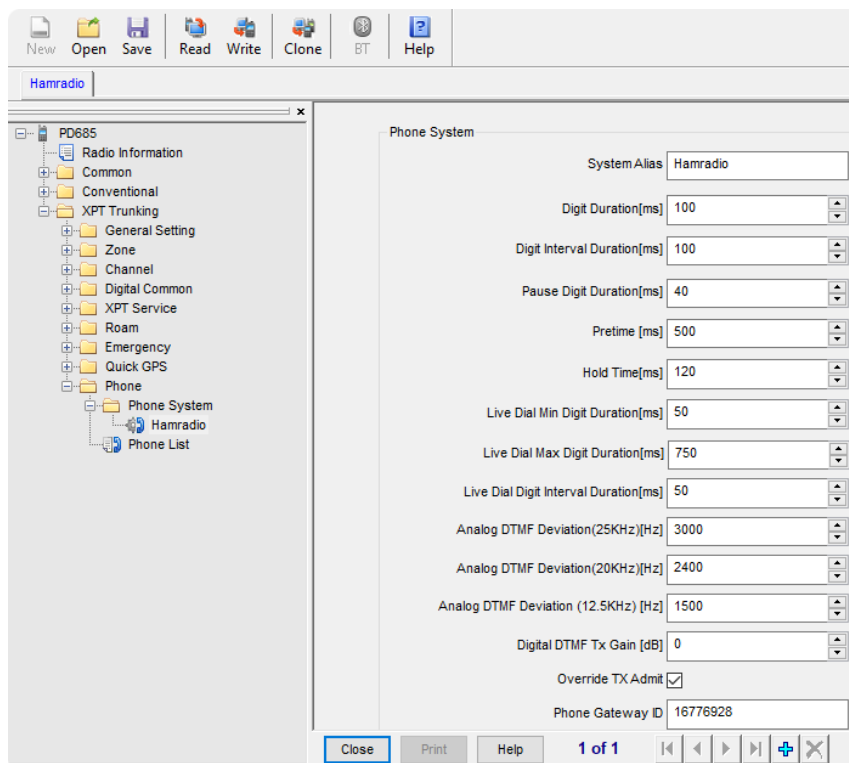


Bild 20: Weitere Parameter für SIP-Einwahl per Hytera-Repeater

b) Konfiguration eines mobilen Endgerätes (z.B. PD685)

In der CPS wählen Sie: *PD685* → *XPT-Trunking* → *Phone* → *Phone System*

Vergeben Sie am besten einen sprechenden Namen (z.B. „Hamradio“). Beachten Sie außerdem, dass die Phone-Gateway-ID in den jeweiligen Codeplugs übereinstimmen müssen (hier im Beispiel 16776928). Die im Bild 19 gezeigten Parameter sind bei mir im Einsatz, sie erheben keinen Anspruch auf Richtigkeit. Möglicherweise ergeben sich noch bessere Einstellungen.

Weiterführende Informationen zum Thema kann man unter [44] und [45] finden.

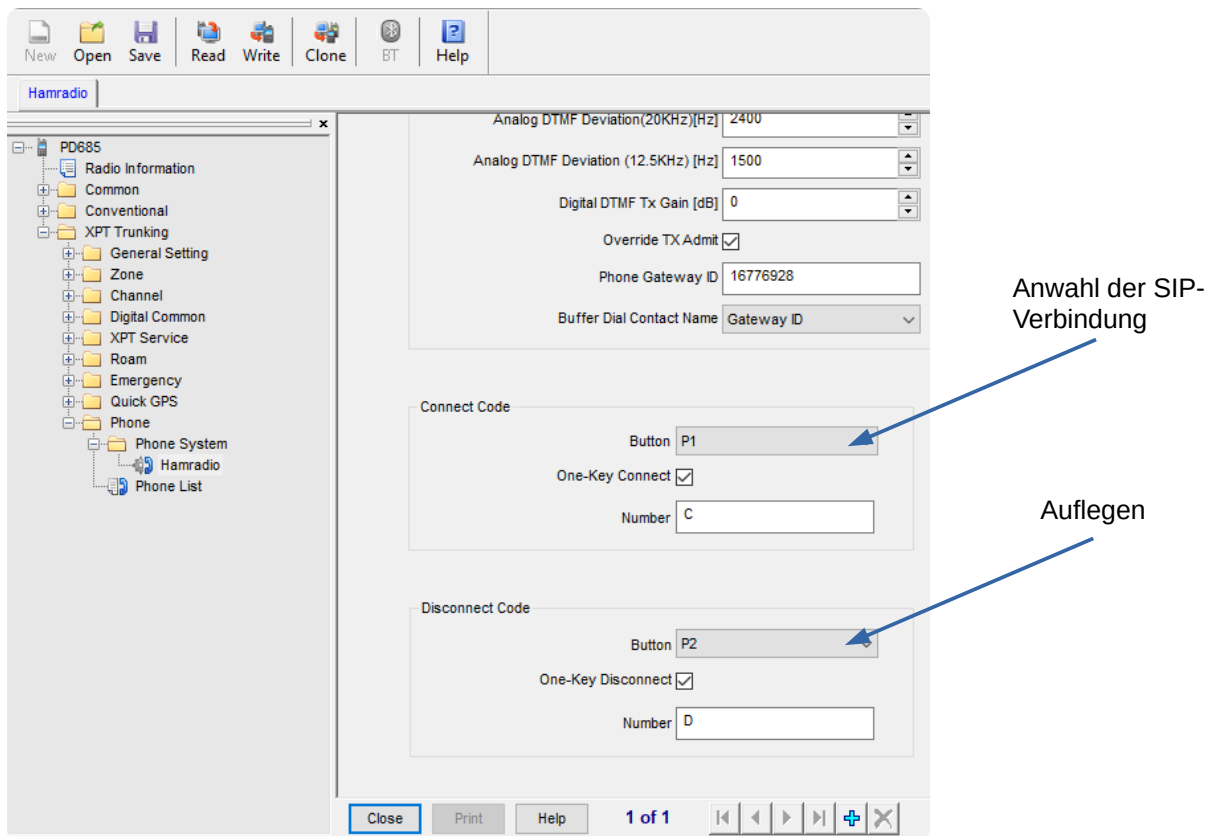


Bild 21: Parameter für Anwahl und Auflegen der Verbindung im mobilen Endgerät

c) Einrichtung des eigenen Asterisk-Servers als Vermittlungsstelle zwischen DMR- und FM-Relais
Auf dem FM-Relais muß ein Sipfähiger SvxFLink-Branh installiert und lauffähig sein

Konfiguration der sip.conf:

```
[222620055]
type=friend
callerid="Adi/DL1HRC" <2620055>
host=dynamic
defaultuser=222620055
fromuser=222620055
context=dmr
secret=pass123
directmedia=no
```

Konfiguration der extensions.conf, im Beispiel wird ein eingehender Anruf vom DMR-Relais an einen SIP-Kontakt weitergeleitet:

```
[dmr]
; DMR-repeater und user über Hytera [12][12]262|3xxx
exten => _[12][12]26[23][089]XX.,1,Log(NOTICE,Incoming call context dmr to
Hytera: SIP/${EXTEN} - ${CALLERID(all)} ${CALLERID(name)})
exten => _[12][12]26[23][089]XX.,2,Dial(SIP/${EXTEN},,tT)
exten => _[12][12]26[23][089]XX.,n,Hangup()
```

d) Konfiguration eines FM-Relais mit pjSipLogic-Branh

Die Konfiguration einer SipLogic bzw. des pjSip-Branches ist im Abschnitt 3.1. beschrieben.

e) Aufbau einer SIP-Verbindung über den DMR-Repeater zu einem mobilen Endgerät (z.B. PD685):

Menü → Telefon → Telefonliste → einen Eintrag wählen → PTT betätigen

oder
Menü → Telefon → Manuelle Wahl → Zielnummer eingeben → PTT betätigen

Auflegen: Rote Hörertaste betätigen

3.8. Einrichtung eines Zuganges

Für interessierte OM's kann ein Zugang zu meinem kleinen SIP-Netz zur Verfügung gestellt werden. Anfragen bitte an Adi / DL1HRC (dl1hrc@gmx.de). Voraussetzung ist das Vorhandensein einer DMR-ID und natürlich einer gültigen Amateurfunklizenz. Die DMR-ID gilt hierbei als Telefonnummer unter der man erreichbar sein wird.

Bitte die folgenden Daten angeben:

- Rufzeichen
- Typ (Repeater, Userzugang,...)
- DMR-ID
- Name des OM
- optional: Locator, Stadt, Kommentar

Unter http://svxlink.ham-radio-op.net/cgi-bin/show_users.pl hat man einen kleinen Überblick über bereits registrierte Nutzer. Wenn ich Zeit finde werde ich ein kleines Portal einrichten an dem man sich anmelden kann. Mal schauen....

4. USRP-Erweiterung (Branch svxlink-usrp)

Mit der USRP-Erweiterung ist es möglich SvxFLink mit einem DigitalVoice-Netzwerk zu verbinden.

Beachten Sie, dass für den Betrieb ein Transcoder erforderlich ist, der den analogen Audiostream in einen DV (DMR, Dstar, P25,...) transcodiert. Das kann per DV3k-Stick und dem AMBEServer oder dem md380-emu erfolgen. Für den AMBEServer benötigen Sie noch einen DV3000-USB-Stick, den sie z.B. bei der Firma Wimo erwerben können [32].

Bild 22 zeigt den Audio-Signalfluß sowie die zu konfigurierenden Parameter zwischen SvxFLink ↔ USRP ↔ DMR- bzw. DV-Netzwerk. Ein Analog_Reflector kann weggelassen bzw. übersprungen werden.

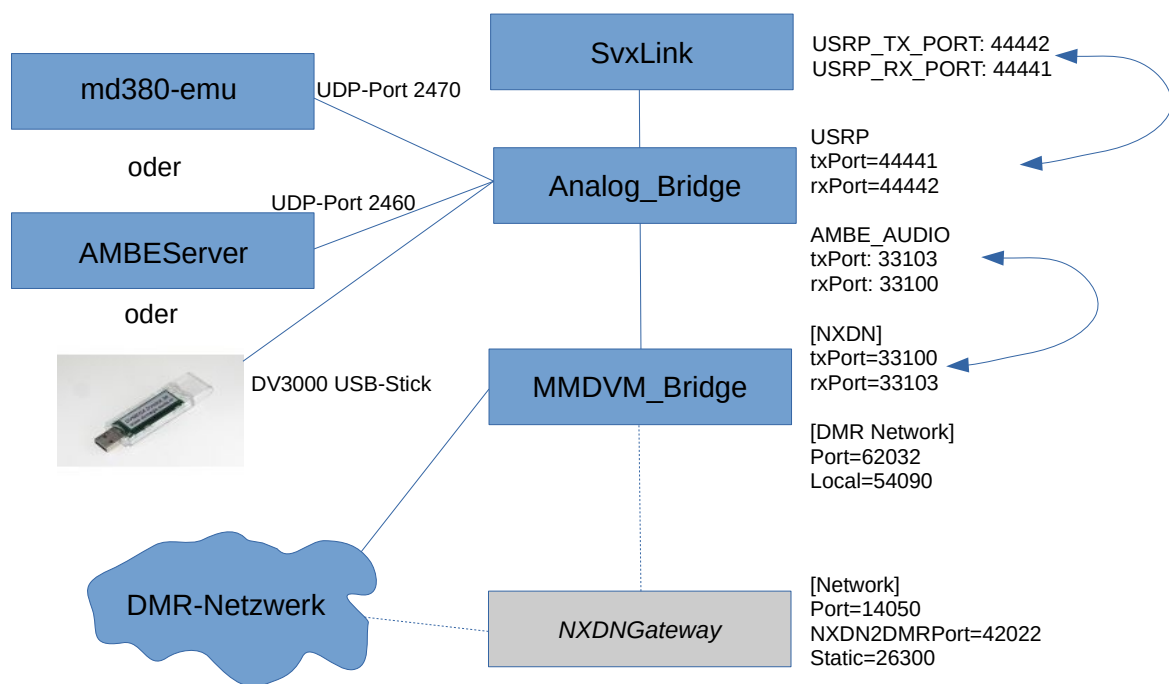


Bild 22: Übersicht über die prinzipielle Portkonfiguration und den Datenfluß zwischen den Applikationen

Hinweis zu NXDN und Anbindungen an andere Netzwerke (z.B. Brandmeister):

Ich rate dringend davon ab, sich mit diesen Netzwerken zu verbinden. Bei einem Test bekam ich eine bitterböse eMail. Was mich bei meinen Versuchen sehr geärgert hatte war, dass praktisch ohne weitere Informationen die Sip-Zugänge innerhalb des Brandmeister-Netzwerkes abgeschaltet wurden obwohl dort ständig mehr als 70 OM's qrv waren. Auf Rückfrage bekam ich die lapidare Antwort (sinngemäß): "Is halt so". Wer Interesse an Experimenten hat, dem empfehle ich die Anbindung an FreeDMR [21]. Es gibt in DL bereits zwei FreeDMR-Server (ID=2621 und 2622), siehe [22].

Zukünftig werde ich auf Anfrage für OM's einen SIP-Zugang bereitstellen, damit sie sich mit den im Netz befindlichen Nodes per DMR-ID als Telefonnummer verbinden können. Im HAMNET ist das ja schon längere Zeit verfügbar. Allerdings befinden sich in meinem Einzugsbereich kaum HAMNET-Zugänge so dass ich das leider per „normalem“ Internet nutzen muß.

Um diesen Branch sinnvoll zu betreiben ist die Installation der MMDVM-Umgebung notwendig.

Sie benötigen:

- MMDVM_Bridge [28]
- Analog_Bridge [29]
- ThumbDV 3000 USB-Stick [32] oder md380-emu [31] (eventuell AMBEServer [30])

Im Bild 23 sind die erforderlichen Installationsschritte dargestellt.

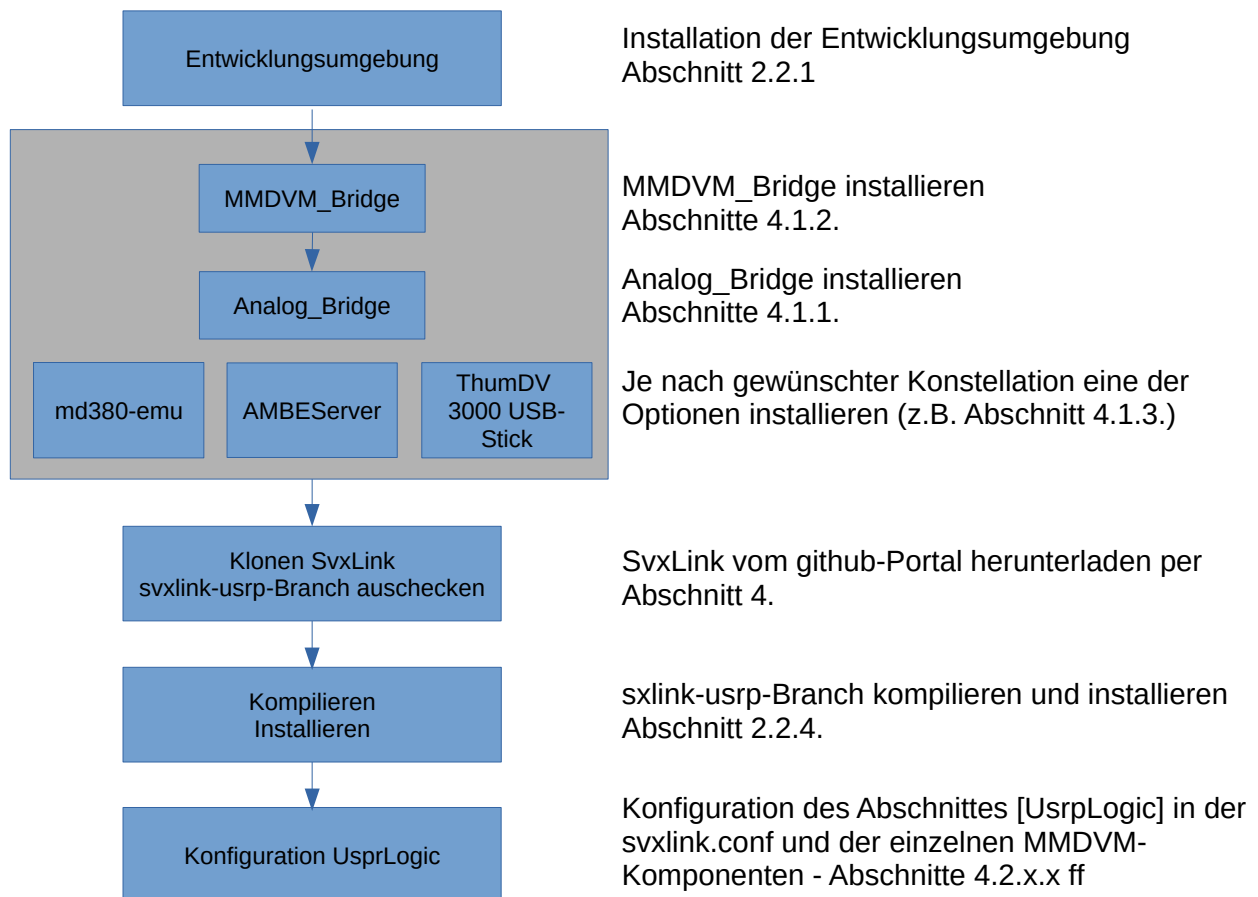


Bild 23: Notwendige Installationsschritte zur Installation des svxlink-usrp-Branches

4.1. Installation der MMDVM-Umgebung und des AMBEServers

4.1.1. Analog_Bridge

Die Analog_Bridge und MMDVM_Bridge stehen leider nur als Binaries auf github.com zum Download zur Verfügung.

Der Download der Analog_Bridge erfolgt per git in das /opt-Verzeichnis:

```
cd /opt
git clone https://github.com/DVSwitch/Analog\_Bridge
```

unter `/opt/Analog_Bridge/bin` stehen Binaries für verschiedene Hardware-Umgebungen zur Verfügung. Sie sollten entweder das passende Binary nach `Analog_Bridge` umbenennen oder einen Symlink anlegen. Für den RaspberryPi geht das wie folgt:

```
ln -s /opt/Analog_Bridge/bin/Analog_Bridge.armhf  
/opt/Analog_Bridge/bin/Analog_Bridge
```

Kopieren Sie dann das service-File nach `/usr/lib/systemd/user`:

```
sudo cp /opt/Analog_Bridge/systemd/analog_bridge.service /usr/lib/systemd/user/
```

und ermöglichen Sie den automatischen Start der `Analog_Bridge`. Um die `Analog_Bridge` automatisch zu starten muß dieser Service enabled werden:

```
sudo systemctl enable analog_bridge
```

Jetzt wird bei jedem Boot-Vorgang oder Absturz dieser Service neu gestartet.

4.1.2. MMDVM_Bridge

Die `Analog_Bridge` und `MMDVM_Bridge` stehen leider nur als Binaries auf github.com zum Download zur Verfügung.

Der Download der `MMDVM_Bridge` erfolgt per git in das `/opt`-Verzeichnis:

```
cd /opt  
git clone https://github.com/DVSwitch/MMDVM\_Bridge
```

unter `/opt/MMDVM_Bridge/bin` stehen Binaries für verschiedene Hardware-Umgebungen zur Verfügung. Sie sollten entweder das passende Binary nach `MMDVM_Bridge` umbenennen oder einen Symlink anlegen. Für den RaspberryPi geht das wie folgt:

```
sudo ln -s /opt/MMDVM_Bridge/bin/MMDVM_Bridge.armhf \  
/opt/Analog_Bridge/bin/MMDVM_Bridge
```

Kopieren Sie dann das service-File nach `/usr/lib/systemd/user`:

```
sudo cp /opt/MMDVM_Bridge/systemd/mmdvm_bridge.service /usr/lib/systemd/user/
```

Dieses File steuert den automatischen Start per System-Daemon. Um die `MMDVM_Bridge` automatisch zu starten muß dieser Service enabled werden:

```
sudo systemctl enable mmdvm_bridge
```

Jetzt wird bei jedem Boot-Vorgang oder Absturz dieser Service neu gestartet.

4.1.3. AMBEServer

Der `AMBEServer` stellt den En-/Decoding des Audiostreams von Analog nach DV und zurück über das Netzwerk zur Verfügung, standardmäßig lauscht dieser Dienst auf dem Port 2460/UDP. Der `AMBEServer` muß eigentlich nur installiert werden, wenn Sie den ThumbDV3000-USB-Stick auf einer anderen Hardware betreiben wollen. Der `AMBEServer` ist Teil des OpenDV-Frameworks [41].

4.2. Konfiguration der MMDVM-Umgebung

4.2.1. Analog_Bridge

Hier die Konfigurationsdatei Analog_Bridge.ini als Beispielkonfiguration. Je nach verwendetem Vocoder muß die Konfiguration entsprechend angepasst werden:

```
include = dvsm.macro

; General Section describes settings for Analog_Bridge itself.
[GENERAL]

; Show messages and above 0=No logging, 1=Debug, 2=Message, 3=Info, 4=Warning,
; 5=Error, 6=Fatal
logLevel = 2

; Export metadata to USRP partner (transcode setups require this)
exportMetadata = true

; Export database files to USRP partner
transferRootDir = /tmp

; DMR ID to callsign lookup data
subscriberFile = /var/lib/dvswitch/subscriber_ids.csv

; General vocoder setup information
; Allow software AMBE decoding if a hardware decoder is not found
decoderFallback = true

; Use the MD380 AMBE emulator for AMBE72 (DMR/YSFN/NXDN)
useEmulator = true

; IP address and port of the md380 server
emulatorAddress = 127.0.0.1:2470

; UDP port to send to the WebProxy
pcmPort = 2222

[AMBE_AUDIO]
; IP address of xx_Bridge
address = 127.0.0.1

; Transmit TLV frames to partner on this port
txPort = 33103

; Listen for TLV frames from partner on this port
rxPort = 33100

; DMR, DMR_IPSC, DSTAR, NXDN, P25, YSFN, YSFW (encode PCM to this format)
ambeMode = DMR

; Analog -> Digital Minimum time in MS for hang delay (0-10000)
minTxTimeMS = 2500

; ID to use when transmitting from Analog_Bridge 7 digit ID
gatewayDmrId = 2620055

; ID of source repeater 7 digit ID plus 2 digit SSID
repeaterID = 262005589
```

```

; TG to use for all frames sent from Analog_Bridge -> xx_Bridge
txTg = 2629

; Slot to use for frames sent from Analog_Bridge -> xx_Bridge
txTs = 2

; Color Code to assign DMR frames
colorCode = 1

[USRP]
; IP address of USRP partner
address = 127.0.0.1

; (Allstar/Asterisk or another Analog_Bridge)
; Transmit USRP frames on this port
txPort = 44441

; Listen for USRP frames on this port
rxPort = 44442

; Digital -> Analog (AUDIO_UNITY, AUDIO_USE_GAIN, AUDIO_USE_AGC)
usrpAudio = AUDIO_USE_AGC

; Gain factor when usrpAudio = AUDIO_USE_GAIN (0.0 to 5.0)  (1.0 = AUDIO_UNITY)
usrpGain = 1.10

; Set the agc threshold (db), slope (db) and decay (ms)
usrpAGC = -26,10,100

; Analog -> Digital (AUDIO_UNITY, AUDIO_USE_GAIN, AUDIO_BPF)
tlvAudio = AUDIO_GAIN

; Gain factor when tlvAudio = AUDIO_USE_GAIN (0.0 to 5.0)  (1.0 = AUDIO_UNITY)
tlvGain = 0.35

[DV3000]
; IP address of AMBEServer
address = 127.0.0.1

; Port of AMBEServer
rxPort = 2470

#address = /dev/ttyUSB0 ; Device of DV3000U on this machine
baud = 460800           ; Baud rate of the dongle (230400 or 460800)
serial = false          ; Use serial=true for direct connect or
                        ; serial=false for AMBEServer

```

Der automatische Start der Analog_Bridge und die Überwachung übernimmt der System-Daemon und erfolgt über diesen Befehl:

```
sudo systemctl enable analog_bridge
```

Damit ist gewährleistet, dass die Analog-Bridge bei jedem Bootvorgang und bei einem eventuellen Crash gestartet wird.

4.2.1.1. ThumbDV 3000-USB-Stick

Der ThumbDV 3000 USB-Stick (bzw. NWDR ThumbDV) ist ein hardwarebasierter Vocoder. Die Ansteuerung des Sticks kann entweder direkt von der Analog_Bridge erfolgen oder per AMBEServer, siehe Abschnitt 4.2.3.

Der ThumbDV 3000-Stick muß beim direkten Zugriff durch die Analog_Bridge an einem USB-Port des selben Gerätes stecken. Nachfolgen der entsprechende Abschnitt innerhalb der Analog_bridge.ini für den direkten Betrieb per ThumbDV 3000:

```
[GENERAL]
useEmulator = false

[DV3000]
address = /dev/ttyUSB0
baud = 460800
serial = true
```

4.2.1.2. Konfiguration md380-emu

Um den md380-emu nutzen zu können ist in der Analog_Bridge.ini der Parameter useEmulator auf true zu setzen und IP-Adresse und Port festzulegen unter der Verbindung zum md380-emu-daemon aufgebaut werden soll:

```
[GENERAL]
useEmulator = true
emulatorAddress = 127.0.0.1:2470
```

Das sind die Start-Optionen für den md380-emu:

```
Usage: /opt/md380-emu/md380-emu [OPTION]
  -d          Decodes AMBE
  -e          Encodes AMBE
  -V          Version Info
  -h          Help!
  -v          Verbose mode.
  -vv         Very verbose!
  -i foo      Input File
  -o bar      Output File
```

Der Aufruf auf der Kommandozeile erfolgt wie folgt:

```
md380-emu -S 2470
```

Um dem md380-emu permanent auszuführen (automatischer Start nach dem Boot-Vorgang) sollte ein entsprechender Service eingerichtet werden. Dazu ist eine md380-emu.service Datei zu erstellen und der Service per

```
sudo systemctl enable md380-emu
```

einzurichten. Hier ein Beispiel für die md380-emu.service-Datei:

```
[Unit]
Description=MD-380 Emulator Service
# Description=Place this file in /lib/systemd/system
# Description=N4IRS 10/04/2020
```

```
[Service]
Type=simple
Restart=always
RestartSec=3
StandardOutput=null
WorkingDirectory=/opt/md380-emu
ExecStartPre = /bin/sh -c 'echo "Starting md380-emu: [`date +%T.%N`]" >>
/var/log/netcheck'
ExecStart=/usr/bin/qemu-arm-static /opt/md380-emu/md380-emu -S 2470
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process

[Install]
WantedBy=multi-user.target
```

4.2.1.3. AMBEServer und ThumbDV 3000 USB-Stick

Standardmäßig lauscht der AMBEServer auf dem UDP-Port 2460, dies kann natürlich bei Bedarf auch geändert werden. Für weitere Informationen siehe [30].

```
[GENERAL]
useEmulator = false
```

```
[DV3000]
address = 127.0.0.1
rxPort = 2460
serial = false
```

Das sind die Optionen für den AMBEServer:

```
Usage: AMBEServer
  -d          im daemon mode starten (im Hintergrund)
  -r
  -s speed    set speed 460800 or 230400 baud
  -p port     UDPPort (Standard 2460)
  -i tty      Port, z.B. /dev/ttyUSB0
  -v          Verbose output
  -x          hexadezimale Ausgabe der Frames
```

Der Aufruf auf der Kommandozeile erfolgt wie folgt, z.B.:

```
AMBEServer -s 460800 -p 2460 -i /dev/ttyUSB0
```

4.2.2. MMDVM_Bridge

Nachfolgend eine MMDVM_Bridge.ini als Beispielkonfiguration, für weitere Informationen schauen Sie auf der Webseite des MMDVM-Frameworks nach

```
[General]
Callsign=DB0ABC
Id=2621234
Timeout=240
Duplex=0

[Info]
RXFrequency = 438687500
TXFrequency = 431087500
Power=15
```

```
Latitude=51.123456
Longitude=11.98765
Height=40
Location=Testburg, J011ZZ
Description=Test-Gate
URL=www.svxlink.org
```

[Log]

```
# Logging levels: 0=No logging, 1=Debug, 2=Message, 3=Info, 4=Warning,
# 5=Error, 6=Fatal
DisplayLevel=1
FileLevel=1
FilePath=/var/log/mmdvm
FileRoot=MMDVM_Bridge
```

[DMR Id Lookup]

```
File=/var/lib/mmdvm/DMRIds.dat
Time=24
```

[DMR]

```
Enable=1
ColorCode=1
EmbeddedLCOnly=1
DumpTADData=0
```

[DMR Network]

```
Enable=1
Address=svxreflector.org
Port=62031
Jitter=360
Local=62032
Password=passw0rd
# for DMR+ see
#https://github.com/DVSwitch/MMDVM\_Bridge/blob/master/DOC/DMRplus\_startup\_options.md
# for XLX the syntax is: Options=XLX:4009
# Options=
Slot1=0
Slot2=1
Debug=1
```

Der automatische Start der MMDVM_Bridge und die Überwachung übernimmt der System-Daemon und erfolgt über diesen Befehl:

```
sudo systemctl enable mmdvm_bridge
```

Damit ist gewährleistet, dass die MMDVM-Bridge bei jedem Bootvorgang und bei einem eventuellen Crash gestartet wird.

4.3. SvxLink-Abschnitt [UsrpLogic]

Der Abschnitt [UsrpLogic] ist für die Verbindung zwischen SvxLink und der Analog_Bridge zuständig. Usp-Support und damit der Abschnitt [UsrpLogic] ist nur verfügbar wenn Sie den usrp-svxlink-Branch auf Ihrem System installiert haben.

```

[UsrpLogic]
TYPE=Usrp

# IP-Adresse oder Hostname auf dem die Analog_Bridge bzw. der Analog_Reflector
# läuft
USRP_HOST=127.0.0.1

# UDP-Verbindung zur Analog_Bridge rxPort=44442
USRP_TX_PORT=44442

# UDP-Verbindung zur Analog_Bridge rxPort=44441
USRP_RX_PORT=44441

CALL=DL1HRC
DMRID=2620055
RPTID=262005511
DEFAULT_CC=1
DEFAULT_TG=2629
DEFAULT_TS=2

# die folgenden Parameter sind default-Einstellungen, sie müssen auf Ihrem
# System angepasst werden!
# Verstärkungsfaktor für Audio aus dem analogen System ins DV-Netz
PREAMP=-12

# Audiofilter für aus dem DV-Netz eingehendes Audio
#FILTER_FROM_USRP=HsBq2/0.01/-18/4000

# Audiofilter für aus dem analogen Netz ausgehendes Audio ins DV-Netz
FILTER_TO_USRP=BpBu1/650-3800

# Verstärkungsfaktor für Audio aus dem DV-Netz zum analogen System
NET_PREAMP=6
JITTER_BUFFER_DELAY=100

# der TCL-Eventhandler
EVENT_HANDLER=/usr/share/svxlink/events.tcl

```

Vergessen Sie bitte nicht einen Abschnitt LINK unter [GLOBAL] zu definieren, damit die Logiken audiomäßig miteinander verbunden werden. Eine einzelne [UsrpLogic] erfüllt keinen Zweck, sie muß mit mindestens mit einer weiteren Logic verLINKt werden!

5. Tetra-Erweiterung (Branch tetra-contrib)

5.1. Vorbereitungen

Dieser Branch ermöglicht die Anbindung eines mobilen Tetra-Endgerätes (MTM800E, MTM5400/5500) an SvxFLink. Für die Ansteuerung ist ein modifiziertes digitales Datenkabel notwendig, welches die Verbindung zwischen PC/Raspi und MTMxxx über die PEI ermöglicht. Über dieses aktive Datenkabel erfolgt die Kommunikation inklusive Squelch-Kriterium und PTT-Steuerung. Auf <https://vktetra.com> ist ein einfacher Selbstbau-PEI-Adapter beschrieben [11].

Eine „abgespeckte“ Variante wäre die altbekannte Ansteuerung mittels GPIO über entsprechende PC/Raspi-Ports oder mittels USB-2-TTL-Adapter. Damit können aber keine SDSen ausgewertet oder gesendet werden und viele erweiterte Funktionen stehen nicht zur Verfügung. Es soll daher nicht näher beschrieben werden.

Getestet wurden die Funktionen mit einem Datenkabel GMKN1022A und PMKN4105A.



Bild 24: Aktives Datenkabel für einen RS232-Anschluß

Es ist möglich, dass es auch mit anderen Kabeln funktioniert, dafür kann aber keine Garantie übernommen werden. Diese Kabel sind um die entsprechenden Audioverbindungen zu ergänzen damit ein- und ausgehende NF vom Funkgerät an die Soundkarte des RaspberryPi/PCs gelangen kann.

5.2. Konfiguration der PEI

5.2.1. PEI-Hardware

Tabelle 1: Pinout eines MTM5x00, die fett hervorgehobenen Anschlüsse müssen am Datenkabel als PINs manuell nachgerüstet werden.

Pin	Funktion	Bedeutung
1	UART1_TXD / USBx_D+	USB 1.1 – Standard-Host RS232 oder UART2 – Alternative Einstellung
2	UART1_RXD / USBx_D-	
3	UART1_RTS / USBx_VBUS	
4	GND_USBx	
5	Einadrig	1-adriger Standardanschluss (über 2,2k auf 5V)
6	KEYFAIL/FLASH	Schlüsselladung (über 10K auf 5V) Flash-Eingang (>10V löst Flashmode aus)

7	SWB +	A+-Spannung (begrenzt auf 14V) mit 1A-Strombegrenzung
8	GND_MAIN	Haupt- und Leistungsmasse
9	LAUTSPRECHER-	Lautsprecher (PA) Ausgang - (KEINE ERDUNG!)
10	LAUTSPRECHER+	Lautsprecher (PA) Ausgang +
11	TX_AUDIO	TX-Audioeingang
12	GND_ANALOG	Hauptaudiomasse
13	MIC1 / EXT_MIC	Ext. Mikrofoneingang/MIC1 für Geräuschminimierung dualer Mikrofoneingang
14	RX_AUDIO	RX-Audioausgang
15	MIC2	MIC2 für Geräuschminimierung dualer Mikrofoneingang
16	GND_MIC	Masse (für MIC)
17	EXTERNAL_PTT	PTT-Eingang (über 4,7k auf 5V)
18	UART2_DTR/USBy_ID	RS232 oder UART1/UART2 DTR/ 2. USB 2.0
19	HOOK_PA_EN	HOOK_PA_EN Eingang (oder der programmierbare 5V-GPIO)
20	UART2_TXD/USBy_TX	RS232 oder UART2 TXD/2. USB 2.0 (OTG) D+
21	UART2_RTS/USBy_VBUS	RS232 oder UART2 RTS/2. USB 2.0 (OTG) VBUS- 100 mA
22	UART2_RXD/USBy_RX	RS232 oder UART2 TXD/2. USB 2.0 (OTG) D-
23	Notruf	Eingang für Durchsagen in Notfällen (gezogen über 24,9k auf A+) - zum Einschalten auf Low ziehen
24	UART_CTS	RS232 oder UART1/UART2 CTS-Eingang
25	Zündung	Zündungseingang (über Serie 15K): zum Einschalten > 5V ziehen
26	EXTERNER ALARM	Externer Alarmeingang (gezogen über 4,7k auf A+)

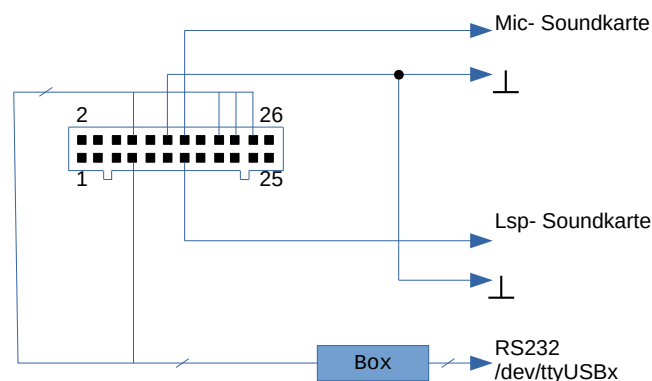


Bild 25: erweiterter Adapter (digitales Datenkabel) mit Audio-Erweiterungen



Bild 26: Ein um die Audioanschlüsse (Tx+Rx-Audio) erweitertes aktives Datenkabel, die C's können auch weggelassen werden

Wie schon erwähnt: Voraussetzung ist das Vorhandensein des tetra-contrib-Branches auf dem System. Um den externen Line-IN Anschluß nutzen zu können muß im Codeplug ein externes Erweiterungskit konfiguriert werden, da sonst das Audio nicht an die entsprechenden PEI-PINs des MTMs geleitet werden.

5.2.2. PEI-Einstellungen in der CPS

In der CPS erfolgt die Parametrierung der PEI-Schnittstelle über den Menüpunkt „Codeplug → Data Services → PEI Parameters“. Empfohlen werden folgende Einstellungen:

- Baud Rate = 115200
- Parity Bit = PARITY_NONE
- Flow Control = No flow control
- Stop Bits = 1
- Data Bits = 8

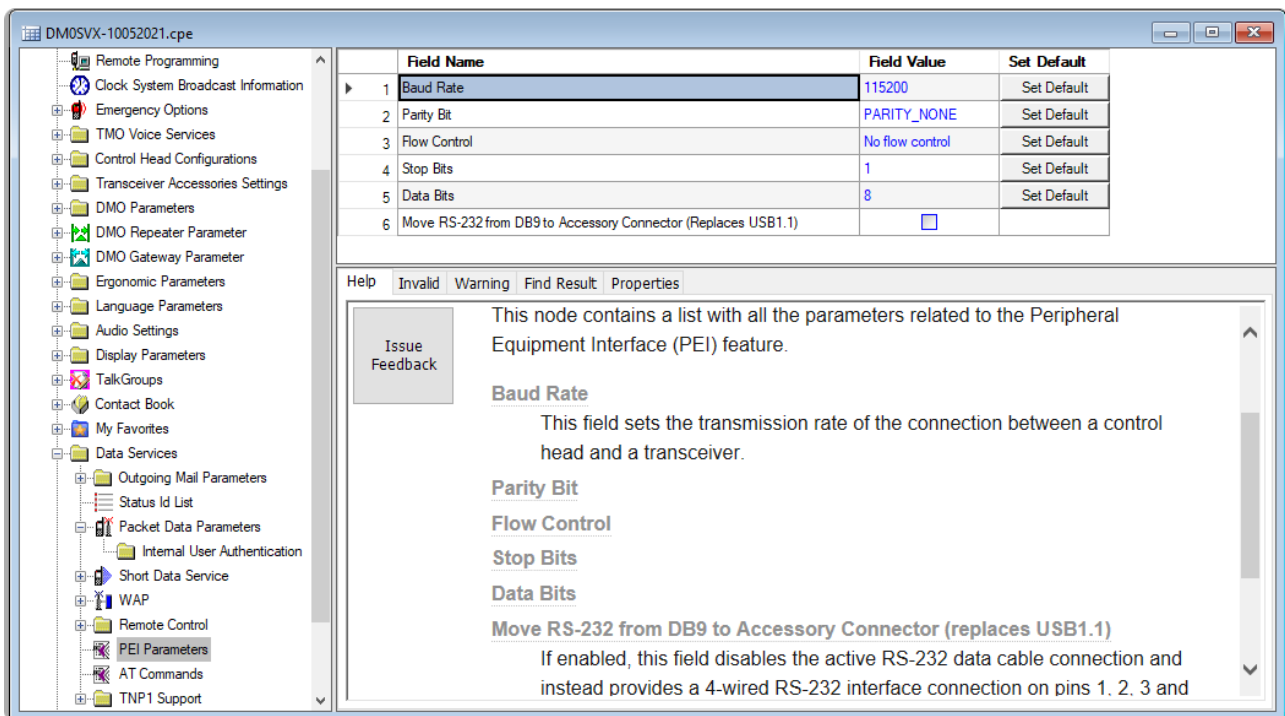


Bild 27: Parametereinstellungen für die PEI (115k2, 8N1)

5.2.3. Test der PEI-Konfiguration

Testen Sie die Verbindung zum MTM zunächst mit dem minicom oder einem anderen Terminalprogramm. Konfigurieren Sie das Terminalprogramm mit den im MTM zugewiesenen Baudraten.

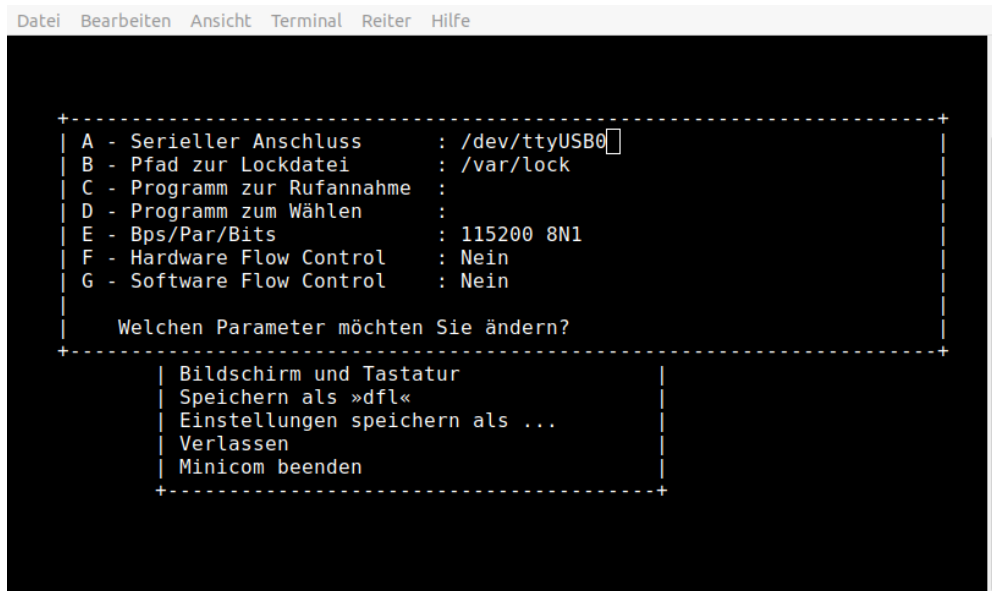


Bild 28: Konfiguration von minicom

Das Senden eines einfachen [ENTER]s wird mit dem „+CME ERROR: 35“ quittiert, das Senden des Befehls „AT\r\n“ vom Gerät mit „OK“.

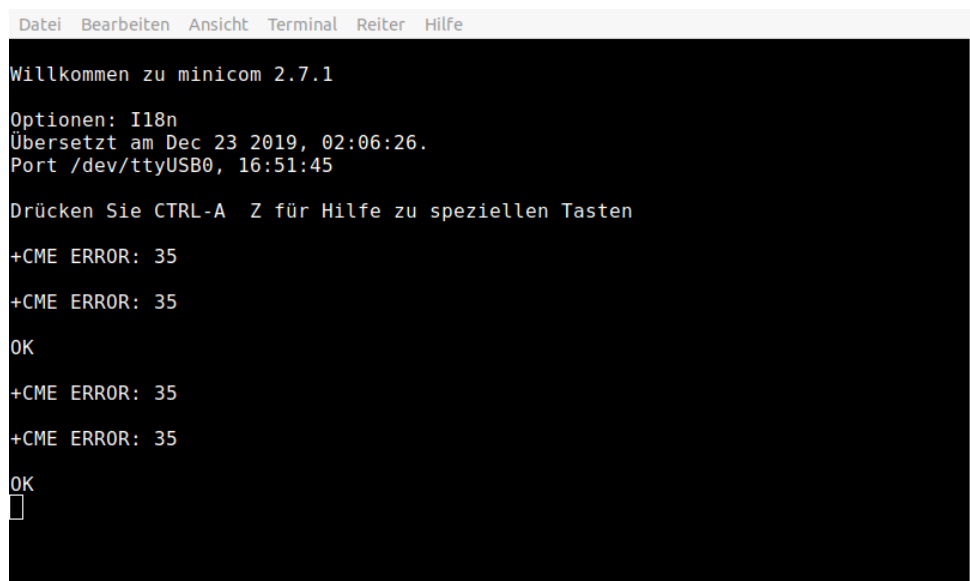


Bild 29: Kommunikation zwischen PC und Tetra-Endgerät (MTMxxx), die ERROR-Meldungen kommen beim Drücken der ENTER-Taste, das OK bei der Eingabe von AT+ENTER

In der CPS (für Motorola-Geräte) sollten Sie die PEI wie in Bild 28 konfigurieren. Wenn die Kommunikation gestört wird prüfen Sie bitte unbedingt ob möglicherweise die HF des Gerätes in die serielle Verkabelung einstreut. Eine fehlerhafte Kommunikation äußert sich in häufigen ERROR-Meldungen im svxlink-Log durch unvollständig empfangene AT-Antworten des MRT/HRT's. Eine HF-technisch guter Aufbau ist für den Betrieb essentiell.

5.3. Konfiguration der svxlink.conf (Abschnitt [TetraLogic])

Der Abschnitt TetraLogic konfiguriert den Datenfluß zwischen dem MTMxxx und SvxLink. Beim tetra-contrib-Branch wird ein neuer, speziell für die PEI-Kommunikation entwickelter

Rauschsperrtyp TETRA_SQL verwendet. Das Squelchkriterium wird aus den PEI-Antworten des MRT's generiert.

Auf Senden geschaltet wird nicht über einen GPIO-Pin (PTT-Eingang) sondern per ATD-Befehl über die PEI.

Hier der Abschnitt [TetraLogic] der für die Logik verantwortlich ist, es sind wie immer nur die wichtigsten Parameter dargestellt.

```
[TetraLogic]
TYPE=Tetra
RX=Rx1
TX=Tx1
MODULES=ModuleHelp,ModuleParrot,ModuleTclVoiceMail
CALLSIGN=DL1HRC
DEFAULT_LANG=de_DE

# Serielle Schnittstelle ↔ PEI
PORT=/dev/ttyUSB0
BAUD=115200

# Debuglevel 0-wenig Ausgaben, 4-viele Ausgaben (verbose)
DEBUG=4

# die Daten für MCC, MNC und ISSI sollten mit den im MTM übereinstimmen
ISSI=9999
GSSI=1
MNC=16383
MCC=901

# Initialisierung der PEI des MRT/HRT
INIT_PEI=AT+CTOM=1;AT+CTSP=1,3,131;AT+CTSP=1,3,130;AT+CTSP=1,3,138;AT+CTSP=1,2,2
0;AT+CTSP=2,0,0;AT+CTSP=1,3,3;AT+CTSP=1,3,10;AT+CTSP=1,1,11;AT+CTSP=1,3,138;AT+C
TSP=1,3,12;AT+CTSP=1,3,9;AT+CTSDC=0,0,0,1,1,0,1,1,0,0,0

# der Parameter APRSPATH sollte auskommentiert sein, wenn Sie sich nicht sicher
# sind was Sie dort konfigurieren wollen
APRSPATH=APRS,qAR,DL1HRC-10

# das Pseudo-tty über das Kommandos wie z.B. eine SDS gesendet werden können,
# das ist eigentlich nur interessant wenn Sie eigene Skripte oder Programme für
# entsprechende Funktionalitäten nutzen möchten
SDS_PTY=/tmp/tetra_pty

# Digits, die über das Pseudo-tty gesendet werden [0-9A-D*#] werden als DTMF-
# Eingaben gewertet und entsprechend konfigurierte Funktionalitäten gestartet
DTMF_CTRL_PTY=/tmp/tetra_dtmf_ctrl

# Annäherungs-Warnung – wenn die Entfernung (in km) zur eigenen Station
# unterschritten wird erhält die entsprechende Gegenstation eine SDS
#PROXIMITY_WARNING=3.1

TIME_BETWEEN_SDS=3600

# Default-Text, der an die Stationen gesendet wird wenn diese sich erstmalig im
# Funkkreis „anmelden“
INFO_SDS=Herzlich Willkommen

# veraltet, nicht mehr nutzen: verweist auf den Abschnitt [Tetra_Users]
# innerhalb der svxlink.conf, in der einige Userdaten der Funkkreisteilnehmer
```

```

# gespeichert sind
TETRA_USERS=Tetra_Users

# NEU: File im json-Format, das die Userdaten beinhaltet, Ersatz für den
# Abschnitt [Tetra_Users]
TETRA_USER_INFOFILE=/etc/svxlink/tetra_users.json

# Status einer teilnehmenden Station, sendet diese eine Staus-SDS an den
# SvxLinkNode an den das SvxLink-System angeschlossen ist (im Allgemeinen
# das MRT/HRT mit SvxLink-Anbindung, das die ISSI 9999 besitzt), so
# wird der aktuelle Status von SvxLink per Aprs-Message verteilt
TETRA_STATUS=Tetra_Status
SDS_ON_USERACTIVITY=SdsOnUserActivity

# wenn sich der Status bzw. die Aktivität einer teilnehmen Station ändert, wird
# an alle anderen teilnehmenden Stationen eine SDS gesendet
SDS_TO_OTHERS_ON_ACTIVITY=PROXIMITY
SDS_TO_COMMAND=SdsToCommand

# Dapnet-Konfiguration
DAPNET_SERVER=dapnet.afu.rwth-aachen.de
DAPNET_PORT=43434
DAPNET_CALLSIGN=dl1abc1
DAPNET_KEY=ganz_geheim
DAPNET_RIC2ISSI=Ric2Issi
DAPNET_RUBRIC_REGISTRATION=Ric2Rubrics
DAPNET_USERNAME=dl1abc
DAPNET_PASSWORD=ganz_geheim
DAPNET_WEBHOST=www.hampager.de
DAPNET_WEBPORT=8080
DAPNET_WEBPATH=/calls
DAPNET_TXGROUP=dl-all

[Ric2Rubrics]
# RIC=Rubric1,Rubric2,Rubric33
42568=1028,1069,1062,1061,1080

# Zordnung einer Dapnetmessage zu einer ISSI
[Ric2Issi]
# RIC=TSI
#200=23404
42568=23404
2020008=2001

[Tetra_Status]
# status=message
32849=not available
32850=available
32851=no further information
33629=EchoLink

[SdsOnUserActivity]
0=Hello user you have powered on
1=Hello user you have powered off
2=Hello user you want to declare the state of emergency?
3=Hello user Push-to-talk condition is detected
4=Hello user Status
5=Hello user TXI=on

[SdsToCommand]
33280=121

```

```
33281=120
61000=9*
33009=919
33391=9191
```

Dieser Abschnitt beschreibt den Rx-Teil der TetraLogic, das Audio wird wie bei allen anderen Logiken über den Line-In oder Mic-Eingang der Soundkarte eingespeist und das Squelch-Kriterium wird aus den PEI-Antworten generiert. Der Squelchtyp TETRA_SQL steht nur im Branch tetra-contrib zur Verfügung.

```
[Rx1]
TYPE=Local
RX_ID=1
AUDIO_DEV=alsa:plughw:0
AUDIO_CHANNEL=0
AUDIO_DEV_KEEP_OPEN=1
LIMITER_THRESH=-6
SQL_DET=TETRA_SQL
SQL_START_DELAY=50
SQL_DELAY=0
SQL_HANGTIME=200
SQL_TIMEOUT=60
SERIAL_PORT=/dev/ttyS0
SERIAL_PIN=CTS
PTY_PATH=/tmp/rx1_sql
DEEMPHASIS=0
DTMF_DEC_TYPE=INTERNAL
DTMF_MUTING=1
DTMF_HANGTIME=40
DTMF_SERIAL=/dev/ttyS0
DTMF_PTY=/tmp/rx1_dtmf
```

```
[Tx1]
TYPE=Local
TX_ID=1
AUDIO_DEV=alsa:plughw:0
AUDIO_CHANNEL=0
AUDIO_DEV_KEEP_OPEN=1
LIMITER_THRESH=-6
```

Als PTT ist bei einer TetraLogic NONE zu konfigurieren. Die
Sende-/Empfangumschaltung wird über AT-Befehle erzeugt

```
PTT_TYPE=NONE
PTT_PORT=/dev/ttyS0
PTT_PIN=DTRRTS
TIMEOUT=300
TX_DELAY=500
PREEMPHASIS=0
DTMF_TONE_LENGTH=100
DTMF_TONE_SPACING=50
DTMF_DIGIT_PWR=-15
```

Die Daten der lokalen Tetra-User werden in einem eigenen json-File gespeichert bzw. konfiguriert. Diese Datei liegt unter `/etc/svxlink` und ist entsprechend der json-Regeln zu handhaben.

```
[
  { "tsi" : "026200030000001000", "name" : "Adi", "call" : "DL1HRC-0", "location" : "Leuna", "symbol" : "\\r", "comment" : "SvxLink Sysop" },
```

```
{ "tsi" : "02620003000001001", "name" : "Adi", "call" : "DL1HRC-1", "location"
: "Leuna", "symbol" : "\\r", "comment" : "SvxLink Sysop" }
}
```

Die Parameter sind wie folgt zu konfigurieren:

tsi	- die Teilnehmernummer des Tetra-Users, MCC+MNC+ISSI (exakt 17 Digits)
name	- der Name des OM's
call	- das Rufzeichen des OM's
location	- der Standort des OM's
symbol	- das APRS-Symbol entsprechen der APRS-Vorgabe
comment	- ein Kommentar zur Station

Beim Starten liest SvxLink diese User-Daten aus der *tetra_users.json* ein und sendet diese an den angeschlossenen SvxReflector (wenn SHARE_USERDATA=1 konfiguriert wurde). Der SvxReflector sammelt diese Informationen, „mischt“ sie und sendet die aktualisierten Userdaten an alle angeschlossenen Nodes.

Für die ersten Versuche sollte der DEBUG-Level auf 4 gesetzt werden, damit man einen möglichst detaillierten Verlauf über die Funktion erhält.

In der Logausgabe während des svxLink-Startvorganges wird die aktuell installierte Version ausgegeben:

```
>>> Started SvxLink with special TetraLogic extension (v24092021)
>>> No guarantee! Please send a bug report to
>>> Adi/DL1HRC <dl1hrc@gmx.de> or use the groups.io mailing list
```

Im Log werden auch Informationen über die Zeitschlitze im DAPNET (wenn konfiguriert) die der Node nutzt ausgegeben. DAPNET ist für die ordnungsgemäße Funktion der TetraLogic nicht erforderlich.

```
+++ DAPNET: registered at time slot 0
+++ DAPNET: registered at time slot 1
+++ DAPNET: registered at time slot 2
+++ DAPNET: registered at time slot 3
+++ DAPNET: registered at time slot 4
+++ DAPNET: registered at time slot 5
+++ DAPNET: registered at time slot 6
+++ DAPNET: registered at time slot 7
+++ DAPNET: registered at time slot 8
+++ DAPNET: registered at time slot 9
+++ DAPNET: registered at time slot A
+++ DAPNET: registered at time slot B
+++ DAPNET: registered at time slot C
+++ DAPNET: registered at time slot D
+++ DAPNET: registered at time slot E
+++ DAPNET: registered at time slot F
```

Fehlermeldungen im SvxLink-Log

Während des Starts und des Betriebs erhält man immer wieder mal Fehlermeldungen wie diese:

```
To PEI:AT+CTSP=1,2,20
From PEI:+CME ERROR: 3
3 - This is a general error report code which indicates that the MT supports the command but not in its
current state. This code shall be used when no other code is more appropriate for the specific context
```

Diese Meldung ist die PEI-Antwort auf einen AT-Befehl, der im aktuell aktiven Mode, in dem sich das MRT/HRT gerade befindet, nicht unterstützt wird. Ursachen dafür könnten sein: Betriebsmodus (DMO, TMO, GATEWAY, DMO-REPEATER) oder eine Firmware, die diesen Befehl nicht unterstützt oder ein Hersteller, der diesen Befehl nicht in seine Geräte implementiert hat.

Senden einer SDS an das TETRA-SvxLink-Gateway (9999):

Im Log sind folgende Einträge zu finden:

```
From PEI:+CTSDSR: 12,23410,0,9999,0,72
From PEI:82040F012E6A746A61
+++ sending confirmation Sds to 09011638300023410
+++ sending Sds (type=25) 00023410 "8210000F", tries: 1
  To PEI:AT+CMGS=23410,032
8210000F
From PEI:+CMGS: 0
From PEI:OK
From PEI:+CMGS: 0,4,15
+++ Message sent OK, #15
```

Empfang einer DAPNET-Nachricht als Tetra-SDS:

```
---1240z FW2924/Dieskau: 13.3C w: 7.6m/s=102deg h: 50% hPa: 1002.6 r: 2.8mm/h
+++ new Dapnet message received for 23404:1240z FW2924/Dieskau: 13.3C w: 7.6m/s=102deg h:
50% hPa: 1002.6 r: 2.8mm/h
+++ sending Sds (type=1) 00023404 "1240z FW2924/Dieskau: 13.3C w: 7.6m/s=102deg h: 50%
hPa: 1002.6 r: 2.8mm/h", tries: 1
  To PEI:AT+CMGS=23404,624
82040901313234307a204657323932342f446965736b61753a2031332e334320773a20372e366d2f733d31303
264656720683a20353025206850613a20313030322e3620723a20322e386d6d2f68
---n 1240z FW2924/Dieskau: 13.3C w: 7.6m/s=102deg h: 50% hPa: 1002.6 r: 2.8mm/h
From PEI:+CMGS: 0
From PEI:OK
From PEI:+CMGS: 0,4,9
+++ Message sent OK, #9
```

Senden einer Tetra-SDS an einen DAPNET-Pager:

Eine Verbindung Tetra ↔ DAPNET ist möglich, wenn Sie eine Registrierung im DAPNET konfiguriert haben und sich Ihr SvxLink erfolgreich beim DAPNET angemeldet hat. Einen Key erhalten Sie durch das Anlegen eines DAPNET-Tickets [34].

Um eine SDS an einen Pager zu senden benötigen Sie das Rufzeichen des Empfängers und erstellen eine neue SDS-Nachricht im Format „DAP:rufzeichen:Textnachricht“ als Message.

Beispiel:

```
DAP:dl1hrc:das ist ein Test, mni 73s
```

Als Startzeichenfolge kann dabei DAP oder dap (also entweder Groß oder Kleinschreibung) verwendet werden. Diese SDS senden Sie an Ihren SvxLink-Gateway, dieser organisiert dann die Weiterleitung an das DAPNET.

5.4. Konfiguration der Tetra-Mobilgeräte

5.4.1. ISSI

Für die Anwendung im Amateurfunkbereich sind folgende Einstellungen zu empfehlen (DMO):

MCC (Mobile Country Code) = 0901
MNC (Mobile Network Code) = 16383

ISSI (*Individual Short Subscriber Identity*) = DMR-Id mit führender 0, z.B. 02620055
GSSI (*Group Short Subscriber Identity*) = 1

Die Station die jede SDS (inklusive LIP-SDS) empfängt sollte die ISSI 9999 haben. Das ist i.A. nicht der DMO-Repeater selbst, da dieser im DMO-RPT-Modus keine SDS senden kann. Es empfiehlt sich die Verwendung eines Standalone-DMO-Repeater und eines Audio-Zubringers (i.A. ein MTM800e) welches mit einem Reflector verbunden ist und die Daten von/zu diesem weiterleitet.



Bild 30: Prinzip DMO-Repeater und „Audio-Zubringer“

5.4.2. LocationInfo

Die Einstellung der LIP-SDSen erfolgt in der CPS über die Position:

Codeplug → GPS → LIP Configuration

Field Name	Field Value	Set Default
1 Send Initialization Request	☐	
2 Destination ISSI	9999	Set Default
3 Destination ISSI in Emergency Call	100	Set Default
4 Destination ISSI in Assistance Call	100	Set Default
5 Destination IP Address	0 .0 .0 .0	
6 Remote Port	1	Set Default
7 Minimum Reporting Interval	10 s	Set Default
8 Enable/Disable Call Temporary Minimum Reporting Interval	☒	
9 Call Temporary Minimum Reporting Interval	Eight Times Normal Minimum Reporting Interval	Set Default
10 Max Data Age	15 s	Set Default
11 Max Response Time	10 s	Set Default
12 Send Immediate Location Reports to Requestor	☐	
13 Delete All OTA LIP Report Requests		Reset
14 Enable LIP Emergency Entry Trigger when Location is Off	☐	
15 Sending of Initialisation Request when Lifetime is "Until Migration"	Only at Migration	Set Default
16 Enable LIP Notifications	☒	
17 Enable LIP Location Report Backlog	☒	
18 LIP Backlog Report Method	Automatic	Set Default
19 LIP Backlog Reporting Order	Ascending	Set Default
20 LIP Backlog Report Initial Delay, sec	30	Set Default
21 LIP Backlog Reporting Interval, sec	30	Set Default
22 Max LIP Location Backlog Number	180	Set Default
23 Report Type when No GPS Fix	Use Existing Report Type	Set Default
24 Basic Location Report Type	Short location report preferred	Set Default

Bild 31: Konfiguration der LocationInfo (LIP-SDS an Tetra-Link)

Idealerweise sollte die Ziel-ISSI für den Empfang der SDS'en und Positionsdaten überall die selbe sein (9999).

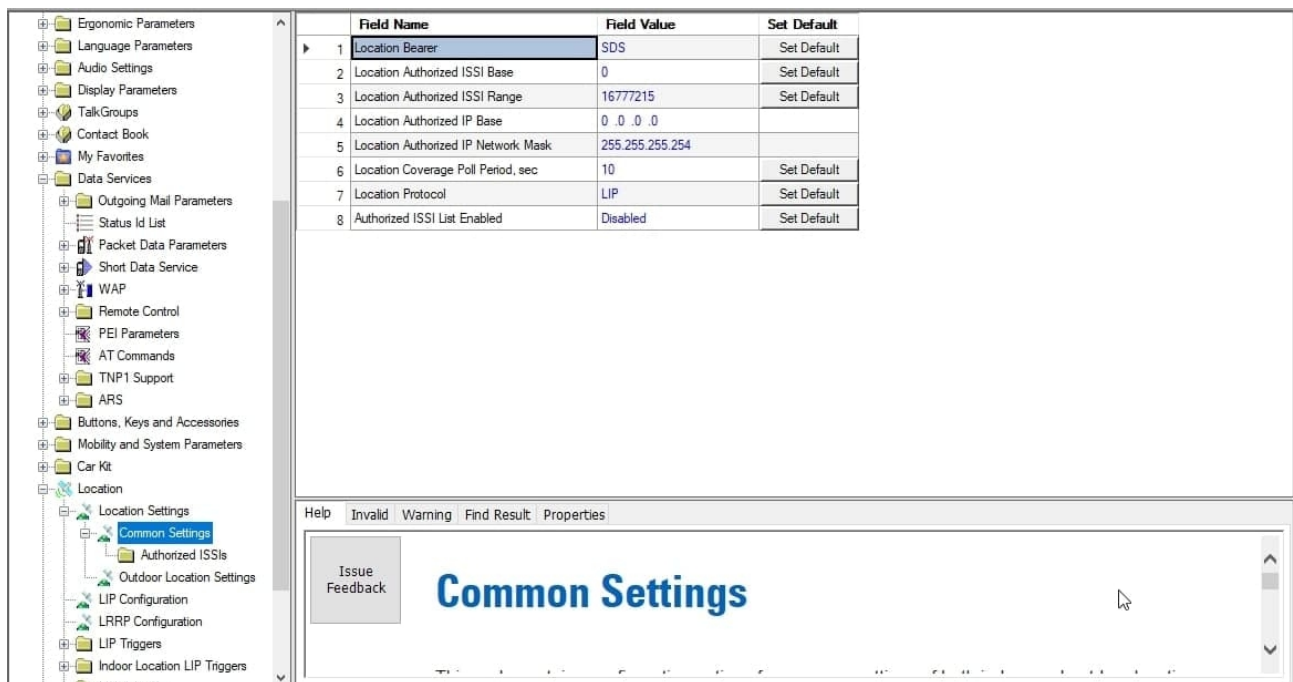


Bild 32: Konfiguration der Locationinfo 1/2 (LIP-SDS an Tetra-Link)

Die Übertragung der Positionsmeldung kann entweder per LIP-SDS oder LIRP-SDS erfolgen. SvxLink unterstützt derzeit nur das LIP-Protokoll.

Im Menüpunkt „Codeplug → Location → Location Settings → Outdoor Location Settings“ müssen folgende Optionen gesetzt werden:

- Location Coordinates = „Latitude/Longitude“
- Icon for Location Coverage = [✓]
- Location Enable [✓]

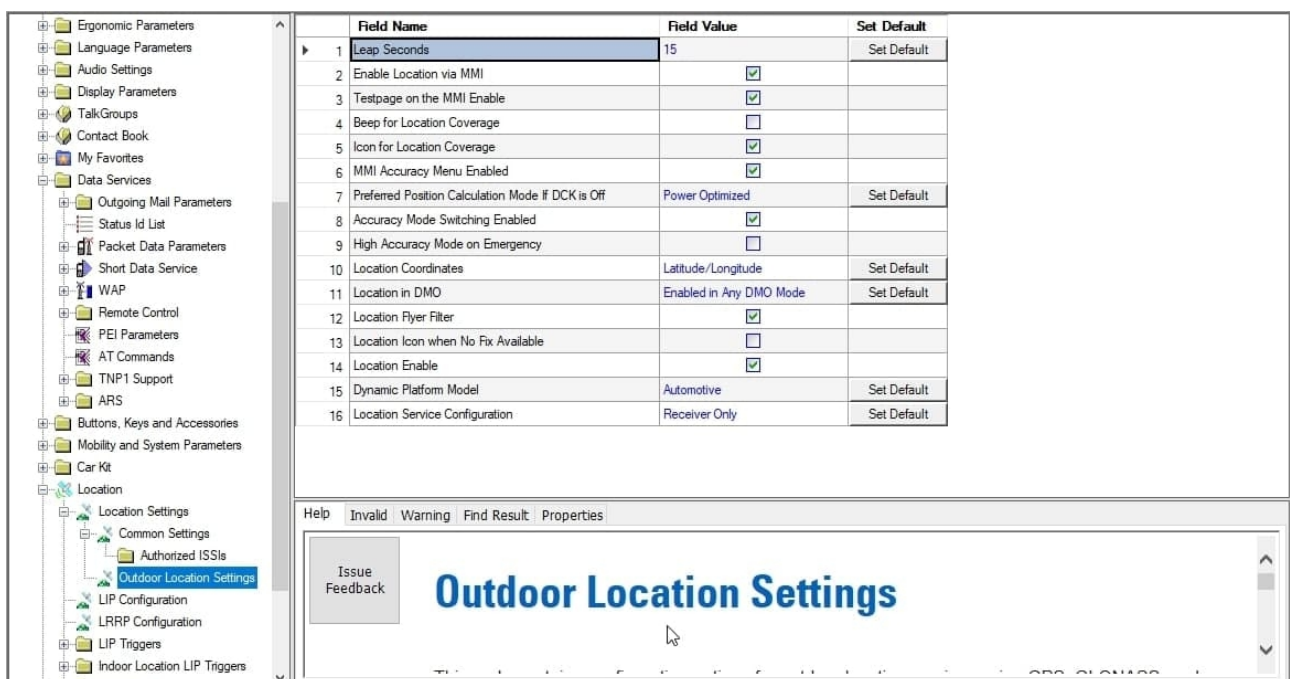


Bild 33: Konfiguration der Locationinfo (2/2)

5.4.2.1. Positionsmeldung periodisch senden

Im Menüpunkt „Copdeplug → Location → LIP Trigger → Periodic“ kann man die periodischen Positionsmeldungen einschalten. Hier sollte man die Baken zeitlich nicht zu eng setzen, 30 Minuten sind sicher ausreichend.

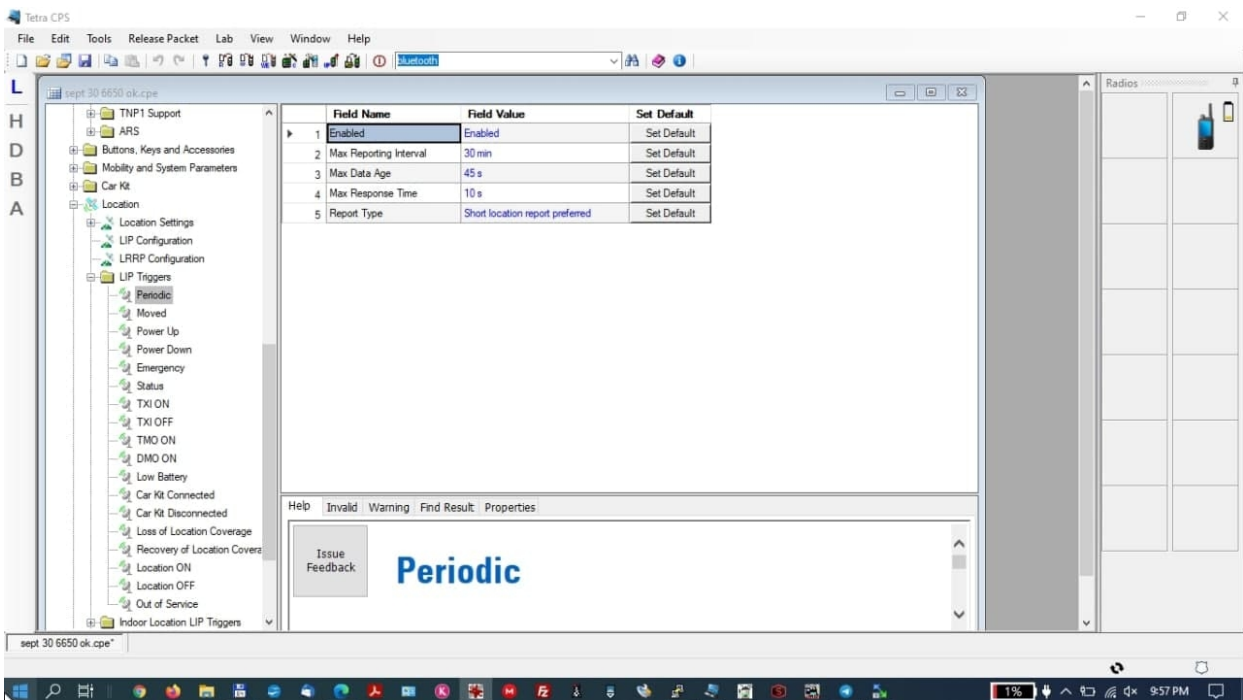


Bild 34: Periodische Positionsmeldung einschalten „periodic=on“

5.4.2.2. Positionsmeldung senden bei DMO=on

Bei Aktivierung wird eine Positionsmeldung gesendet, sobald das MS/HRT vom TMO-Modus, DMO-Repeater- oder Gateway-Modus in den DMO-Modus geschaltet wird.

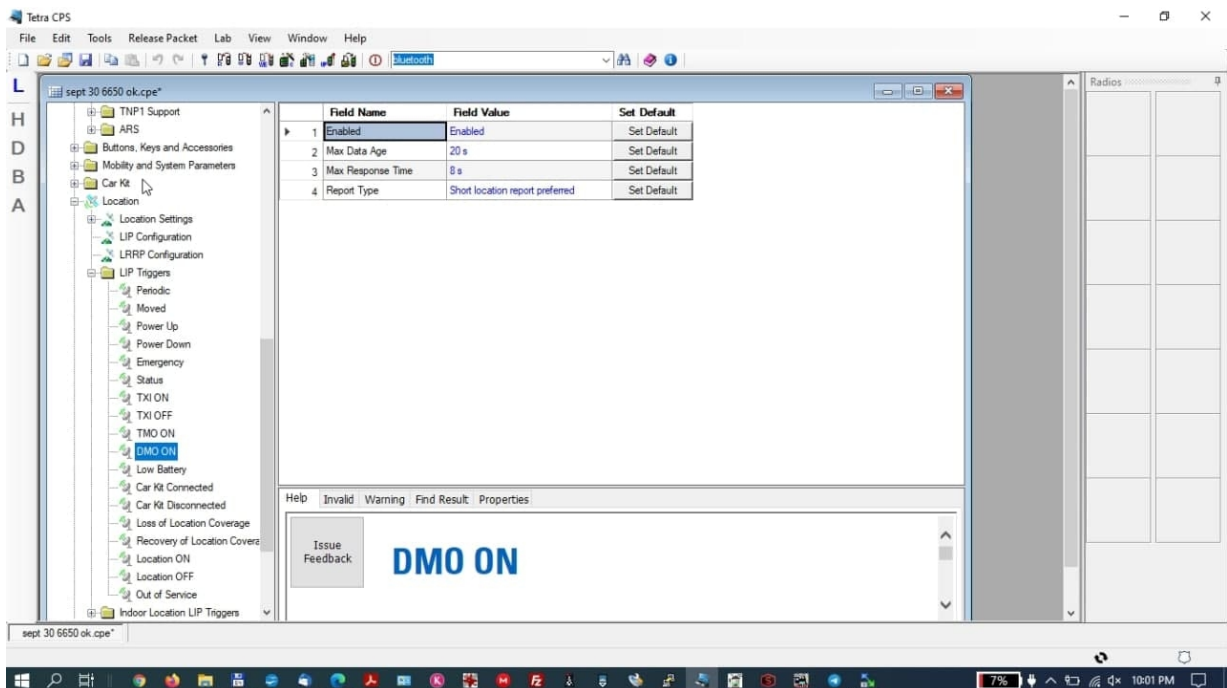


Bild 35: Positionsmeldung bei „DMO=on“

5.4.2.3. Positionsmeldung bei Low Battery

Es empfiehlt sich keine Positionsmeldung bei einer geringen Akkukapazität zu senden

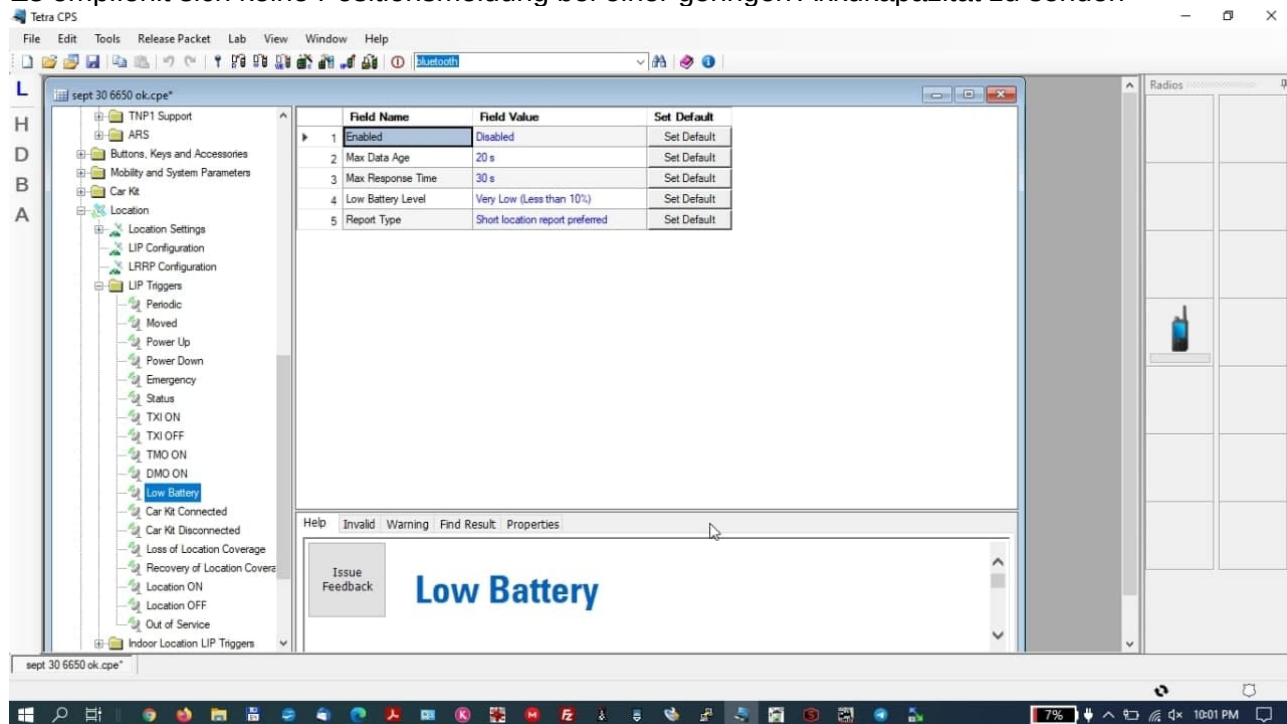


Bild 36: Keine Positionsmeldung bei „low battery“

5.4.2.4. Positionsmeldung bei Gefahr

Beim Drücken des Notruf-Buttons erfolgt gleichzeitig die Aussendung einer Positionsmeldung als LIP-SDS.

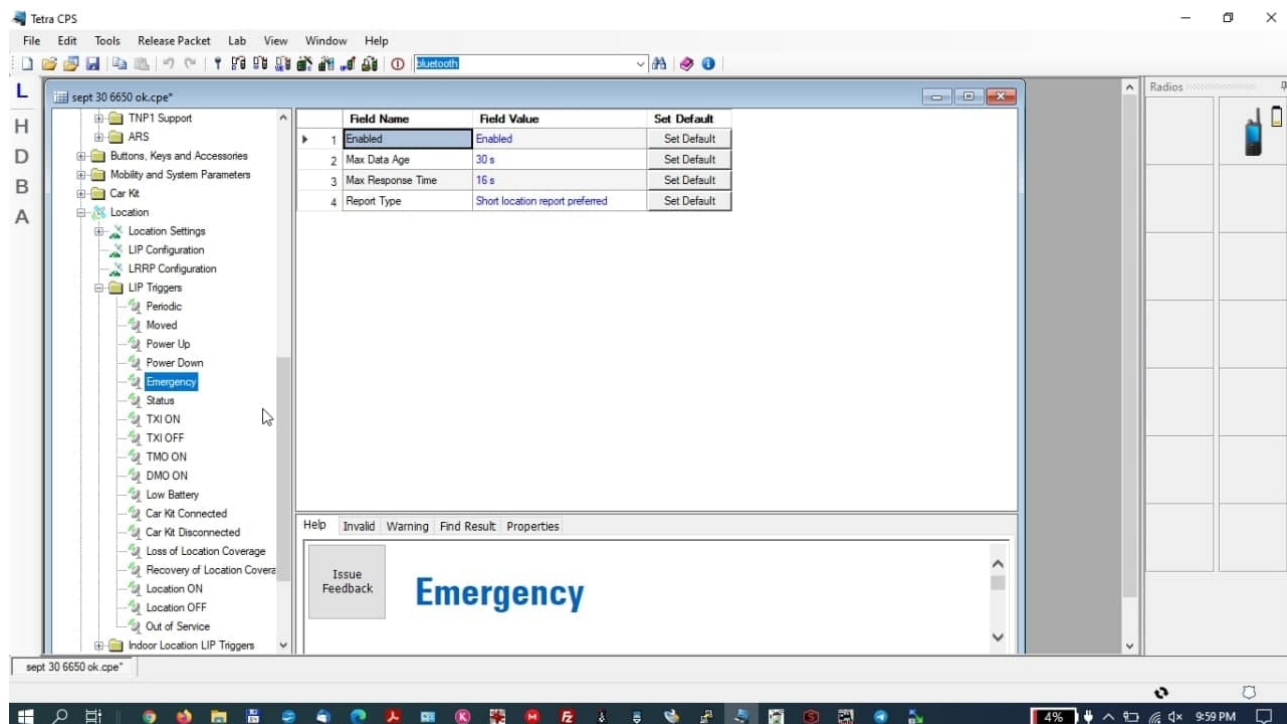


Bild 37: Positionsmeldung bei „Emergency“

5.4.2.5. Postionsmeldung beim Senden einer Status-SDS

Wird dieser Parameter aktiviert, so wird parallel zum Senden eines Status automatisch eine LocationInfo ausgesendet.

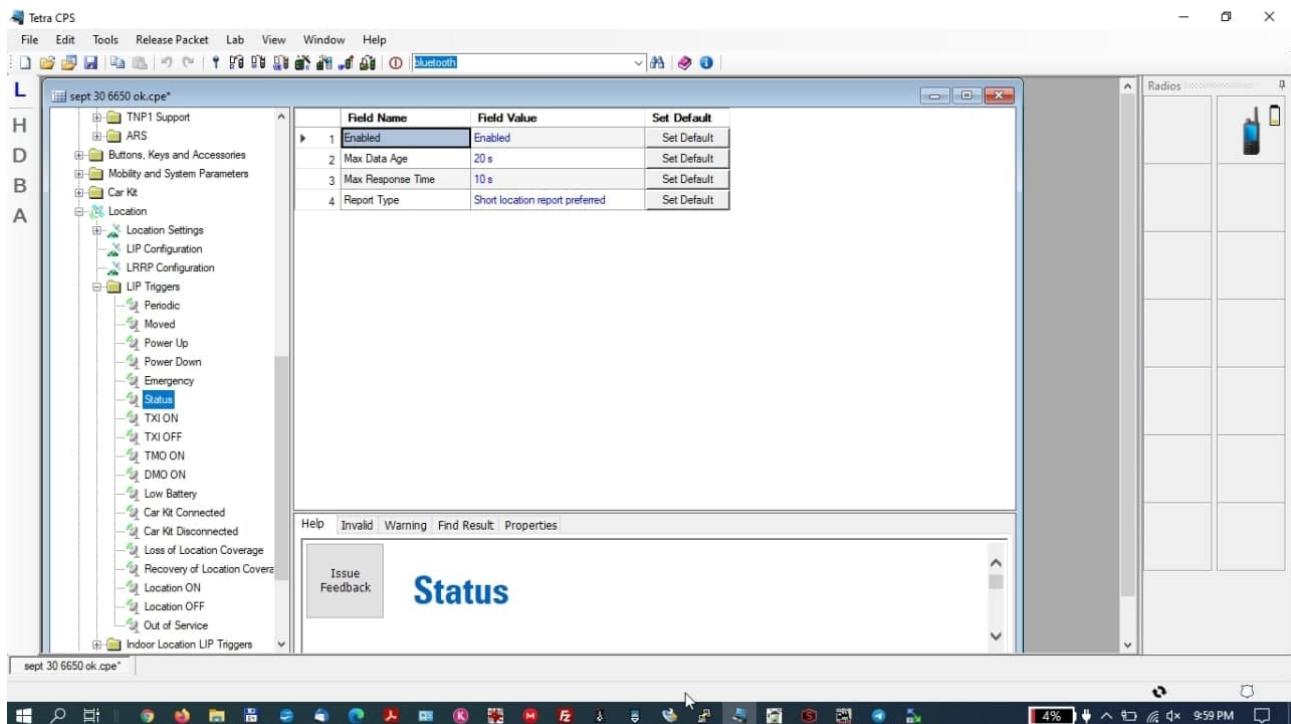


Bild 38: Positionsmeldung beim Senden eines „Status“

5.4.2.6. Postionsmeldung beim Ausschalten des HRT/MRT's

Empfehlenswert ist eine letzte Bake vor dem Ausschalten des HRT/MRT's zu senden. Dazu muß der Parameter „Codeplug → Location → LIP Triggers → Power Down“ aktiviert sein - nicht wie im Bild 39 dargestellt.

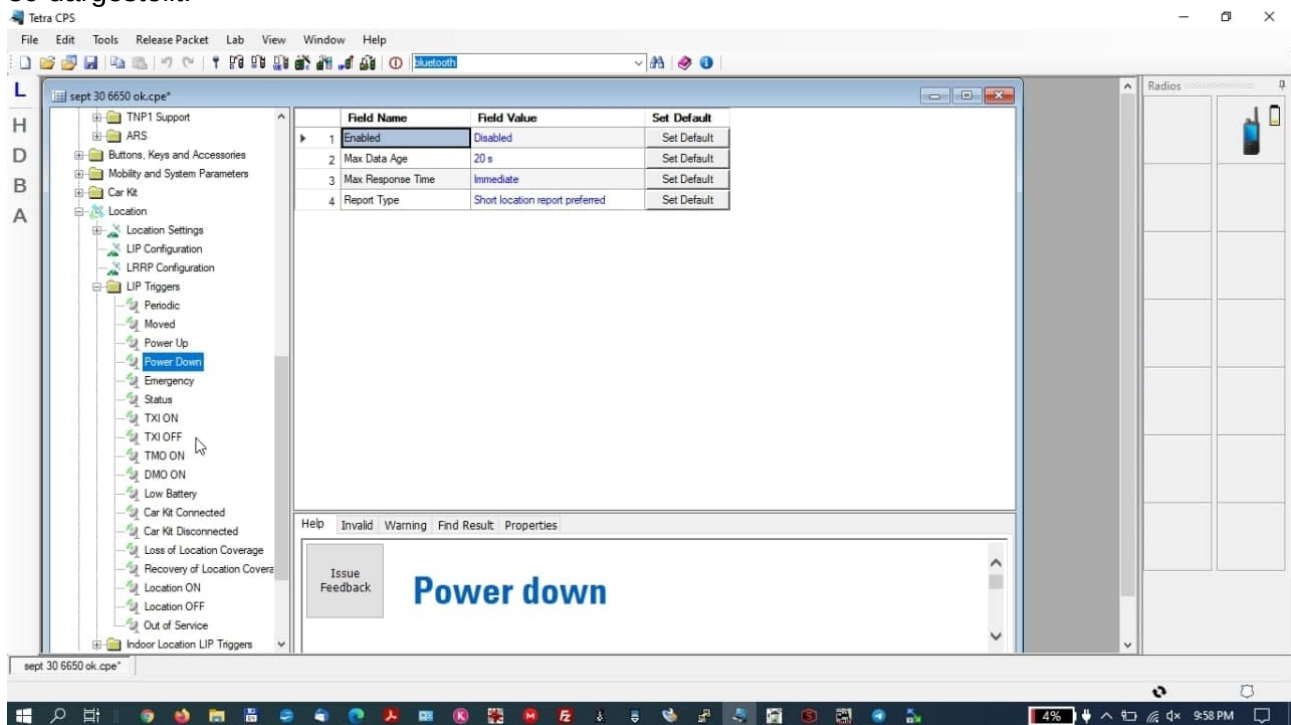


Bild 39: Positionsmeldung beim Ausschalten des Gerätes (sollte aktiviert sein)

5.5. Branch tetra-sip

Der Branch „tetra-sip“ beinhaltet alle Funktionen der Branche „tetra-contrib“ und „pjSipLogic“. Sie können also von einem Tetra-Gerät aus eine Telefonieverbindung (im DMO leider nur simplex und VOX-gesteuert) aufbauen.

Zur Installation und Inbetriebnahme dieses Branches gehen Sie wie folgt vor:

- Installieren Sie das pjSip-Framework wie in Abschnitt 3 angegeben
- Klonen Sie den tetra-sip-Branch und installieren Sie diesen
- Konfigurieren Sie den Abschnitt [SipLogic] wie im Abschnitt 3.1. angegeben entsprechend Ihren Anforderungen
- Konfigurieren Sie den Abschnitt [TetraLogic] wie im Abschnitt 5 angegeben entsprechend Ihren Anforderungen

Um die Anwahl eines Sip-Teilnehmers vom Tetra-Endgerät aus zu ermöglichen sind einige Vorarbeiten erforderlich.

In den Sourcen, welche Sie von github heruntergeladen haben, befindet sich die Datei SipDialOutfromTetra.tcl. Diese Datei ist sozusagen die Schnittstelle zwischen Tetra-Bereich und SipLogic-Bereich und leitet die im Tetra-Bereich empfangenen Kommandos an Sip weiter. Hier der Inhalt der entsprechenden Prozedur:

```
namespace eval TetraLogic {

# Executed when a DTMF command has been received
# cmd - The command
#
# Return 1 to hide the command from further processing in SvxFLink or
# return 0 to make SvxFLink continue processing as normal.
# ---> must return 0!
# adapt the port according to your needs
#
proc dtmf_cmd_received {cmd} {
    variable port;
    variable number;

    if {[string length $cmd] > 5 && [string range $cmd 0 1] != 91} {
        puts "### dialing number $cmd via sip pty";
        # adjust the variable port according to your specifications
        # "/tmp/ctrl" must be match to SIP_CTRL_PTY in [SipLogic]-section
        set port [open "/tmp/sipctrl" w+];
        set number "C$cmd";
        puts $port $number;
        close $port;
    }

    # if a call isn't answered you can stop the call by sending an other ssc, eg
    # [SdsToCommand]
    # 32771=12
    # the next section will end the pending call
    if {$cmd == "12"} {
        set port [open "/tmp/sipctrl" w+];
        set number "C#";
        puts $port $number;
        close $port;
    }
}
```

```
    return 0;
}

# end of namespace
}
```

Momentan ist noch die händische Anpassung verschiedener Parameter und Variablen notwendig, in kommenden Versionen will ich versuchen diesen Anteil zu reduzieren.

Definieren Sie in der *svxlink.conf* im Bereich [SdsToCommand] mindestens ein Kommando für die Anwahl eines Sip-Partners, ein weiteres für das Beenden des Gesprächs bzw. der Anwahl und weiterhin im Bereich [SipLogic] das Pseudo-TTY an das die Kommandos gesendet werden sollen, Beispiel:

```
[SdsToCommand]
▶ 32776=12
▶ 32768=2620055

[SipLogic]
SIP_CTRL_PTY=/tmp/sipctrl
```

Eine Status-SDS mit der ID 32768 (8000hex) soll in diesem Fall die Sip-Nummer 2620055 anwählen. Die Status-SDS 32776 initiiert das Kommando 12, das bei der Beispielkonfiguration eine bestehende Sip-Anwahl beenden soll (siehe oben dargestellte Tcl-Datei – fettgedruckter Eintrag).

6. Installation und Konfiguration weiterer Branche

6.1. hotspot-nmea-Branch

Der Branch hotspot-nmea greift in die Funktion der LocationInfo ein. Die Positionsdaten werden dabei nicht aus der *svxlink.conf* über die Parameter LAT_POSITION und LON_POSITION bereitgestellt sondern entweder über eine NMEA_Gerät (GPS-Maus über */dev/ttyUSBx*) ausgewertet oder per TCP-Verbindung vom *gpsd* übernommen. Mögliche Anwendungen wären z.B. wenn man den Hotspot im Auto, Fahrrad oder Boot mitführt. Im Branch ist ein einfaches Beaconsing enthalten, das durch den Parameter BEACON_INTERVAL für die festen Baken und intern eine zusätzliche Bake bei einer Positionsänderung von >0.5km oder einer Winkeldifferenz von 12.5° definiert wird.



Bild 40: Beispiel für einen USB-GPS-Empfänger (U-blox 7, Bezug z.B. über eBay ~8€ Stand 5/2022)

Installieren Sie den Branch hotspot-nmea wie im Abschnitt 2.2.3 beschrieben (Auschecken per „git checkout hotspot-nmea“).

Konfigurieren Sie die neuen und nur in diesem Branch enthaltenen Parameter entsprechend Ihren Anforderungen. Falls Sie eine frühere Installation von SvXLink nutzen so fügen Sie bitte die fehlenden Parameter in der *svxlink.conf* hinzu.

`NMEA_DEVICE=/dev/ttyACM0`

Dieser Parameter definiert die Schnittstelle an die das NMEA-Gerät/die GPS-Maus angeschlossen ist, Beispiele: */dev/ttyUSB0*, */dev/ttyACM0*. Es wird dabei erwartet dass das Gerät eine Positionsmeldung \$GPRMC,xxxxx bereitstellt, sonst funktioniert es nicht!

`NMEA_BAUD=4800`

Definiert die Baudrate um mit dem NMEA-Gerät zu kommunizieren (im Allgemeinen 4800 oder 9600)

`GPSD=1`

Wenn GPSD=1, dann wird versucht die Position von einem per TCP-Verbindung erreichbaren *gpsd* zu ermitteln. Die oben beschriebenen Parameter NMEA_DEVICE und NMEA_BAUD sind dann ohne Funktion.

`GPSD_HOST=127.0.0.1`

IP-Adresse des *gpsd*, meist localhost (127.0.0.1). Achtung: der *gpsd* muß explizit für die Kommunikation über ein Netzwerk konfiguriert werden. Ohne weitere Konfiguration kann nur ein lokal laufender *gpsd* angefragt werden.

GPSD_PORT=2947

Der Standardport für die Kommunikation mit dem gpsd.

Der gpsd wird auf debianbasierten Installationen über die Datei */etc/default/gpsd* konfiguriert, hier ein Beispiel:

```
# Default settings for the gpsd init script and the hotplug wrapper.
# Start the gpsd daemon automatically at boot time
START_DAEMON="true"

# Use USB hotplugging to add new USB devices automatically to the daemon
USB_AUTO="true"

# Devices gpsd should collect to at boot time.
# They need to be read/writeable, either by user gpsd or the group dialout.
DEVICES="/dev/ttyACM0"

# Other options you want to pass to gpsd
GPSD_OPTIONS="-G -n -b"
```

Sie müssen den Parameter DEVICES entsprechend Ihrer Gegebenheiten anpassen (Schnittstelle per minicom testen)

Der Parameter GPSD_OPTIONS sollte das Flag -G enthalten, damit der gpsd über Ihr Netzwerk auch von anderen Clients angesprochen werden kann.

6.2. AnnounceLogic-Branch

Die AnnounceLogic kann für das Abspielen von Rundsprüchen oder sonstigen zeitgesteuerten Ansagen verwendet werden. Dabei wird das erzeugte Audio direkt an eine verbundene Logik oder in einen Repeater-Verbund gestreamt. Eine Einspielung per HF ist nicht notwendig.

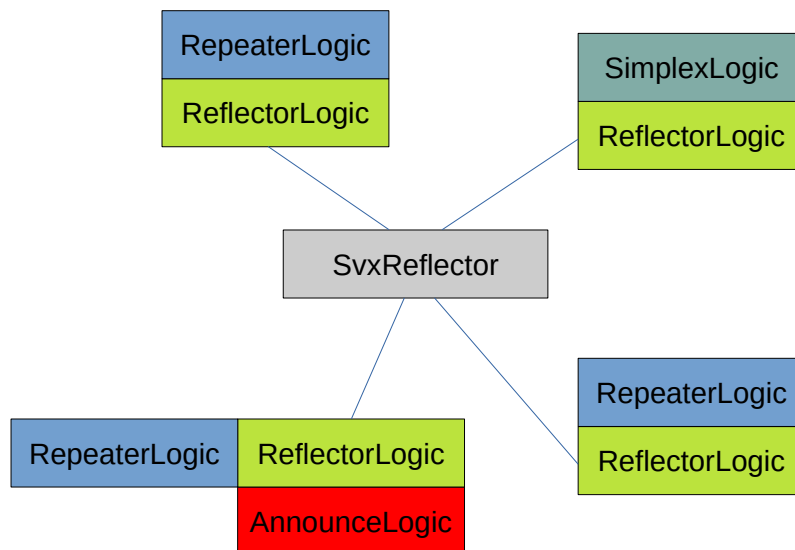


Bild 41: AnnounceLogic in einem Relaisverbund

Bild 41 zeigt die prinzipielle Verwendung einer *AnnounceLogic* in einem Verbund. Im Tcl-Userspace können einmal der Rundspruch selbst als auch eine Vorabinformation konfiguriert werden. Der Gesamtablauf ist in *Tabelle 2* dargestellt. Angenommen wird der Beginn des Rundspruchs um 18Uhr mit dem Beginn der Vorabinformation („Achtung, in Kürze wird der Rundspruch auf diesem Relais übertragen.“) um 17:40 und dessen Wiederholung im Intervall von jeweils 5 Minuten.

Tabelle 2: Abfolge zur Aussendung eines Rundspruchs

Schritt	UTC	AnnounceLogic-Node	Verbundene Nodes	Bemerkung
1	< 17:40	Normales Arbeiten vor dem Rundspruch, Vorabinformation erfolgt um 17:40		alle Rx'e = aus
2	17:40	Abschaltung Rx	Abschaltung aller Rx'e	
3	17:40	Ausgabe Vorankündigung Rundspruch	Ausgabe Vorankündigung Rundspruch	
4	17:45	Wiederholung Ausgabe Vorankündigung Rundspruch	Wiederholung Ausgabe Vorankündigung Rundspruch	alle Tx=on
5	17:50	Wiederholung Ausgabe Vorankündigung Rundspruch	Wiederholung Ausgabe Vorankündigung Rundspruch	
6	17:55	Wiederholung Ausgabe Vorankündigung Rundspruch	Wiederholung Ausgabe Vorankündigung Rundspruch	
7	18:00	Rundsprucheinspeisung per AnnounceLogic	Ausgabe Rundspruch über den Tx	
8	~18:32	Beendigung Rundspruch, Tx fällt ab	Alle Tx'e fallen ab	
9	18:32	Aktivierung Rx	Aktivierung aller Rx'e	
10	> 18:32	Normales Arbeiten wie vor dem Rundspruch		

Nachfolgend die Beschreibung der Konfigurationsparameter in der *svxlink.conf* für den Ablauf entsprechend *Tabelle 2*:

```
[AnnounceLogic]
TYPE=Announce
START_PRENOTIFICATION_MINUTES_BEFORE=20
PRENOTIFICATION_INTERVAL=5
DAY_OF_WEEK=3
DAYS_IN_MONTH=1, 3
HOUR_OF_QST=18
MINUTE_OF_QST=0
EVENT_HANDLER=/usr/share/svxlink/events.tcl
```

Beschreibung der Parameter:

TYPE=Announce

Der Typ der Logik muß immer „Announce“ sein.

START_PRENOTIFICATION_MINUTES_BEFORE=20

Zeit in Minuten vor Beginn des eigentlichen Rundspruches zu der eine Vorankündigung des Rundspruches starten soll.

PRENOTIFICATION_INTERVAL=5

Zeitinterval in Minuten zu dem die Vorankündigung jeweils wiederholt werden soll.

DAY_OF_WEEK=3

Wochentag an dem der Rundspruch ausgegeben wird (0=Sonntag, 1=Montag, ..., 6=Samstag).

DAYS_IN_MONTH=1, 3

Definiert den Zyklus, hier erster und dritter „DAY_OF_WEEK“ im Monat.

HOURL_OF_QST=12

Stunde der Ausgabe des Rundspruchs im 24h-Format

MINUTE_OF_QST=15

Minute der Ausgabe des Rundspruches

EVENT_HANDLER=/usr/share/svxlink/events.tcl

Der Eventhandler, Pfad zur events.tcl (immer angeben)

Die neue Datei *AnnounceLogic.tcl* beinhaltet zwei neue Prozeduren. Die Prozedur *announce_prenotification* gibt die Vorabinformation zum Rundspruch aus.

```
#
# Executed when the pre broadcast notification is played
#
proc announce_prenotification {} {
    puts "Playing pre broadcast notification";
    playMsg "AnnounceLogic" "prenotify";
}
```

Die Prozedur *announce_qst* gibt den den eigentlichen Rundspruch aus:

```
#
# Executed when the broadcast is played
#
proc announce_qst {} {
    puts "Playing broadcast";
    playMsg "AnnounceLogic" "Broadcast";
}
```

Damit die Ausgaben gefunden werden, müssen im obigen Beispiel die Dateien „prenotify.wav“ und „Broadcast.wav“ im Verzeichnis */usr/share/svxlink/events.d/sounds/de_DE/AnnounceLogic* liegen.

Damit die Rundspruchausgabe störungsfrei erfolgt, kann man die einzelnen RX'e aller verbundenen Relais für diesen Zeitraum abgeschaltet werden. Hierzu muß auf jedem Relais ein Voter konfiguriert werden, auch wenn die Voting-Funktionalität nicht verwendet wird.

Hier ein Konfigurations-Beispiel wobei nur die wichtigsten Parameter aufgeführt werden:

```
[RepeaterLogic]
```

```
...
```

```
RX=Voter
```

```
[Voter]
```

```
TYPE=Voter
```

```
RECEIVERS=Rx1
```

```
...
```

```
COMMAND_PTY=/tmp/voter_ctrl_pty
```

Über das zur Laufzeit erzeugte Pty-Device kann die Voter-Funktionalität gesteuert werden. Auf der Kommandozeile kann der Rx1 eines Relais durch

```
echo "DISABLE Rx1">/tmp/voter_ctrl_pty
```

abgeschaltet und durch

```
echo "ENABLE Rx1">/tmp/voter_ctrl_pty
```

wieder zugeschaltet werden. Idealerweise kann man das im TCL-Framework der einzelnen Relais automatisiert bzw. zeitgesteuert konfigurieren. Die passende Prozeduren um den entsprechenden Rx ein- bzw. auszuschalten könnte so aussehen (im Namespace RepeaterLogic). Die Datei *Announcement_remote_nodes.tcl* liegt im Branch AnnounceLogic und sollte auf den Nodes aktiviert werden, die nur verbunden sind und selbst nicht die Broadcast-Message abspielen. Dieses File sollte unter Beachtung des locale-Konzepts im Verzeichnis */usr/share/svxlink/events.d/local* installiert werden.

```
#####  
#  
# RepeaterLogic event handlers for Announcements/qst transmissions  
#  
#####  
  
#  
# This is the namespace in which all functions below will exist. The name  
# must match the corresponding section "[RepeaterLogic]" in the configuration  
# file. The name may be changed but it must be changed in both places.  
#  
namespace eval RepeaterLogic {  
  
# state of playing the qst announcement, do not edit  
variable playqst 0;  
  
# activates the announcement handling  
variable enable_qst 1;  
  
# defines if the Rx's should be deactivated  
variable disable_rx 1;  
  
# time when the Rx's are deactivate, format HHMM, e.g. 0730  
variable disable_rx_time "1740";  
  
# time when the Qst starts  
variable qst_starts_at "1800";  
  
# time when the Rx's are activate again  
variable enable_rx_time "1830";  
  
# name of the Rx defined in svxlink.conf  
variable rx_name "Rx1";  
  
# path of the pty-device to control the voter  
variable ctrl_pty "/tmp/voter_ctrl_pty"  
  
# day of week on which the announcement file will be played  
# 0=Sunday, 1=Monday, ..., 6=Saturday  
variable dayofqst 3;  
  
# weeks on which the announcement file will be played  
# 2nd and 4th week  ="2,4";  
variable weeksofmonth "2,4";  
  
#  
# extended procedure, check_if_time_for_qst  
#  
proc check_if_time_for_qst {} {  
    variable enable_qst;  

```

```

variable dayofqst;
variable disable_rx_time;
variable enable_rx_time;
variable weeksofqst;
variable qst_starts_at;
variable weeksofmonth;
variable dom;

if {!$enable_qst} {
    return;
}

# check if day of week is ok
set systime [clock seconds];
set dayofweek [clock format $systime -format %w];
set dayofmonth [clock format $systime -format %d];
set items [split $weeksofmonth ","];
set acttime [clock format $systime -format %H%M];

foreach dom $items {
    if {$dom * 7 >= $dayofmonth && ($dom - 1) * 7 <= $dayofmonth} {
        if {$dayofqst == $dayofweek} {
            if {$disable_rx_time == $acttime} {
                enableRx 0;
            } elseif {$enable_rx_time == $acttime} {
                enableRx 1;
            }

            if {$qst_starts_at == $acttime} {
                set playqst 1;
            }
        }
    }
}
}

```

```

#
# enable/disable Rx via pty-device
#
proc enableRx {enable} {
    variable rx_name;
    variable ctrl_pty;
    set ena "ENABLE";

    if {$enable == 0} {
        set ena "DISABLE";
    }

    set port [open $ctrl_pty w+];
    puts "Setting Rx \"$rx_name\" to $ena";
    puts $port "$ena $rx_name";
    close $port;
}

```

```

#
# extended procedure transmit
#
proc transmit {is_on} {
    variable playqst;

```

```

    if {!$is_on && $playqst} {
        enableRx 1;
        set playqst 0;
    }
    Logic::transmit $is_on;
}

#
# check if an announcement is available in
#
proc every_minute {} {
    set systime [clock seconds];
    set dateFile [clock format $systime -format %d%m%Y%-H%M];
    check_if_time_for_qst;
    Logic::every_minute;
}

# end of namespace
}

#
# This file has not been truncated
#

```

Die Beschreibung der Parameter am Anfang der tcl-Datei erfolgt in Kürze.

7. Installation von Zusatzmodulen

Diese Anleitung beschreibt die Installation von Zusatzmodulen für SvxLink:

- PropagationMonitor
- KatWarn
- Unwetterwarnung
- Announcement

Zu empfehlen ist schrittweise vorzugehen und nicht alle Module mit einem Mal zu installieren.

Die folgenden Punkte gelten für alle Module. Wenn Sie einen der folgenden Punkte nicht erfüllen können bzw. es zu Fehlern kommt, macht es keinen Sinn mit den weiteren Punkten weiterzumachen.

- die Anleitung gilt nur für Raspbian/Debian/Ubuntu-Derivate
- der SvxLink-Node muß mit einem Bein im Internet stehen, d.h. er ist über Port 25/TCP von außen erreichbar. Anbindungen über Mobilfunk oder NAT's vieler Provider funktionieren nicht!
- es ist eine aktuelle SvxLink-Version installiert (Master, SvxLink v1.7.99.20)
- beim Betrieb über DSL muß eine Portweiterleitung von Port 25/TCP auf den SvxLink-Node eingerichtet werden
- der SvxLink-Node muß per Hostnamen vom Internet aus erreichbar sein. Beim Betrieb über DSL o.ä. muß ein Dienst laufen, der den Hostnamen (hier als Beispiel für diese Anleitung im Folgenden „dl1abc.dyndns.org“) mit der öffentlichen IP synchronisiert, z.B. ddclient

Bevor Sie weiterarbeiten muß gewährleistet sein, dass der Aufruf

```
nslookup dl1abc.dyndns.org
```

ein valides Ergebnis liefert.

- mit Ausnahme des PropagationMonitors müssen die folgenden Perl-Erweiterungen installiert werden:

```
sudo apt-get update
```

```
sudo apt-get install libdigest-perl-md5-perl libwww-perl libnet-http-perl libmime-tools-perl
```

- auf dem SvxLink-Node muß ein Mailserver (postfix) und procmail zur Mail-Nachbearbeitung installiert werden:

```
sudo apt-get install procmail postfix
```

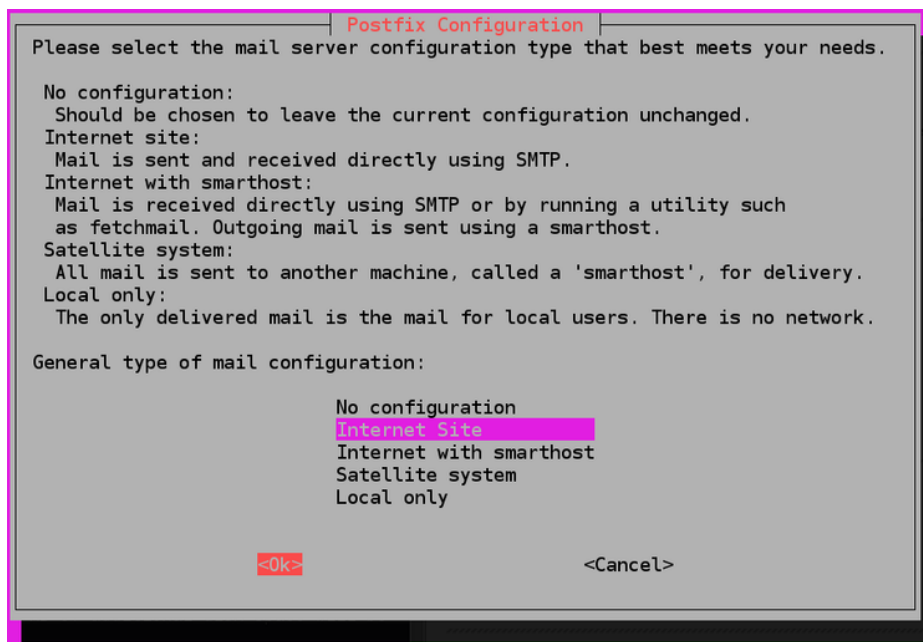


Bild 42: Abfragen während des Installationsprozesses

Während der Installation von postfix erfolgen einige Abfragen.

Konfigurieren Sie den MTA entsprechend des von Ihnen gewählten Hostnamens, hier: *dl1abc.dyndns.org*

Bearbeiten Sie manuell die Datei */etc/postfix/main.cf* mit einem Editor Ihrer Wahl (BTW: „Word“ ist kein Editor). Prüfen Sie, ob nachfolgende Einträge vorhanden sind bzw. passen Sie diese an Ihre Umgebung an:

```
myhostname = dl1abc.dyndns.org
mydomain = dl1abc.dyndns.org
myorigin = $myhostname
inet_interfaces = all
mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain,
mynetworks = 192.168.178.0/24, 127.0.0.0/8 #← IP Ihres lokalen Netzes
relay_domains = $mydestination
mailbox_command = /usr/bin/procmail
sendmail_path = /usr/sbin/sendmail
mailq_path = /usr/bin/mailq
inet_protocols = ipv4
```

Noch ein wichtiger Hinweis: die anderen Parameter wie *html_directory*, *manpage_directory* oder *sample_directory* sollten entweder konfiguriert oder auskommentiert werden. In meinem Fall funktionierte der postfix im Falle von

html_directory=
nicht richtig.

Laden Sie sich jetzt das Git-Repository SvxLink-Contributions herunter.

git clone <https://github.com/dl1hrc/SvxLink-Contributions>

In diesem sind die benötigten Files und auch die Verzeichnisstruktur enthalten. Stellen Sie sicher, dass diese Verzeichnisse für den User svxlink und die Gruppe svxlink beschreibbar sind. Bild 43 zeigt die Verzeichnisstruktur der SvXLink-Contributions.

.git	/.	ÜBERVZ.	15. Nov 14:18
etc	/modules.d	4096	15. Nov 14:18
└─ svxlink	/sounds	4096	15. Nov 14:18
└─ svxlink.d	*make_DWD.pl	8704	15. Nov 14:18
home	*make_KatWarn.pl	8746	15. Nov 14:18
└─ svxlink	*make_announcement.pl	8614	15. Nov 14:18
usr			
└─ share			
└─ perl			
└─ 5.28.1			
└─ svxlink			
└─ modules.d			
└─ sounds			
└─ de_DE			
└─ WeatherInfo			

Bild 43: Verzeichnisstruktur der SvxDLink-Contributions

Übernehmen Sie diese 1-zu-1 auf Ihr System (mit Ausnahme des Verzeichnisses „git“). Im Folgenden gehen wir davon aus, dass der gesamte eMail-Verkehr über den Nutzer svxlink durchgeführt wird, d.h. Sie müssen die eMail-Adresse svxlink@dl1abc.dyndns.org auf dem jeweiligen Portal registrieren. Eine Ausnahme bildet hier der Deutsche Wetterdienst, der leider keine Registrierung von dyndns-Adressen zulässt. Um eingehende eMails gezielt weiterleiten zu können erzeugen Sie eine Datei `.procmail` im `/home/svxlink` Verzeichnis. Achten Sie darauf, dass Eigentümer und Gruppe immer svxlink:svxlink ist, **das gilt für alle Dateien/Verzeichnisse** im Weiteren.

7.1. PropagationMonitor (Ausbreitungsmonitor)

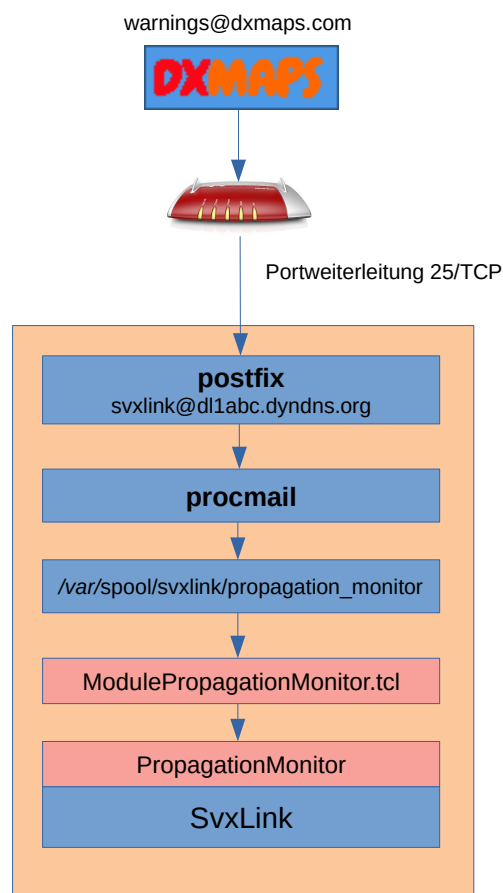


Bild 44: Prinzipieller Ablauf beim Empfang einer eMail von dxmaps.com



Wenn noch nicht geschehen, ergänzen Sie die `/home/svmlink/.procmailrc` wie folgt:

```
# warnings@dxmaps.com
:0:dxmaps.lock
* ^From.*warnings@dxmaps.com
/var/spool/svmlink/propagation_monitor/vhfdx/
```

Procmail erzeugt im o.g. Verzeichnis eine weitere Verzeichnisstruktur (z.B. *new*, *cur*, *tmp*). Eingegangene eMails von warnings@dxmaps.com werden im Verzeichnis *new* gespeichert. Da nur die Subject-Zeile ausgewertet wird, ist eine Bearbeitung des eigentlichen Mail-Bodys nicht erforderlich. Bitte beachten Sie, dass die mitgelieferte *ModulePropagationMonitor.tcl* noch nicht damit umgehen kann.

Passen Sie die `/etc/svmlink/svmlink.d/ModulePropagationMonitor.conf` Ihren Anforderungen an:

```
[ModulePropagationMonitor]
NAME=PropagationMonitor
PLUGIN_NAME=Tcl
ID=10
TIMEOUT=10
SPOOL_DIR=/var/spool/svmlink/propagation_monitor
```

Registrieren Sie die eMail: svmlink@dl1abc.dyndns.org auf der Webseite <https://www.dxmaps.com/spots/warnings.php>

Ein Bestätigungslink wird danach an diese eMail-Adresse gesendet (`/var/spool/mail/svmlink/*`). Diesen Link müssen Sie über einen Webbrowser einmalig aufrufen, damit ist die eMail-Adresse registriert. Falls Sie auf dem SvxLink-Node keine eMail erhalten prüfen Sie die Konfiguration Ihres eMail-Servers auf dem SvxLink-Node. Mögliche Fehler können Sie dem File `/home/svmlink/procmail.log` entnehmen.

Starten Sie danach svmlink neu.

Für die folgenden Module ist ein Text-to-Speech-System (TTS) erforderlich. In dieser Anleitung wird VoiceRSS (<http://www.voicerss.org>) verwendet für diese sind die Skripte entsprechend angepasst. Laden Sie die voicerss-perl-Lib von hier:

<http://www.voicerss.org/sdk/perl.aspx>

herunter und installieren Sie das File nach `/usr/share/perl/Perl-$Versionsnummer/`

Um die empfangenen WAV-Dateien in das für SvxLink benötigte Format zu konvertieren muß man noch sox installieren:

```
sudo apt-get install sox
```

Um VoiceRSS nutzen zu können benötigen Sie einen API-Key den Sie nach einer kostenlosen Registrierung unter <http://www.voicerss.org/login.aspx> erhalten. Dieser API-Key ist in den Contribution-Perl-Files

```
/usr/share/svmlink/make_DWD.pl
/usr/share/svmlink/make_announcement.pl
/usr/share/svmlink/make_KatWarn.pl
```

jeweils in der Zeile

my \$apikey = "hier API Key von VoiceRSS eintragen";
einzufügen.

7.2. KatWarn

Der Katwarn-Dienst wird inzwischen von einer ganzen Reihe Landkreise und kreisfreien Städten angeboten, weitere Informationen finden Sie hier: <https://katwarn.de/>

Im Bild 45 ist der Ablauf der Benachrichtigung vom Versand bis zur Ausgabe auf dem SvxLink-Node dargestellt.

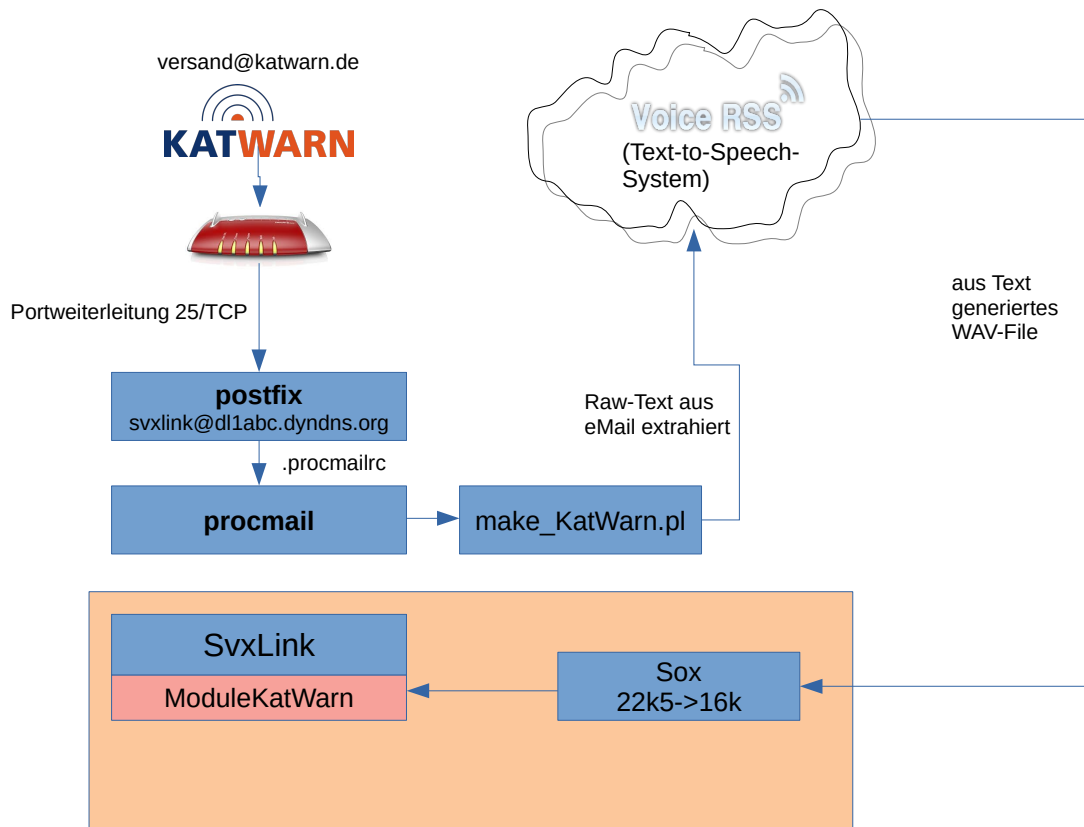


Bild 45: Benachrichtigungsverlauf bei einer KatWarn-eMail

	Meldung der zuständigen Behörde
	Bevölkerungswarnung: Sonderfall
	Postleitzahl: 06108
	gültig ab sofort Ausgabezeitpunkt: 9.10.2019, 15:21 Uhr
	Stadt Halle(Saale) meldet: Warnung Sonderfall. Details: Zur Amok-Lage: Die Stadt erneuert den dringenden Hinweis an die Bevölkerung bis auf weiteres, Wohnungen und Gebäude nicht zu verlassen beziehungsweise an sicheren Orten zu verweilen. Die Vertreter der Medien werden erneut gebeten, die Meldung Gebäude und Wohnungen bis auf weiteres nicht verlassen, zu veröffentlichen.
	Schalten Sie Rundfunk und Fernsehen an.
Diese Nachricht wurde von Katwarn im Auftrag der örtlich zuständigen Behörde versendet. Für Rückmeldungen und Anregungen nutzen Sie bitte info@katwarn.de . DocId: 1570627267167.0@kwrn	

Bild 46: Beispiel für eine eMail von KatWarn

Diese eMails sind multipart-Mails aus denen der weiter zu verarbeitende Text per Perl-Routinen extrahiert werden muß.

Wenn noch nicht geschehen, ergänzen Sie die `/home/svmlink/.procmailrc` wie folgt:

```
# KATWARN
:0:katwarn.lock
* ^From.*versand@katwarn.de
|usr/share/svmlink/make_KatWarn.pl
```

Kopieren Sie die Datei `make_KatWarn.pl` aus dem oben erwähnten SvxLink-Contributions-Repository in das reale Verzeichnis `/usr/share/svmlink`. Machen Sie die Datei für den User `svmlink` ausführbar.

Aktivieren Sie das Modul `ModuleKatWarn` in der `svmlink.conf` wie folgt:

```
[RepeaterLogic] oder [SimplexLogic]
...
MODULES=..., ModuleKatWarn, ...
```

Passen Sie die `ModuleKatWarn.conf` Ihren Anforderungen an:

```
[ModuleKatWarn]
NAME=KatWarn
PLUGIN_NAME=Tcl
ID=11
TIMEOUT=10
CALL=DL1ABC
```

ALERT=1
SP00L_DIR=/usr/share/svmlink/sounds/de_DE/KatWarn

Kurzbeschreibung der wichtigsten Parameter:

TIMEOUT=10

Das Modul wird bei Inaktivität (in diesem Fall 10 Sekunden) automatisch beendet.

ALERT=1

Vor der Ausgabe der Sprachausgabe erfolgt die Signalisierung durch einen Alarmton.

CALL=DL1ABC

Es werden nur Ansagen ausgewertet, die das konfigurierte Call im Dateinamen enthalten. Das ist vorteilhaft, wenn eine Svmlink-Instanz mehrere Nodes gleichzeitig ansteuert.

Starten Sie nach Konfigurationsänderungen svmlink neu.

7.3. Unwetterwarnungen vom Deutschen Wetterdienst

Der Deutsche Wetterdienst bietet die Möglichkeit für verschiedene Gefahrenlagen eMails mit einer Warnung an eine vordefinierte eMail-Adresse zu senden. Im Bild 47 ist der prinzipielle Ablauf vom eMail-Versand durch den DWD bis zur Ausgabe auf dem Svmlink-Node dargestellt.

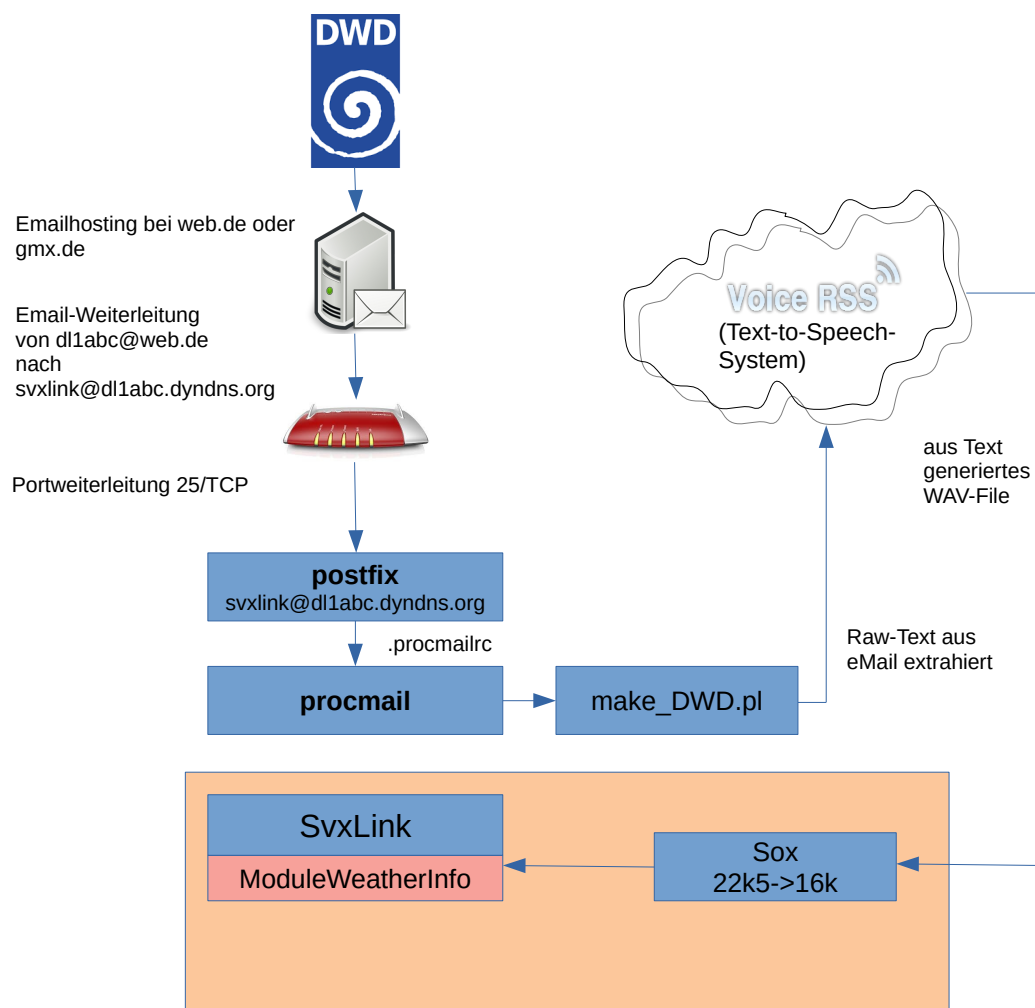


Bild 47: Prinzipieller Ablauf einer eMail-Abarbeitung durch das Modul ModuleWeatherInfo

Erfahrungsgemäß geht das verhältnismäßig schnell, von der offiziellen Ausgabe der Warnung vom DWD bis zur Ausgabe per Funk durch SvxLink sollten maximal 2 Minuten vergehen. Die Registrierung für den Empfang von Wetterwarnungen für einen bestimmten Landkreis oder kreisfreie Stadt erfolgt unter [37]. Die folgenden Wetterereignisse können gewählt werden, dabei stehen jeweils bis zu vier Identitätsstufen zur Verfügung.

- Wind / Sturm / Orkan
- Gewitter
- Starkregen
- Dauerregen
- Schneefall/-verwehungen
- Glätte
- Frost
- Nebel

DWD Deutscher Wetterdienst
Wetter und Klima aus einer Hand

WETTER KLIMA UND UMWELT FORSCHUNG LEISTUNGEN DER DWD

Startseite > Newsletter > Amtliche Warnungen

☐ Ich stimme den Nutzungsbedingungen ([AGB/Disclaimer](#)) zu.
☐ Ich habe die [Datenschutzinformation](#), die Bestandteil dieser Erklärung ist, zur Kenntnis genommen und verstanden. Ich bin damit einverstanden, dass meine Daten zu dem in der [Datenschutzinformation](#) unter Punkt 2.2. beschriebenen Zweck verarbeitet werden.

☒ Wind / Sturm / Orkan
☒ Gewitter
☒ Starkregen
☒ Dauerregen
☒ Schneefall/-verwehungen
☒ Glätte
☒ Frost
☒ Nebel

Suchbegriff:

LANDKREISE

☒ Saalekreis

Auswahl:
Saalekreis

Vielen Dank für Ihr Interesse an den "amtlichen Warnungen" des Deutschen Wetterdienstes!

Hier finden Sie unsere Kriterien/Warnschwellenwerte:
(Warnkriterien für amtliche Wetter- und Unwetterwarnungen des DWD)

Ein Newsletter ist die Kombination aus einem Landkreis und einem Warnereignis.

Anmeldung am Newsletter:

1. Wählen Sie zunächst die Warnereignisse und die gewünschte Intensitätsstufe aus:
☒ = Keine Auswahl
☒ = Amtliche Wetterwarnung (Stufe 1, Gelb)
☒ = Amtliche Warnung vor markantem Wetter (Stufe 2, Orange)
☒ = Amtliche Unwetterwarnung (inkl. Warnung vor extremem Unwetter, Stufe 3+4, Rot+Lila)

WICHTIG:

- Wenn Sie die Stufe ☒ wählen, erhalten Sie auch alle Warnungen der Stufen ☒ und ☒.
- Stufe ☒ enthält alle Unwetterwarnungen der Stufe ☒, nicht aber Warnungen der Stufe ☒.
- Im Vorfeld von amtlichen Unwetterwarnungen Stufe ☒ erhalten Sie ggf. Vorabinformationen.

2. Danach wählen Sie bitte den/die interessierenden Landkreis(e) aus.
 3. Sobald Sie Ihre E-Mail-Adresse in das Formular eingetragen haben, klicken Sie bitte die **Anmelde-Schaltfläche** an. Sie erhalten dann nach wenigen Minuten eine E-Mail.
 4. Über den in der E-Mail enthaltenen Link müssen Sie Ihre Anmeldung an den Deutschen Wetterdienst bestätigen

Bild 48: Registrierung für den Empfang von DWD-Warnungen per eMail

Vergessen Sie nicht die AGB's und Datenschutzbestimmungen zu bestätigen, sonst kann keine Registrierung erfolgen.

Achtung: Die Registrierung von dyndns-Adressen ist leider nicht möglich. Zu empfehlen ist seine persönliche gmx- oder web.de-Adresse zu nutzen und im Portal für die eMail-Adresse @dwd.de eine automatische Weiterleitung auf z.B. svxlink@meinEigenerNode.dyndns.org einzurichten.

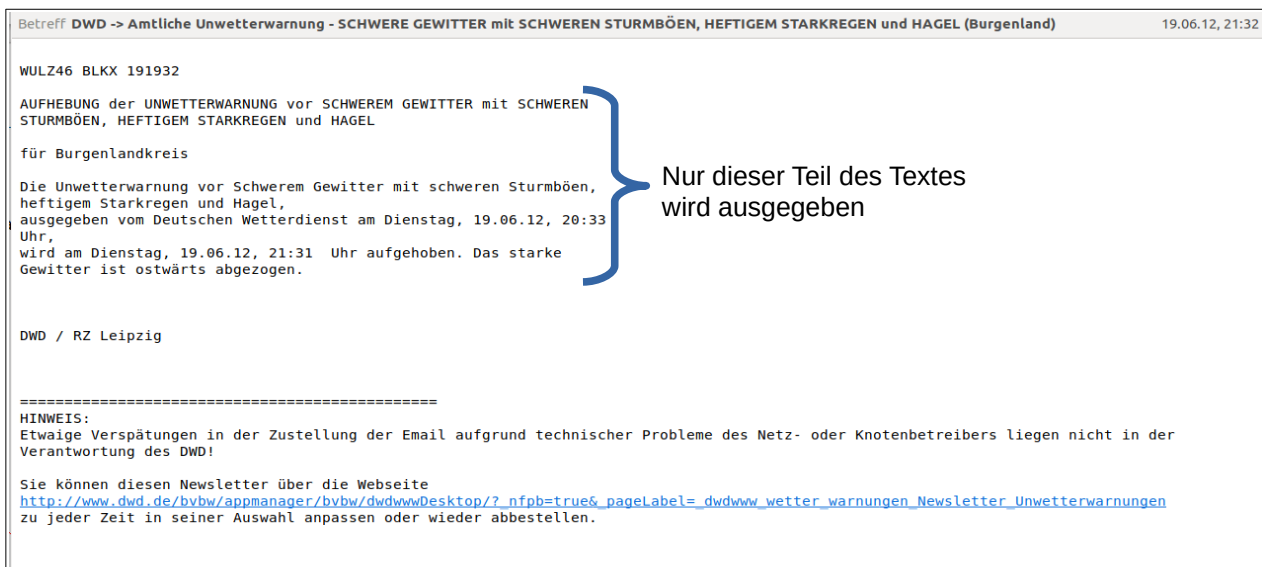


Bild 49: Typische Unwetterwarnung des Deutschen Wetterdienstes

Testen Sie zunächst ob Warnungen vom DWD auf Ihrer gmx.de- oder web.de-Adresse empfangen werden und richten Sie die eMail-Weiterleitung erst danach ein. Wenn Sie keine eMails erhalten sollten überprüfen Sie die ordnungsgemäße Registrierung beim DWD. Wenn noch nicht geschehen, ergänzen Sie die */home/svmlink/.procmailrc* wie folgt:

```
# DWD
:0:dwd.lock
* ^From.*umgeleitete-email@adresse.de
|/usr/share/svmlink/make_DWD.pl
```

Kopieren Sie die Datei *make_DWD.pl* aus dem oben erwähnten SvxLink-Contributions-Repository in das reale Verzeichnis */usr/share/svmlink*. Machen Sie die Datei für den User svmlink ausführbar.

Aktivieren Sie das Modul ModuleWeatherInfo in der *svmlink.conf* wie folgt:

```
[RepeaterLogic] oder [SimplexLogic]
...
MODULES=..., ModuleWeatherInfo, ...
```

Passen Sie die ModuleWeatherInfo.conf Ihren Anforderungen an:

```
[ModuleWeatherInfo]
NAME=WeatherInfo
PLUGIN_NAME=Tcl
ID=12
TIMEOUT=15
DELETE_AFTER=120
ALERT=1
CALL=DL1ABC
SPOOL_DIR=/usr/share/svmlink/sounds/de_DE/WeatherInfo
```

Kurzbeschreibung der wichtigsten Parameter:

TIMEOUT=15
Das Modul wird bei Inaktivität (in diesem Fall 15 Sekunden) automatisch beendet.

ALERT=1

Vor der Ausgabe der Sprachausgabe erfolgt die Signalisierung durch einen Alarmton.

CALL=DL1ABC

Es werden nur Ansagen ausgewertet, die das konfigurierte Call im Dateinamen enthalten. Das ist vorteilhaft, wenn eine SvxFLink-Instanz mehrere Nodes gleichzeitig ansteuert.

Starten Sie nach Konfigurationsänderungen svxlink neu.

7.4. Announcement

Dieses Modul empfängt eine eMail von einer berechtigten eMail-Adresse und erzeugt daraus ein wav, welches dann auf dem Node zur nächsten vollen Minute ausgegeben wird.

Einsatzmöglichkeiten ergeben sich z.B. für die kurzfristige Ansage von Treffen oder bestimmten Funk-Aktivitäten.

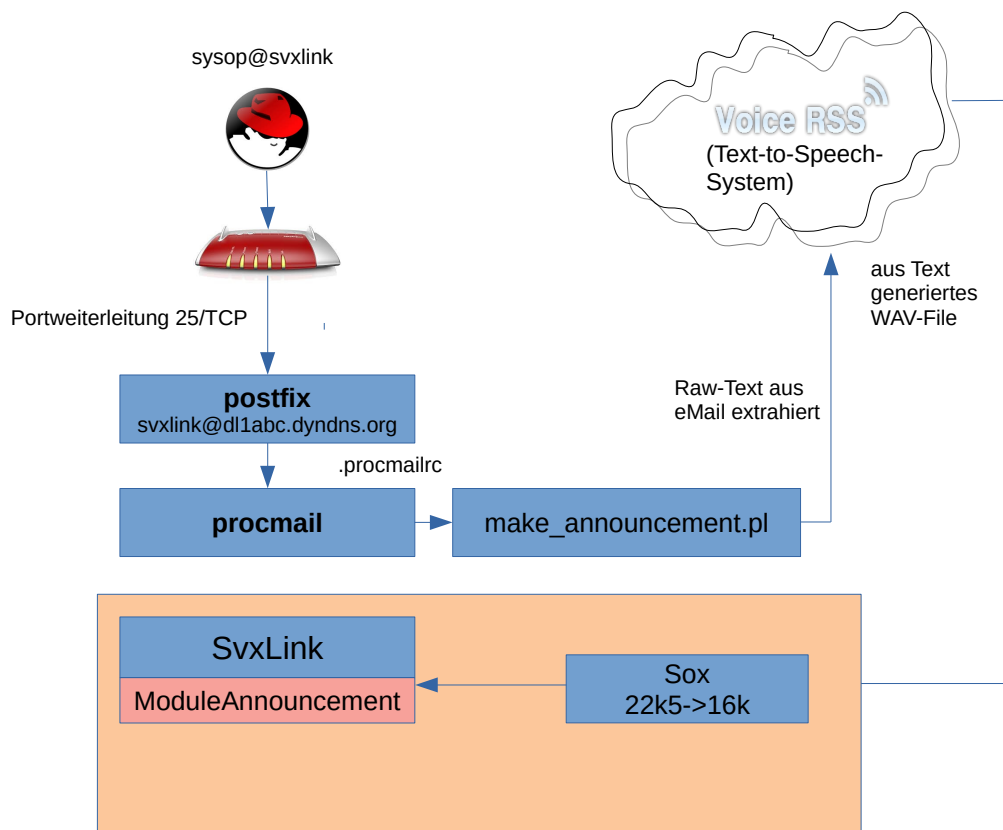


Bild 50: Ablauf bei Erzeugung einer Ansage im Modul „Announcement“

Wenn noch nicht geschehen, ergänzen Sie die `/home/svxlink/.procmailrc` wie folgt:

```
# berechnete eMail-Adresse konfigurieren, von der Ansagen empfangen werden
:0:
* ^From.*meine-berechtigte-email@adresse.de
|/usr/share/svxlink/make_announcement.pl
```

Kopieren Sie die Datei `make_announcement.pl` aus dem oben erwähnten SvxFLink-Contributions-Repository in das reale Verzeichnis `/usr/share/svxlink`. Machen Sie die Datei für den User svxlink ausführbar.

Aktivieren Sie das Modul ModuleAnnouncement in der *svxlink.conf* wie folgt:

```
[RepeaterLogic] oder [SimplexLogic]
```

```
...
```

```
MODULES=..., ModuleAnnouncement, ...
```

Passen Sie die *ModuleAnnouncement.conf* Ihren Anforderungen an:

```
[ModuleAnnouncement]
```

```
NAME=Announcement
```

```
PLUGIN_NAME=Tcl
```

```
ID=22
```

```
TIMEOUT=10
```

```
ALERT=1
```

```
CALL=DL1ABC
```

```
SP00L_DIR=/usr/share/svxlink/sounds/de_DE/Announcement
```

Kurzbeschreibung der wichtigsten Parameter:

```
TIMEOUT=10
```

Das Modul wird bei Inaktivität (in diesem Fall 10 Sekunden) automatisch beendet.

```
ALERT=1
```

Vor der Ausgabe der Sprachausgabe erfolgt die Signalisierung durch einen Alarmton.

```
CALL=DL1ABC
```

Es werden nur Ansagen ausgewertet, die das konfigurierte Call im Dateinamen enthalten. Das ist vorteilhaft, wenn eine SvxLink-Instanz mehrere Nodes gleichzeitig ansteuert.

Starten Sie nach Konfigurationsänderungen svxlink neu.

8. Installation SvxBPortal 2.5 (beta)

Der folgende Abschnitt beschreibt die Installation des SvxBPortals (Dashboard) unter Debian Linux und wurde mit freundlicher Genehmigung von Peter Lindquist / SM5GXQ und Pette Lundberg / SA2BLV übersetzt und in dieses Dokument aufgenommen. Ich selbst habe daran keine Anteil.

Ein wichtiger Hinweis: Das SvxBPortal ist ausdrücklich nicht geeignet um auf einem RaspberryPi oder allgemein auf Systemen mit SD-Karte betrieben zu werden! Das SvxBPortal nutzt eine Datenbank und damit verbunden sind viele Schreibzyklen, die die SD-Karte in kurzer Zeit zerstört!

Rufzeichen	TG#	Ist Sprecher	Überwachte Gesprächsgruppen	Fangen Sie an zu reden	Sprechzeit
SA0CAM	0	NEIN	240 2400 24020 24070		
SA5BJM	0	NEIN	240 2405 24020 240582		
SA6GDS-2	0	NEIN	240 2406 24061 24062 24063 24070 24098		
SK0BO	0	NEIN	24003		
SK0RDZ	0	NEIN	240 2400 24020 24033 24070		
SK2AZ	0	NEIN	240 2402 24020 24021 24022 240211		
SK2AZ-L	0	NEIN	24020 240211		
SK2LY	0	NEIN	240 2402 24020 24021 24022 24033 24034		
SK2RIU	0	NEIN	240 2402 24020 24021 24022 24033 240210		
SK2RWJ	0	NEIN	240 2402 24020 24021 24022 240211		
SK3GK	0	NEIN	2403 24020 24031 24070		
SK3GW	0	NEIN	240 2403 24020 24031		
SK3LH	0	NEIN	240 2402 2403 24020 24021 24022 24033		
SK3RFG	0	NEIN	240 2403 24021 24022 24033 24070 240301		
SK3RIN	0	NEIN	240 2402 2403 24021 24033 24070 240306		
SK3W	0	NEIN	240 2403 24020 24031 24099 240582 2403100		
SK4RGL	0	NEIN	240 2404 24070		
SK5AS	0	NEIN	240 2405 24070		
SK5LW-2	0	NEIN	240515		
SK5LW-70	0	NEIN	240 2402 2405 24020 24022 24070 240515		
SK6IF	0	NEIN	240 2406 24062 24070		

Bild 51: SvxBPortal in der Version 2.5 auf <https://svxbportal.sm2ampr.net/>

8.1. Systemanforderungen

Auf dem System muß folgende Software installiert bzw. verfügbar sein:

- php5 oder spätere Versionen
- MySQL
- Apache2 oder ngx
- crontab
- screen

In jedem Fall sollten Sie vor Beginn der Installation Ihr System auf den neusten Stand bringen:

```
sudo apt update  
sudo apt upgrade
```

8.2. Installation des MySql-Datenbankservers

MySql kann nicht direkt per apt installiert werden, daher muß zunächst das MySql-Repository importiert werden. Rufen Sie den Link <https://dev.mysql.com/downloads/repo/apt/> auf und laden Sie von dort die neuste Version auf Ihr System herunter. Führen sie die folgenden Befehle aus wobei x.x.x.x der von Ihnen heruntergeladenen Version entspricht:

```
sudo apt install gnupg
wget https://dev.mysql.com/get/mysql-apt-config_x.x.x-x_all.deb
sudo dpkg -i mysql-apt-config_x.x.x-x_all.deb
```

Wählen Sie in den folgenden Installationsschritten jeweils die Default-Einstellungen wenn es nicht explizit anders angegeben wird. Setzen Sie jetzt wie folgt fort:

```
sudo apt update
sudo apt install mysql-server mysql-client libmysqlclient-dev
```

Geben Sie ein Passwort ein (welches Sie nach Möglichkeit nicht vergessen), es wird später für den Datenbankzugriff verwendet werden.

Wählen Sie „*Legacy authentication*“ aus.

8.3. Installation des Apache2-Webservers

Führen Sie zunächst folgende Befehle aus:

```
sudo apt install apache2 apache2-doc apache2-utils libexpat1 ssl-cert
```

Danach installieren Sie php:

```
sudo apt install libapache2-mod-php php php-common php-curl php-dev php-gd php-pear php-imagick php-mysql php-ps php-xml
```

Mcrypt muß leider per Hand installiert werden. Dafür sind folgende Abhängigkeiten wie folgt zu lösen:

```
sudo apt install gcc make autoconf libc-dev pkg-config libmcrypt-dev php-pear php-dev
```

Danach führen Sie folgende Befehle aus:

```
sudo pecl channel-update pecl.php.net
sudo pecl update-channels
sudo pecl install mcrypt
```

Jetzt kann mit der Installation von phpMyAdmin begonnen werden:

```
sudo apt install phpmyadmin
```

Wenn Sie aufgefordert werden Datenbankeinstellungen durchzuführen antworten Sie mit Nein bzw. No. Einige Funktionen sind dann nicht verfügbar, diese werden aber auch nicht benötigt. Wenn Sie hier Yes oder Ja wählen versucht die Installationsroutine einen speziellen Nutzer anzulegen um die Verbindung zur Datenbank vom phpMyAdmin aus zu testen.

Die Installalation des Apache Webserver hat das Verzeichnis `/var/www/html` erstellt. Löschen Sie dieses Verzeichnis über den Befehl:

```
sudo rm -rf /var/www/html
```

Installieren Sie jetzt das SvxPortal:

```
cd /var/www
```

laden Sie die Quellen [42] herunter:

```
sudo wget  
https://github.com/sa2blv/SVXportal/archive/refs/heads/svxportal2.5.zip
```

und entpacken Sie diese:

```
sudo unzip svxportal2.5.zip
```

Bennen Sie das neu erstellte Verzeichnis in html um:

```
sudo mv SVXportal-svxportal2.5 html
```

Wenn Sie das wünschen können Sie Ihren eigenen Useraccount als berechtigt zum Datei-Verzeichnis hinzufügen damit Sie nicht jedes Mal die Berechtigung zur Konfiguration dieser Files per sudo erhalten müssen.

8.4. Erstellen der MySql-Datenbank

Führen Sie folgenden Befehl aus:

```
mysql -u root -p
```

Geben Sie das Passwort ein, welches Sie während der Installation der Software angegeben haben. Wenn Sie diese Hürde erfolgreich gemeistert haben sehen Sie ein Prompt (Eingabeaufforderung):

```
mysql:
```

Geben Sie die folgenden Befehle ein, achten sie darauf, dass diese jeweils durch ein Semikolon abgeschlossen werden:

```
CREATE DATABASE Svxportal;  
CREATE USER 'Svxportal'@'localhost' IDENTIFIED WITH mysql_native_password BY  
'Password';  
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, INDEX, DROP, ALTER, CREATE  
TEMPORARY TABLES, LOCK  
TABLES ON Svxportal.* TO 'Svxportal'@'localhost';
```

Jetzt beenden Sie diesen MySql-Eingabemodus per:

```
quit
```

8.5. Erste Einstellungen im SvxPortal

Starten Sie Ihren Lieblings-Webbrowser (Firefox, Opera, ...) und rufen Sie die url auf wobei „Deine-URL“ mit der Ip-Adresse des SvxPortals ersetzt werden muß:

<http://Deine-URL/install.php>

Konfigurieren Sie die folgenden Parameter:

- MySQL Server: 127.0.0.1 (wenn sich die Datenbank auf der selben Maschine befindet)

- MySQL Username: Svportal
- MySQL Passwort: Password (geben Sie das vorher festgelegte Passwort ein)
- MySQL DB: Svportal
- Reflector_proxyserver URL: http://Deine-URL/reflector_proxy
(ersetzen durch die entsprechende URL/IP-Adresse Ihrer Installation)
- IceCast server URL: <http://Deine-URL:8000/live>
(ersetzen durch die entsprechende URL/IP-Adresse Ihrer Installation)
- Sv recording folder: /var/www/htmlsvxrecording
- Reflector address: <http://reflector:8080/status> („reflector“ ersetzen durch die Url des verwendeten Svreflectors)

Es ist sehr wichtig immer die öffentlich URL zu nutzen. Für Versuche kann auch der Hostname verwendet werden allerdings kann man dann nicht feststellen ob es möglicherweise Probleme beim Zugriff von „außerhalb“ gibt. Wenn der svxreflector im selben LAN oder auf dem selben Host läuft, dann kann hier der lokale Hostname als Reflektor-Adresse konfiguriert werden. Die Web-Clients greifen nur auf das Webportal zu einschließlich des Reflector-Proxys. Das Portal selbst erhält die json-Dateien vom Reflector.

In dieser Anleitung wird die Installation von IceCast nicht behandelt. Wenn Sie die Live-Wiedergabe auf Ihrem Portal wünschen dann Sie benötigen hierfür eine SvLink-Instanz und DarkIce für jede einzelne TG. Falls Sie das nicht wünschen dann können Sie das gesamte Monitoring-Menü in der Datenbank deaktiviert werden.

Aktivieren Sie den Webserver in der Datei *svxreflector.conf* wobei der darin konfigurierte Webserver-Port dem oben aufgeführten Port entsprechen muß. Hier ein Beispiel:

```
[GLOBAL]
#CFG_DIR=svxreflector.d
TIMESTAMP_FORMAT="%C"
LISTEN_PORT=5300
#SQL_TIMEOUT=600
#SQL_TIMEOUT_BLOCKTIME=60
#CODECS=OPUS
TG_FOR_V1_CLIENTS=999
#RANDOM_QSY_RANGE=12399:100
HTTP_SRV_PORT=8080
```

Starten Sie danach den svxreflector neu:

```
sudo systemctl restart svxreflector
```

Klicken Sie jetzt auf den Button „Continue“. Wenn alles soweit funktioniert, erscheint eine andere Webseite. Als erstes kopieren Sie den Inhalt der Stage 1 box.

Editieren oder erstellen Sie die/eine neue Datei: */var/www/html/config.php*

und kopieren Sie den Inhalt in diese Datei, danach bitte Abspeichern. Auf die selbe Weise verfahren Sie mit dem Inhalt der Stage 2, kopieren Sie den Inhalt in die Datei:

```
/var/www/html/reflector_proxy/config.php
```

Drücken Sie auf die „SelfCheck“-Schaltfläche im Browserfenster.

Es startet eine Überprüfungsroutine, welche den Datenbankzugriff und die Proxy-Einstellungen testet. Weiterhin wird geprüft, ob ein json-File vom SvReflector geladen werden kann.

Installieren Sie Screen:

```
sudo apt install screen
```

Starten Sie es als Hintergrundprozess:

```
screen -d -m bash -c 'cd /var/www/html; watch -n 20 php station_heartbeat.php;'
screen -d -m bash -c 'cd /var/www/html; watch -n 1 php logdaemon.php;'
```

Das muß außerdem als crontab-Eintrag erfolgen. Dort muss jeder Zeile ein @reboot vorangestellt werden. Bei nachfolgenden Neustarts werden sie von nun an automatisch gestartet.

Mit dem Befehl

```
screen -x
```

können Sie sich aktive Sitzungen anzeigen lassen und nachschauen ob der Dienst ordnungsgemäß läuft. Um sich in eine screen-Session „einzuklinken“ führen sie

```
screen -x nbr
```

aus. Beenden Sie screen mit CTRL -A D. CTRL -C beendet den aktuellen Prozess.

8.6. Administrator Interface

Klicken Sie auf „Admin interface“ und loggen Sie sich ein. Bitte daran denken das Passwort zu ändern oder mit einem eigenen Administrator-Account zu ersetzen.

- Username: Svportal
- Password: svportal

Gehen Sie jetzt zum „Settings“-Tabulator.

Konfigurieren Sie „Externe Dokumentations Webseite“: Yes bzw. Ja
Die Dokumentation zum System findet sich hier:

```
http://Deine-URL/system_description/SystemDescription.htm
```

Ändern Sie die anderen Einstellungen nach Ihren Erfordernissen. Das Administrations-Portal kann auch über die Portal-Startseite erreicht werden. Loggen Sie sich zunächst ein, weitere Einstellungsmöglichkeiten sind über phpMyAdmin verfügbar.

8.7. Sprachen

Nur einige Sprachen werden derzeit unterstützt. Das Portal versucht selbst herauszufinden welche Sprache möglicherweise vom Nutzer verwendet wird. Aber natürlich kann der Nutzer die Sprachumgebung manuell konfigurieren.

Das SvPortal verwendet *gettext* um die benötigten Übersetzungen aus den Sprachdateien umzusetzen. Um ganz sicherzugehen sollten Sie die folgenden Pakete installieren:

```
sudo apt install locales gettext php-php-gettext
```

Unter Debian müssen die benötigten locale-Einstellungen manuell hinzugefügt werden. Das wird durch das Editieren der Datei:

```
/etc/locale.gen
```

realisiert. Fügen Sie die folgenden Zeilen ein, um es abzuspeichern sind root-Rechte erforderlich (sudo):

```
uk_UA UTF-8
sv_SE UTF-8
nb_NO UTF-8
it_IT UTF-8
de_DE UTF-8
fr_FR UTF-8
tr_TR UTF-8
```

Dann geben Sie ein:

```
sudo locale-gen
locale -a
```

Jetzt sollten die locale Einstellungen sichtbar sein, die Sie zuvor hinzugefügt hatten. Ein Neustart des Systems ist immer eine gute Idee, nachdem Sie umfangreiche Installations- und Konfigurationsarbeiten durchgeführt haben.

Falls Sie „Automatisch Übersetzen“ in Ihrem Browser aktiviert haben sollten Sie das zumindest für dieses Portal deaktivieren weil es die Sprachanzeigen völlig aus dem Tritt bringen könnte.

8.8. Anzeigen des SvxBPortals

Klicken Sie auf den „Zurück“-Button, jetzt sollte die SvxBPortal-Startseite angezeigt werden. Führen Sie Funkverkehr auf der Frequenz des SvxBLink-Nodes durch und prüfen Sie ob dieser auf der Webseite angezeigt wird.

8.9. Tips

Viele Funktionen des SvxBPortals basieren darauf, dass die Datei *node_info.json* ordnungsgemäß konfiguriert wurde und auch den realen SvxBLink-Node referenzieren.

Wenn Sie beim SvxB-Portal angemeldet sind, kann jeder Knoten-Sysop ein Tool verwenden, um eine korrekte JSON-Datei zu erstellen. Beispielsweise wird ein SvxBLink-Node ohne *node_info.json* nicht in der Liste angezeigt und auch nicht auf der Karte.

Im Administrator-Interface kann bzw. sollte jedem Node und jeder Talkgroup eine eigene Farbe zugewiesen werden. Es funktioniert natürlich auch ohne diese Zuweisung aber auf diese Art hat man dynamische Farbzweisungen.

Dies gilt für die Empfängerliste, die Karte und die Statistikseite. Auf der Seite "Gesprächsgruppen" werden nur verwaltete Gesprächsgruppen aufgelistet. Der SvxB-Portal-Administrator sollte auch die Knoten-Sysops ermutigen, sich selbst zu registrieren und ihre eigenen Konten auf dem SvxB-Portal zu erstellen. Der Admin kann ihnen dann Schreibzugriff auf Informationen zu ihren eigenen Knoten zuweisen. Die Informationen zu jedem Knoten können auch angezeigt werden, indem Sie einfach auf das Rufzeichen des Knotens auf der Startseite des Portals klicken. Wenn Sie angemeldet sind, sieht jeder Sysop das Menüelement „Meine Stationen“. Dort können sie Informationen zu ihren eigenen Knoten erstellen und pflegen. Das Repeater-Login anzufordern funktioniert allerdings nicht unter der aktuellen Version des SvxBReflectors.

Der SvxBPortal-Administrator kann phpMyAdmin verwenden um die Datenbank anzuzeigen oder zu verändern. Geben Sie einfach folgenden Link in Ihren Webbrowser ein:

`http://Deine-URL/phpmyadmin`

Verwenden Sie die Datenbank-root Rechte die Sie bei der Erstellung der MySQL-Installation hinterlegt haben.

9. Sonstiges

9.1. Hinweise für Entwickler (tetra-contrib)

Für Webentwickler sind sicher die Informationen interessant, die im tetra-contrib-Branch zwischen Node und SvxReflector ausgetauscht werden. Die folgenden Event-Messages werden vom Node zum SvxReflector und auch zurück an die verbundenen Nodes gesendet und dort über das svxreflector-Log ausgegeben:

1) Positioning:

Wird vom Endgerät sporadisch gesendet (siehe LIP-Trigger in ETSI), Bsp:

```
[{"last_activity": "1629978333", "lat": 51.457870483398438, "lon": 11.98405647277832, "reasonforsending": 0, "source": "DM0SVX", "tsi": "09011638300170312", "type": 12}]
```

last_activity	die letzte Aktivität (SDS, Aussendung) des Nutzers als UNIX-Timestamp
lat	Latitude als Grad.Decimal
lon	Longitude als Grad.Decimal
reasonforsending	ETSI code
source	Rufzeichen des Nodes, welcher die Übertragung empfangen hat
tsi	TSI des Senders
type	Typ der SDS, siehe ETSI

2) QSO-Start:

Wird zu Beginn eines QSO's und bei jedem Drücken der PTT gesendet, Bsp:

```
[{"call": "NoCall", "dest_issi": 1, "dest_mcc": 901, "dest_mnc": 16383, "last_activity": "1629980825", "source": "DM0SVX", "tsi": "09011638300170312"}]
```

call	Rufzeichen der aussendenden Station
dest_issi	Ziel-ISSI, im Allgemeinen die GSSI bei einem Gruppnruf
dest_mcc	Countrycode des Netzwerkes
dest_mnc	Networkcode des Netzwerkes
last_activity	UNIX-Timestamp des aktuellen Sendezyklus
source	Rufzeichen der empfangenden Station (i.A. das Gateway, das den Ruf empfangen hat)
tsi	TSI der rufenden Station

3) QSO-End:

Wird gesendet, wenn ein QSO beendet wurde, Bsp:

```
[{"members": "NoCall", "qso_active": false}]
```

members	Kommaseparierte Liste der Rufzeichen, die am QSO beteiligt sind
qso_active	true oder false, je nachdem ob das QSO beginnt oder beendet wurde

4) QSO-Info:

```
{"TG": 2626, "aimode": 0, "cci": 3, "dest_issi": 1, "dest_mcc": 901, "dest_mnc": 9999, "gateway": "DB0WIZ", "last_activity": 1634040546, "qso_active": false, "qso_members": "DB20E-1"}
```

TG	aktive Talkgroup
aimode	Air interface mode
cci	cell call identifier (ID des aktiven Rufes)
dest_issi	Ziel ISSI
dest_mcc	Ziel-Landes-Nummer: mobile country code
dest_mnc	Ziel-Netzwerk-Nummer: mobile network code
gateway	Rufzeichen des Gateways über das der Ruf erfolgt
last_activity	Zeit der letzten Aktivität als UNIX timestamp
qso_active	true: aktives QSO, false: QSO beendet
qso_members	Rufzeichen der Teilnehmer des gerade aktiven QSO's

5) USRP-Branch:

```
{"id":"?", "name":"Rx1", "siglev":5, "sql_open":true}
```

id	Id-Nummer des Rx
name	Name des Rx
siglev	Aktueller Signal-Level-Wert, gilt nur bei der Verwendung des Sigleveldetectors als Squelchkriterium, sonst keine Aussage
sql_open	true – Squelch offen, false – Squelch geschlossen

Beispiel für geschlossene Rauschsperr: {"id":"?", "name":"Rx1", "siglev":6, "sql_open":false}

9.2. Hilfe erhalten

Um Hilfe bei Problemen zu erhalten gibt es eine ganze Reihe Möglichkeiten. Bevor man Entwickler direkt anschreibt sollte man zumindest bemüht sein, die öffentlich zugänglichen Ressourcen auszuschöpfen. Leider habe ich in den vielen Jahren, in denen ich mich mit diesem Projekt beschäftige schon so einiges erlebt was letztendlich auch dazu geführt hat mich aus den öffentlichen Seiten wie svxlink.de zurückzuziehen.

Es gibt sie noch, die svxlink.de-Webseite, die allerdings etwas in die Jahre gekommen ist. Zumindest sind dort die wichtigsten Parameter erklärt. Es gibt die Ressourcen auf github.com mit den Manual-Pages, die in englischer Sprache die einzelnen Parameter erklären. Weiterhin sind inzwischen einige Telegram-Gruppen gegründet worden, in denen man Hilfe durch andere Nutzer erhalten kann.

Folgende Informationen sollten dabei bitte bereitgestellt werden:

- Auszug aus dem svxlink-Log
- Auszug aus der svxlink.conf
- möglichst detaillierte Beschreibung des Fehlers
- bei speziellen Anforderungen eine kleine Skizze wie der gewünschte Ablauf erfolgen soll

Wenn das alles zu viel verlangt ist sollte man sich ernsthaft Gedanken darüber machen ob das anvisierte Projekt für einen das richtige ist.

9.3. Danksagung

Als Entwickler lebt man nicht im Nirvana und schreibt keine Software zum Selbstzweck. An dieser stelle einen herzlichen Dank an:

- Tobias / SM0SVX für die wunderbare Software SvxLink
- Ionut / YO9ION für unendlich viele Hinweise, Tests, Screenshots die ich verwenden durfte
- Waldek / SP2ONG für die unermütlche Hilfe bei der Erstellung der UsrcLogic
- Volker / SM5ZBS für die vielen Asterisk-Informationen und den gegenseitigen Gedankenaustausch
- Peter / SM5GXQ und Pette / SA2BLV für die Genehmigung zur Übersetzung und Implementierung der Anleitung zu Installation des SvxPortals 2.5 (beta) im Abschnitt 8
- Lawrence / DL1FLW für die vielen Tests, insbesondere, dass ich jederzeit auf seine Systeme drauf konnte um Tests durchzuführen
- Jens / DJ1JAY für's Querlesen und wichtige Hinweise
- Peter / SA2BLV für das Fixen von Fehlern in der pjSipLogic

10. Quellenangaben

- [1] SvxFLink-Contributions
<https://github.com/dl1hrc/SvxFLink-Contributions>
- [2] SvxFLink-Repository von SM0SVX
<https://github.com/sm0svx/svxfLink>
- [3] SvxFLink-Fork von DL1HRC
<https://github.com/dl1hrc/svxfLink>
- [4] KatWarn (Fraunhofer-Institut)
<https://katwarn.de/>
- [5] VoiceRSS / Text-to-Speech
<https://www.voicerss.org>
- [6] DynDNS.com
<https://dyn.com/>
- [7] DX-Maps.com
<https://www.dxmaps.com/spots/warnings.php>
- [8] Halbbautomatische Installationsskripte für die SvxFLink-Installation
<http://svxfReflector.org/svxfLink>
- [9] Installationsbeschreibung Twinkle
<https://wiki.ubuntuusers.de/Twinkle/>
- [10] Installationsbeschreibung Zoiper
<https://www.zoiper.com/>
- [11] FTDI-Selbstbauadapter
<https://vktetra.com/build-your-own-gateway-connecting-to-the-radio/>
- [12] Asterisk-Dialplan (Asterisk-Buch)
<http://das-asterisk-buch.de/1.6/dialplan-grundlagen.html>
- [13] Asterisk-Dialplan (Wiki)
<https://wiki.asterisk.org/wiki/display/AST/Dialplan>
- [14] Tetra-Reflektor-Dashboard von Lawrence/DL1FLW
<http://leipzig2000.dyndns.org:4047/>
- [15] Installation Debian auf RaspberryPi 4
https://linuxhint.com/install_debian_raspberry_pi_4/
- [16] Git lernen in 30 Minuten
<https://lerneprogrammieren.de/git/>
- [17] Dokumentation svxfLink.conf
<https://github.com/sm0svx/svxfLink/blob/master/src/doc/man/svxfLink.conf.5>
- [18] Wiki zum Allstarprojekt
https://wiki.allstarlink.org/wiki/Main_Page
- [19] Asterisk-Seiten von Volker / SM5ZBS
<https://elektronikbasteln.pl7.de/eine-einfache-asterisk-konfiguration-fuer-einen-sip-server-als-telefonanlage>

- [20] Beschreibung von fail2ban
<https://wiki.ubuntuusers.de/fail2ban/>
- [21] FreeDMR UK
<http://www.freedmr.uk/>
- [22] DL-Master FreeDMR
<http://svxreflector.org/hbmon>
- [23] Englische Sprachpaket nach GPL
https://github.com/sm0svx/svxlink-sounds-en_US-heather
- [24] Linatec-Webseite
<https://www.linatec.de/>
- [25] Deutsches Sprachpaket (an das Vorhandensein einer Lizenz gebunden!)
http://svxlink.ham-radio-op.net/download/svxlink-sounds-de_DE-petra.tar.gz
- [26] Reguläre Ausdrücke
https://de.wikipedia.org/wiki/Regul%C3%A4rer_Ausdruck
- [27] Asterisk config als.conf
<https://www.voip-info.org/asterisk-config-alsacnf/>
- [28] MMDVM_Bridge auf github.com
https://github.com/DVSwitch/MMDVM_Bridge
- [29] Analog_Bridge auf github.com
https://github.com/DVSwitch/Analog_Bridge
- [30] AMBEServer für den DV3000-Stick
<https://github.com/marrolld/AMBEServer>
- [31] md380-tools auf github.com
<https://github.com/travisgoodspeed/md380tools>
- [32] Bezugsquelle DV3000-USB-Stick, Wimo
<https://www.wimo.com/de/dvmega-dv30>
- [33] DVINC (Digital Voice Systems, Inc.)
<https://www.dvsinc.com/>
- [34] Ticket bei hampager.de anlegen
<https://support.hampager.de/open.php>
- [35] DAPNET-Wiki
<https://hampager.de/dokuwiki/doku.php>
- [36] Sipnatic-App, Sip-Client
<https://play.google.com/store/apps/details?id=com.sipnetic.app&hl=de&gl=US>
- [37] Registrierungsseite für den Empfang von Wetterwarnungen per eMail vom DWD
https://www.dwd.de/DE/service/newsletter/form/amtliche_warnungen/amtliche_warnungen_node.html
- [38] Italienische Sprachausgaben
https://github.com/dl1hrc/svxlink-sounds-it_IT-chiara
- [39] Englische Sprachausgaben
https://github.com/sm0svx/svxlink-sounds-en_US-heather

- [40] Schwedische Sprachausgaben
https://github.com/sm0svx/svmlink-sounds-sv_SE-elin
- [41] AMBEServer im OpenDV-Framework auf github.com
<https://github.com/dl5di/OpenDV/tree/master/DummyRepeater/DV3000>
- [42] SvXPortal von SM5GXQ und SA2BLV auf github.com
<https://github.com/sa2blv/SVXportal/tree/svxportal2.5>
- [43] Französische Sprachausgaben
https://github.com/F8ASB/fr_FR_Agnes/archive/fr_FR_Agnes.zip
- [44] Konfiguration des Hytera-Repeater für den SIP-Gateway
<https://www.hamnet.hamburg/12-sip-dmr/28-konfiguration-des-hytera-repeater-fuer-den-sip-gateway>
- [45] Hytera RD625 DMR repeater tests, VoIP setup
<http://dp.nonoo.hu/hytera-rd625-dmr-repeater-tests-voip-setup/>