Soundmodem on modern Operating Systems

Thomas Sailer, HB9JNX/AE4WA

August 1, 2000

Abstract

Five years ago I presented drivers for using standard PC's with soundcards as packet radio modems [131. The mainstream CPUs of that era were not quite powerful enough for complex signal processing, so the design at that time had to trade robustness for computational simplicity. Futhermore, operating system preferences have changed since. It is therefore time to rethink the design. In this article, an up to date implementation of an amateur soundcard packet radio driver is presented that features a common source base supporting all major operating systems, and the most common modulation formats.

1 Introduction

Five years ago I presented drivers for using standard PC's with soundcards as packet radio modems [131. Later, I also presented Linux drivers [1 I] and a VxD driver for Windows95 [l2]. At that time, my development PC was a *66* MHz 80486, and PC1 was just born. To keep CPU utilization at an acceptable level in spite of CPU's with slow multiplication, design decisions trading robustness for computational simplicity had to be taken. Today, processors below 500 MHz are getting hard to obtain!, and today's CPU's contain fast multipliers, floating point execution units and multimedia instructions

Operating system preferences have also changed. DOS and early Windows versions either did not have soundcard support at all or their support had too much latency to be usable. Today, all common operating systems support low latency soundcard drivers.

The outline of the paper is as follows. In section 2, the architecture of the 1995 drivers is presented together with its problems. Section 3 briefly presents the architecture of the new driver. Section 4 describes installation and usage of the new driver, and Section 5 discusses how faster transmission could be achieved using unmodified handheld transceivers. Section 6 concludes the article.

2 The 1995 Architecture and its problems

2.1 Audio Input/Output

Five years ago, DOS was a widely used operating system for packet radio. DOS did not have any driver support for soundcards, so the packet radio driver had to implement its own soundcard driver. On the other

hand, there were only two standard register level soundcard interfaces, namely the ISA SoundBlaster and the WSS (Windows Sound System) register interface. Virtually all soundcards on the market supported either or both of these pseudo standards.

Windows did have sound driver support relatively early, but the original support had long intolerable latencies for packet radio applications. Low latency sound input/output was later added mainly to support games under the name “DirectSound”.

Today, PC1 has largely displaced the ISA bus, and with it the old register interfaces, since they were tied to the ISA DMA architecture. On the other hand, the importance of DOS fa.ded and all major operating systems feature low latency soundcard drivers.

2.2 Modems

Because of the limited CPU power available five years ago, the modems had to trade performance for computational simplicity. For example, the 9600 Baud FSK modem lacked a receiver filter, resulting in performance degradations from barely noticeable to devastating depending on the receiver used.

Also, the modem code was designed to operate only at a specific bit rate and at a specific sampling rate, limiting the choice of soundcards that could be used.

Because the modem driver included its own soundcard driver, it had to be running in kernel mode. In kernel mode, it is difficult or impossible to use floating point and multimedia instruction sets. The huge differences of the kernel mode environment forced vastly different code bases for the supported operating systems.

3 The new Soundcard Modem Driver Architecture

One important goal of the new soundcard modem driver is supporting all the major operating systems with constrained developer resources. To achieve this, a common source tree for all platforms is employed. Of course certain tasks still require platform specific code, namely the sound driver abstraction (sound driver interfaces vary widely even among UNIX operating systems), and the packet input/output code. The GNU C Compiler [3] was used because it is the defacto standard compiler on mlany UNIX-like systems, is freely available and can also target 32-bit Windows [9]. The Windows threading primitives differ from the standard POSIX ones used under UNIX and Linux. A small library has been used that implements the standard POSIX threading primitives on top of the proprietary Windows ones [6]. To enable portable GUI applications, the GTK [l] widget library has been used, because it is one of the two defacto standard libraries under Linux and UNIX, and GUI builder tool [2] and a port to Win32 [7] exists.

Perhaps the most user visible new feature is the fact that the modems are now fully parameterizable. I have been often asked to support higher FSK bitrates and the many 2400 baud AFSK modes. All these modes are now supported. To make this possible, the modems must now support arbitrary sampling rates. Of course there is a lower limit to satisfy Nyquist, therefore each modem declares its minimal sampling rate depending on the user chosen parameters, and the driver selects the largest and rounds it to the next

After configuration with the `soundmodemconf ig` tool (see below), the modem can be started by running `soundmodem` as `root.` The driver interfaces to the Kernel `mki` s s driver. The result is that the user sees a standard kernel AX.25 network interface that can be used with the kernel AX.25 stack, even though the driver completely runs in userspace. `,soundmodem` also sets up the interface callsign and IP addresses.

4.3 UNIX

Under UNIX, there is no kernel AX.25 support, so a usermode AX.25 stack such as xNOS or WAMPES must be used. `soundmodem` exports a KISS stream on a pseudo terminal pair `(pty/tty),` to which `xNOS` or WAMPES can attach.

4.4 Configuring the Driver

A graphical application named `soundmodemconf ig` makes configuring the driver easy. The tool supports multiple configurations one of which can then be selected at driver ru.ntime. Each configuration supports multiple channels, as depicted in the "DualSpeed" configuration in Figure 2. To ease adjusting the audio levels, `soundmodemconf ig` can transmit a test signal and can display the received signal in an oscilloscope like (Figure 5) or spectrum analyzer like (Figure 5) fashion. Figure 3 shows the application monitoring a 9600 baud FSK channel.

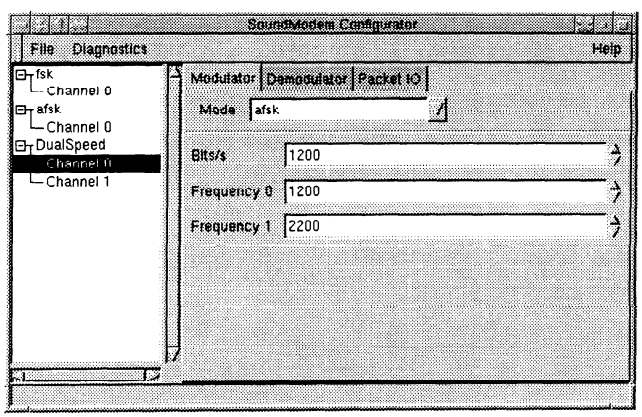

Figure 2: Main Window of Configuration Application

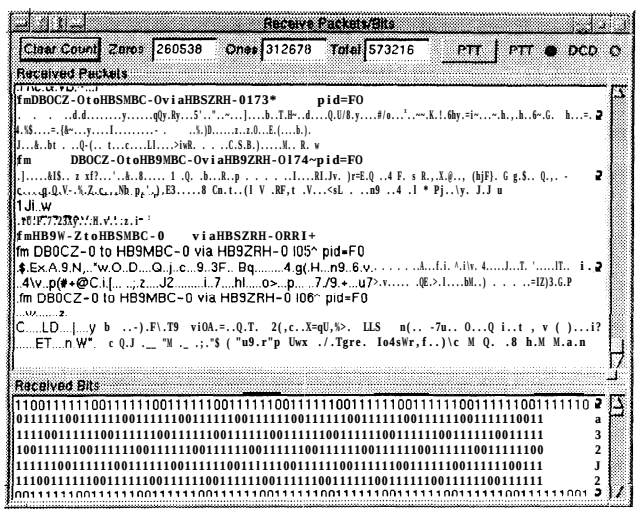

Figure 3: Packet Receive Window of Configuration Application

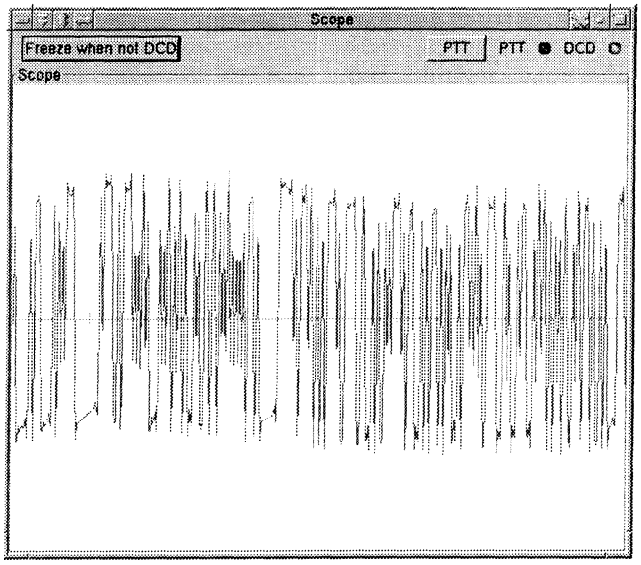

Figure 4: Scope Window of Configuration Application

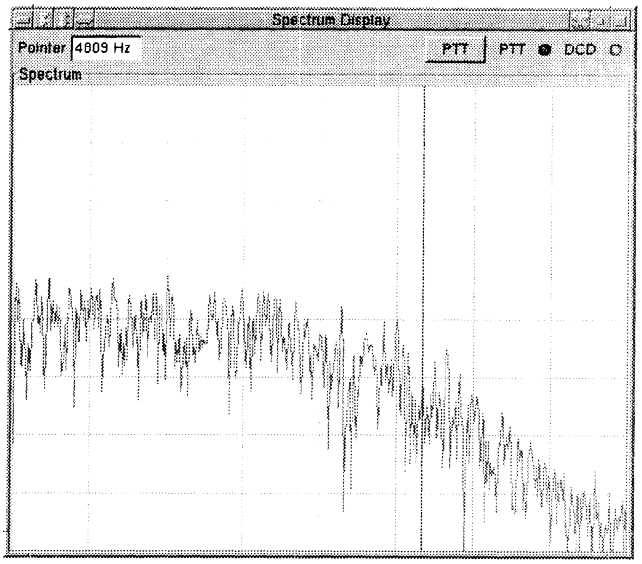

Figure 5: Spectrum Window of Configuration Application

5 Faster Speeds for Handhelds

9600 Bit/s FSK requires transceivers suitable for this modulation. Handhelds usually do not fall into this category. I have often been asked if it was possible to achieve higher transmission using unmodified handheld radios.

I have been working with handhelds from Standard, namely the C558 and the C701. The reasons to choose these handhelds is that I own them and that handhelds from Standard have the worst frequency response, according to measurements performed by DF9IC [151.

One problem when using handhelds for data transmission is their microp:hone amplifier AGC, which introduces nonlinear distortions into the signal. To make matters worse, the Standard handhelds I own do not mute the internal microphone when an external signal is connected, thus making it susceptible to ambient noise.

To achieve some immunity to nonlinear distortion, a constant modulus modulation scheme has been used, namely 8PSK at a symbol rate of 2400 baud centered around a carrier of 1800 Hz.

Because the frequency response of the transceiver introduces intersymbol interference, the receiver has to be designed to cope with it. I have used a constraint length 3 viterbi equalizer to combat the ISI. Constraint length 3 captures most of the channel energy, and is about the maximum that can be implemented with reasonable CPU consumption. For every symbol, $8^3 = 512$ trellis branch,es have to be processed.

Since handhelds are most likely used by end users accessing the backbone network, transmissions will usually be quite short. It is therefore important that the equalizer training is very quick. To achieve this, the modem employs blocks of known symbols. Blocks of 128 data symbols follow blocks of 16 known training

symbols. The training symbols are used for timing synchronisation and frequency offset synchronisation. Furthermore, the channel impulse response is estimated using a maximum likelihood estimator [8].

I have not yet decided on what error correcting code to use, the modem transmits raw HDLC encoded bits at the moment.

The modem is in an experimental stage, and tests over the air were successful if the audio levels have been adjusted carefully. Although not ready for deployment, it is included in the source distribution for those who want to experiment. The modem is named “psk”; and it has not yet been converted to cope with variable sampling rates. It requires 9600 Samples/s and therefore won’t work under Windows.

6 Conclusion and further work

Using a standard soundcard and suitable software as a packet radio modem continues to be a viable low cost solution. In this article, the architecture and the usage of such a soundcard modem driver that runs on all major operating systems currently in use has been presented.

Furthermore, the beginnings of a modulation scheme suitable for achieving higher transmission speeds with unmodified handheld transceivers has been presented as well. Clearly, more work will be needed for it to become useful in production use, namely an FEC layer should be added and the modem needs to be converted to support arbitrary sampling rates.

References

[II The GIMP Toolkit. http://www.gtk.org/.

[.?[Damon Chaplin. GLADE - GTK+ User Interface Builder. http://glade.pn.org.

PI Richard M. Stallman et al. *Using and Porting the GNU Compiler Collect-ion.* Free Software Foundation, 2.95.2 edition, 1999.

PI PC/FlexNet. http://www.flexnet.home.pages.de/. PC/FlexNet.

PI Ulf Haueisen and Gerald Schreiber. Paxon. http://www.paxon.de.

F] Ross Johnson, Ben Elliston, and John Bossom. Pthreads-win32. http://sources.redhat.com/pthreads-win32/, 1999.

VI Tor Lillqvist. GTK+ and GIMP for Windows. http://user.sgic.fi/ tml/gimp/win32/.

[8] Heinrich Meyr, Marc Moeneclaey, and Stefan A. Fechtel. *Digital Communication Receivers, Synchronization, Channel Estimation and Signal Processing.* John Wiley & Sons, Inc, 1997.

[9] Geoffrey J. Noer. Cygwin: A Free Win32 Porting Layer for UNIX@ Applications. http://sources.redhat.com/cygwin/usenix-98/cygwin.html, 1998.

[lo] Thomas Sailer. Soundmodem. http://www.ife.ee.ethz.ch/-sailer/ham/soundmodem.

[l l] Thomas Sailer. “cheaper packet” mit Linux. In 12. *Internationale Packet-Radio-Tagung,* Darmstadt, 1996.

[121 Thomas Sailer. “PacketBlaster 97” - Soundkarten-PR mit aktuellen Betriebssystemen. In *13. Internationale Packet-Radio-Tagung,* Darmstadt, 1997.

[131 Thomas Sailer, HB9JNX. FlexNet-Workshop. 1995.

[141 Wolfgang Winter. ein Windows Packet Programm (WPP). http://dbOexp.de/wpp/.

[151 Wolf-Henning Rech, DF9IC. Equalizer fur 1200 Baud. *Adacom Magazin,* (7):pp. 7 l-73, 1994.