Using Software DSP Solutions to Enhance Weak Signal Communications: A User's Discussion of Linrad, SM5BSZ's Linux PC Radio

Roger Rehr, W3SZ

Abstract. Linrad, a Software Defined Radio created by Leif Asbrink, SM5BSZ, has proven to be an extremely effective tool for weak signal communications. Its noise reduction and weak signal detection abilities are unparalleled, and it provides a comprehensive display of signal information that is vastly superior to all other alternatives. This article will provide a brief overview of DSP processing, and then present an introduction to this revolutionary 'receiver in software' which has replaced, for weak signal work, the conventional receivers at the author's station.

I. Overview

The use of "Digital Signal Processing" or DSP in wireless communications is increasing exponentially due to the burgeoning availability of increasingly inexpensive and capable digital signal processing hardware. The use of digital techniques is moving closer and closer to the antenna. Digital signal processing involves first the conversion of an analog signal to digital form and then the numerical manipulation of the resultant digital signal in some fashion to produce a desired result. As the digital hardware available becomes more sophisticated and cost-efficient, more and more functions that were previously done using analog circuits are being performed with digital hardware. In particular, the use of DSP techniques in amateur radio is rapidly expanding, both in terms of the use of DSP in commercial transceivers and receivers and in terms of homebrew hardware and software construction projects available to and undertaken by hams. The ARRL Handbook since at least its 2000 edition has had an excellent basic introductory chapter on DSP[1]. ARRL publications such as QST[2] and especially QEX[3][4][5][6][7] have featured excellent articles on the subject during the past 2 years. A wealth of information is available on the Internet[8][9][10]. A list of print and Internet resources on DSP is found at the end of this article. This article will provide a brief overview of DSP techniques in Amateur Radio, and move quickly to discuss the excellent Software Defined Radio[11] known as Linrad (short for Linux Radio), created by Leif Asbrink, SM5BSZ. The discussion will be from the viewpoint of a user of the software who is avidly interested in weak signal communications, but not particularly proficient in computer programming, digital theory, or RF electronics. The goal of the article is to help the reader understand what Linrad can do, and to provide a guide to successfully implementing it for aiding in weak signal communications.

II. Why DSP?

I began using DSP techniques because of my interest in doing 144 MHz EME in a very noisy RF environment, and later found that DSP was also very helpful in terrestrial weak signal VHF, UHF, and microwave communications. EME is truly "weak signal" communications. The "typical" round-trip path loss when the moon is at perigee (closest to the earth) is approximately 251.5 dB at 144 MHz. If you consider a system where maximum legal power is present at the antenna, the system starts with 31.76 dBW transmit power. If the antenna array has 19 dB gain, then the signal leaving the antenna will be 51 dBW. The signal arriving back from the moon at the receiving antenna will be on the order of -200 dBW. If the receiving antenna also has 19 dB gain, the signal arriving at the preamplifier on the mast will be -181 dBW. If the antenna has a noise temperature of 200 K, the preamplifier has a noise figure of 0.5 dB, the subsequent 144 to 28 MHz transverter a noise figure of 1 dB, and each has a gain of 20 dB then the receive system will have a noise floor of -187 dBW if a bandwidth of 250 Hz is used. (As long as the 28 MHz IF is reasonably state of the art, its noise figure is irrelevant as it is divided by the product of the gains of the preamplifier and the transverter when figuring its equivalent noise temperature). Thus the receive system will detect the signal as (-181+187) or 6 dB above the noise. Throw in 1-2 dB for cable loss, and 1 or 2 dB for excess sky noise and excess path attenuation and you may be just 2-4 dB above the noise. The signal-to-noise ratios commonly found for EME communications create a real need for tools such as DSP that can pull very weak signals out of the mud to permit the completion of valid two-way contacts.

III. DSP Toolkit

The role of DSP techniques in EME and other weak signal work is of course to provide substantial improvement in signal reception and decoding (interpretation). There are two approaches to using DSP techniques to increase the success potential of signal reception. The first and more obvious approach is to use DSP techniques to improve the human, aural detection of CW signals. There has been much work in this arena over many years. The second approach is to use DSP methods to provide for automated message detection and decoding of signals that may not even be audible with standard audio processing techniques. These methods have only very recently become widely available to amateur radio operators, and are exemplified by the modes PUA43 developed by Bob Larkin, W7PUA[2], and JT44[12], created by Joe Taylor, K1JT. Successfully using either approach for weak signal VHF/UHF/Microwave work requires considerable skill on the part of the operator. Both forms of 'automated' communication have been accepted by the ARRL as meeting the requirements for their Awards Programs (Reference: Personal Communication to W3SZ, by email, Spring 2002). Thus which technique to use for weak signal communications is a matter of personal preference for each operator. Like other experienced EME operators, I have found that programs such as PUA43 and JT44, both examples of the computer decoding paradigm, could at times receive complete and accurate information when I could not hear the other station, and so at least under some circumstances, the human interface represents a weak link when compared with automated decoding by the computer.

When one is using DSP techniques to improve the accuracy of human decoding of the message, there are several features that we would like to have in our "ideal" DSP program. Specifically, the ideal program should provide:

1. A waterfall display with adjustments possible for color gain, baseline level, visualized bandwidth, frequency bin size, and number of averages per displayed line. A waterfall display is basically a way of displaying the time course of signals that have been received by having one axis (usually the horizontal) represent frequency, the second axis (usually vertical) represent time, and then using color to display signal strength. A properly

designed waterfall used in the correct way will allow one to visually detect signals that are considerably below the audible threshold. This is possible by virtue of both signal averaging and by the use of very narrow frequency bins, both of which increase signal-to-noise ratio. Signal averaging increases the signal-to-noise ratio by the square root of 'n', where 'n' is the number of sign als averaged. This means that averaging two signals increases the signal-to-noise ratio by the square root of 2, or 1.414. Expressed in dB, this would be an improvement of 1.5 dB. Narrowing the bin frequency range increases the signal-to-noise ratio by 'n' where 'n' is the fractional bandwidth reduction. For example, decreasing the bandwidth to ½ of its previous width doubles the signal-to-noise ratio, or increases it by 3 db, all other things being equal. However, because reducing the bandwidth by 50% doubles the time required per acquisition, a net gain of 1.5 dB is realized with this bandwidth reduction. An example of an excellent waterfall display is shown in Figure 1. This illustration is a screen grab from Linrad which displays here a 90 kHz portion of the 2 meter band as received at SM5FRH during the ARRL 2001 EME Contest. You can see many vertical dashed lines; each one of these is an EME station's signal. Although here it is reproduced in black and white, the display looks much better in color as can be seen on my website. All of the waterfall display parameters are easily adjustable in Linrad.

2. A spectral display with the following parameters being adjustable: vertical gain, baseline level, visualized frequency range, frequency bin size, and number of averages per displayed spectrum. A spectrum is just the familiar plot of signal intensity vs frequency for a single point in time. A spectrum is shown just below the waterfall display in the Linrad image of Figure 1. Like the waterfall's pa rameters, the spectral display parameters are easily adjustable in Linrad.

3. DSP audio processing with
   a. variable bandwidth filtering with adjustable pitch
   b. a noise reduction algorithm or noise blanker
   c. binaural receive capability
   d. spur removal designed so that it is useful when in CW mode.

The bandwidth filters that can be created with DSP have the advantages of (1) being immune to the problem of aging-induced changes in component values producing altered filter parameters with time, (2) being very flexible (i.e. easily altered by the user as requirements change), and (3) the fact that they can be designed to much more stringent specifications than is generally practical with analog components. They very much lend themselves to experimentation, as trying a different configuration often just involves just changing a parameter value in software. With Linrad adjusting the bandwidth filter involves just clicking on the graphic filter passband display and pulling the filter window so that it is wider or narrower, and steeper or gentler in its slope. Nothing could be easier!

Binaural receiving methods delay the arrival of part or all of the signal going to one ear. This 'pseudo-stereo' sometimes makes the desired signal seem to pop out of the background. Linrad offers four different receiving modes: normal, binaural, and two different ‘coherent’ receiving modes. These modes are selectable with the click of a mouse.

Digital notch filters can be made much sharper and deeper than analog notches. Linrad will remove many spurs, by pointing and clicking on each of them with the mouse. But as a matter of practicality with Linrad, run with a 20 Hz filter (as I generally use it), there is only one signal in the audio pass band and usually no need for a notch filter. The spur removal algorithm is also useful in cleaning up the waterfall so that there are fewer birdies to hide the desired signals.

When the final link in the receive chain is not human hearing and interpretation but computer analysis, the list of desired software characteristics boils down to three items: user friendliness, accuracy of the final result, and efficiency (speed) of achieving the correct solution.

IV. Linrad Overview

Leif Asbrink, SM5BSZ, has developed a superb weak signal receiver in software, which is named Linrad, short for “Linux Radio”. This receiver is the ultimate DSP tool for optimizing the receive chain where the human is the final link. Here is what he has to say about Linrad, by way of introduction.

“Modern computers have the processing power to outperform conventional radios in receiving signals with poor S/N. Particularly when the poor S/N is due to interferences rather than to white (galactic) noise the computer can remove interference within the narrow bandwidth of the desired signal by use of the information about the interference source retrieved by use of larger bandwidths. The signal processing can be far more clever than what has been possible before. Each interference source can be treated as a signal and the DSP radio can receive AND SEPARATE a large number of signals simultaneously. The DSP radio package is under development with flexibility and generality as important aspects. The DSP-radio for LINUX is designed for all narrow band modulation methods for all frequency bands. To start with the following modes will be included: Weak signal CW (primarily EME), Normal CW, High speed CW (meteor scatter), SSB, FM”. He goes on to say, “The system is designed for flexibility so it can be used for many different combinations of computers, A/D boards and analog radio circuitry. The platform is Linux and the package will typically operate with a 486 computer together with a conventional SSB receiver as the minimum configuration. The current high end operation is with a 4-channel 96 kHz A/D board and a Pentium III providing nearly 2 x 90 kHz of useful signal bandwidth in a direct conversion configuration (stereo for two antennas). When the Linux package is in full operation I will interface it to a modern radio A/D chip and digital data decimation chip. The component cost is very low and there will be an exciting improvement in dynamic range, bandwidth and flexibility. The LINUX PC-radio for Intel platforms will be continuously upgraded to show various aspects of digital radio processing and how they are implemented in the DSP package. The Linux PC-radio is not designed for VHF weak signal only. It is very flexible and designed to accommodate routines for all radio communication modes on all frequency bands. The program can run on a 486 to process 3 kHz bandwidth with almost any sound board. It can also run on a Pentium III with a 96 kHz board such as Digital River Delta44 [this is what I use; now called the M-Audio Delta44 -W3SZ] to produce spectra covering about 90 kHz bandwidth, using two mixers to provide a direct conversion receiver. (For EME it may be easiest to make a direct conversion receiver for a fixed frequency such as 10.7 MHz and put some converter in front of it). This is an ongoing project. The package will provide more than 30 kHz bandwidth with a standard audio board and should be very useful for 10 GHz EME and any other mode where a wide spectrum range has to be searched".

Leif started this project years ago with an MS-DOS PC radio (link http://ham.te.hik.se/~sm5bsz/pcdsp/pcdroot.htm) and has expanded the project and moved it to Linux for reasons of hardware portability (link http://ham.te.hik.se/homepage/sm5bsz/linuxdsp/linroot.htm). Details on how to install and get started using Linrad can be found on Leif's website at http://ham.te.hik.se/homepage/sm5bsz/linuxdsp/linrad.htm.

Getting started with Linrad involves a few simple steps. Figure 2 shows these in diagrammatic form, and the remainder of the text elaborates. First, decide on what type of hardware you want to use for the RF-to-Baseband conversion. Second, get a suitable computer for the RF-to-Baseband hardware you have selected. Third, install Linux. Fourth, confirm the function of or install the svgalib graphics package. Linrad will not run without it. Fifth, confirm the function of or install the nasm (Netwide Assembler) package. Linrad will not install without it. Finally, install and setup Linrad. Then run it.

V. Linrad RF-to-Baseband Hardware

Linrad is the current state of the art for weak signal work. The software receiver needs a baseband input to the soundcard from the RF hardware with a bandwidth at least as wide as the bandwidth of the desired waterfall/spectral display. Given such an input, and using quadrature[13] mixing and sampling, the soundcard sampling rate must be equal to or greater than the desired waterfall/main spectral display bandwidth. If standard (non-quadrature) mixers and sampling are used, the maximal waterfall/spectral bandwidth will be only half the sampling rate of the sound card. Using an M-Audio Delta44 soundcard operating at a sampling rate of 96 kHz with quadrature detection and sampling, the useable waterfall bandwidth available is about 90 kHz. If the soundcard is used at the same sampling rate but with a conventional, non-quadrature mixer, the maximal waterfall bandwidth will be about half that, or 45 kHz.

There are several options available for use as a front-end to Linrad. First of all, it's a simple matter to homebrew a very good front-end, as Linrad does all of the hard work. Here I used a circuit with a couple of TUF-1H mixers from Minicircuits, a 10.7 MHz 1st IF and filter, with a transistor amp for the first IF amp and a low-noise OP amp (AD797) for the 2nd IF amp. This circuit is not quadrature, so I get just about 45 kHz of useful bandwidth with this approach when using it with the Delta 44. I use a computer-controlled 1st LO, so that I have wideband frequency coverage with the receiver; essentially from about 0 to 500 MHz (with appropriate filters at the input for each range, to prevent spurious responses). Figure 3 shows a diagram of the circuit for my homebrew receiver front-end. I have two of these units operating simultaneously, one for the horizontally polarized antenna array elements and one for the vertically polarized elements. Linrad uses both of these to provide reception that is always at the correct receive polarization angle; if the incoming wave is at 37 degrees polarization, that is how Linrad receives it. No more 'lock -out" or signal degradati on due to crossed-polarization. And Linrad does this automatically!

The second option is to use a kit that was introduced by Expanded Spectrum Systems at Dayton this year, called 'The Time Machine" (http://www.expandedspectrumsystems.com/prod2.html). This company's principal personnel include N4ESS and N4ES. The Time Machine is a quadrature mixer Direct Conversion HF receive chain that covers the Amateur Bands in the range of 3.5 to 30 MHz. It mixes the incoming signal down to baseband frequency (in this case, 0-45 kHz each for the I and Q channels), and provides 45 KHz of bandwidth coverage above and 45 KHz below the center frequency, for a total bandwidth coverage of 90 KHz. The output of The Time Machine is connected to the input of the Delta44 sound card. I use two of these units, one for each receive polarization, and use a TUF-1H mixer and computer-controlled 1st LO before the input, to mix the 144 MHz signal down to 10.7 MHz, which is then fed into the Time Machine. Another even simpler approach would be to feed the output of a 144 MHz to 28 MHz transverter into the input of The Time Machine instead of using a homebrew front-end. Expanded Spectrum Systems offers an optional daughter board that allows easy connection of an external LO to The Time Machine. Otherwise, you can use the crystals supplied by Expanded Spectrum Systems to control the LO. In the future they may provide options for Direct Conversion at frequencies up to 144 MHz.

The third (and best-performing) option for a front-end is becoming available piece by piece. Leif has designed a superb front-end for Linrad, consisting of several parts. It basically consists of separate units which provide conversion from 144 MHz to 70 MHz, from 70 MHz to 10.7 MHz, from 10.7 MHz to 2.5 MHz, and from 2.5 MHz to baseband. You can see his design for the latter two of these converters, complete with the circuit board masks on his website at http://ham.te.hik.se/homepage/sm5bsz/linuxdsp/optrx.htm. The 2.5 MHz to baseband converter is available now from Svenska Antennspecialisten AB, whose website is at http://www.antennspecialisten.se/ . The unit is called the RX2500, and I have used it, and its performance is outstanding. Linrad with this front-end is now my main station receiver. Eventually the entire receive chain will be available from Antennspecialisten. For now, you must homebrew your own hardware to get down to 2.5 MHz. That is not a difficult task.

The easiest and quickest way to get your feet wet with Linrad is to use it with a conventional receiver, just feeding the audio output from your transceiver into the input of your soundcard. When you do this, you will be limited to processing signals within a bandwidth no wider than the audio bandwidth of your receiver, but this will get you started very quickly and when you see what Linrad does, you will be more motivated to try one of the other approaches listed above.

VI. Computer Hardware Considerations.

How fast a machine do you need to run Linrad? It depends on the parameters. With my setup, Linrad says that my machine, a 1.4 GHz Pentium 4 is idling 92.4% of the time while it is running. In other words, only 7.6% of its processing power is being used by the program. Leif has a very nice page that discusses timing / computation intensity issues and gives some examples for various hardware combinations. It is on his website at http://ham.te.hik.se/homepage/sm5bsz/linuxdsp/fft1time/fft1time.htm . For using Linrad as a DSP processor for SSB bandwidth audio from the headphone jack of a standard transceiver a Pentium at 60 MHz should be plenty fast. To do 90 KHz bandwidth dual channel processing you should probably have at least a Pentium III operating at 650 MHz or so. For SSB bandwidth processing any duplex soundcard that will run under Linux should do. To get 90 KHz bandwidth, you will need quadrature (I/Q) mixing before the soundcard, and a 96 KHz sampling rate as noted above. The M-Audio Delta44 card has worked well with Linrad. My setup here uses a Dell 8100 Pentium 4 running at 1.4 GHz, with 256MB memory and a 40 GB hard drive. I use an ATI Radeon AGP video card, and an M-Audio Delta44 for input and a Creative Labs SoundBlaster PCI64 for output.

VII. Computer Software Considerations.

You must have Linux as your operating system to run Linrad. I use RedHat 7.2 and have also successfully used older versions of RedHat Linux. Others have successfully used RedHat 8.0, Mandrake 7.0 and 8.0, SuSE Linux 6.3 and 8.0. I set my computer up so that when I want to use Linux I boot from a floppy, otherwise it boots to Windows 98. You can also set it up to dual boot from the hard-drive.

Leif has an excellent roadmap of how to proceed once you have Linux installed and working at http://ham.te.hik.se/homepage/sm5bsz/linuxdsp/linrad.htm . Once you have got Linux working, you need to make sure that you have svgalib-1.4.3 or later installed, or Linrad won't work. Under Linux type 'updatedb'. This will take some time and update your file database. Then type 'locate vgagl.h'. If your computer answers with a message like '/usr/local/include/vgagl.h' then svgalib is probably there and working. If it is not, you need to get it and install it. Leif's sit e has the detailed instructions which you can find from the above link. Once you have svgalib installed, you need to see if you have the program 'nasm' installed. To find out, just type 'nasm' at the command prompt. If your computer says 'nasm: no input file specified' or something like that, you are OK. If it says 'bash:nasm:command not found', then you must find and install nasm. Again, the details are on Leif's website proceeding from the above address.

Once all of this is done, you must install Linrad. Again, this software is obtained from Leif's site, listed above. Once you have downloaded the software, follow Leif's instructions on how to proceed to install it. Once installed, Linrad can be run either from a command line in terminal mode, or from a terminal window with Linux running X-Windows (using Gnome, KDE, etc.). Either way, just go to the Linrad directory and type './linrad' and Linrad should run. It will begin in the 'setup' mode, and will ask you to type 'S" to begin the setup routines . Type 'S". It will ask you to choose a video mode. I use 1024 x 768 (option 12 on my machines). Then it will ask you to choose a font scale.

I use “1”. Then it will ask you to enter a mouse speed reduction factor. I type “100”. Then you will be sent to the main start page. Type “W” to save what you have just done. Then type “enter” or “U” to start the D/A and A/D setup routine. Choose the appropriate input device from the list presented to you. There are many possible choices depending upon your hardware, and specific advice cannot be given here, except to be aware of the sampling rate, the number of channels, the bit size, and whether the device is unidirectional or duplex. After you make your selection and type ‘enter’, it will ask you if the device is RDONLY or RDWR. Choose RDONLY first. If there is no output when this is done, then try again choosing RDWR. Choosing RDWR requires that the input and output sampling speeds are the same, and this will tax slower CPU’s. The output sampling speed is intended to be about 5000 Hz for CW, and being forced to run it at 44100 Hz or higher because that is the input sampling speed and you’ve picked RDWR will be very inefficient. If you are using two soundcards, one for input and one for output, the input soundcard should of course be designated as RDONLY. Next select the appropriate type of interface from among the choices presented, which will depend upon your hardware. Finally, select the optimal input sampling rate for your purposes and conditions. This will be determined by the input bandwidth you want to achieve. If you are using quadrature mixing it will be at your desired bandwidth or slightly higher. If you are using conventional mixing it will be set to at least twice the desired bandwidth. You may gain some additional anti-aliasing protection by going above this, and you may also see a small improvement in dynamic range by oversampling. But your CPU speed may limit you in this regard. You may then be asked to type any key to return to the main menu, or to continue on with configuring the output hardware. Once you get back to the main menu, type “W” again to save your settings. Then type “A” to enter the weak signal CW mode. The first time you do this you will be asked to enter parameter values. For the first test, just repeatedly press “enter” to select the default choices. The default choices may not work if your computer is really slow. In that case, try deselecting the second FFT to speed things up. Doing all of this, you will eventually come to the main Linrad receiver screen. Then you can configure the screen and proceed as described under section IX, below.

If Linrad doesn’t work with the sound drivers that came installed with Linux, you will need to install the OSS drivers, available from http://www.opensound.com . After you install OSS, you will need to recompile Linrad by again running ‘./clean’, ‘./configure’, and then ‘make’ before running Linrad again so that Linrad knows to look for the new drivers.

VIII. Linrad Software Block Diagram

A block diagram of the functionality of Leif’s software is helpful in understanding how it works. Figure 4 is a copy of Leif’s block diagram, taken from his website. The receiver input is at the top left of the diagram and the audio output is at the bottom right. Two input signal paths are shown, one for the horizontally polarized antenna elements and one for the vertically polarized elements. The FFT’s are of course fast Fourier transforms, that take the signal from time domain to frequency domain, and the timf’s are reverse transforms that take the signal from frequency domain back to time domain. The first FFT is used to generate the wideband spectrum display, and signals are separated into two groups: strong signals on the one hand, and weak signals and noise on the other. AGC functions are then performed on the strong signals so that they stay within the desired bit range, and then the signals are subjected to reverse Fourier transformation that puts them back in the time domain. The noise blanker is then applied. The noise blanker is a novel, two stage circuit if the Linrad receiver has been calibrated for frequency and phase response using a pulser unit. (This procedure is not difficult, and is described in detail on Leif's website). The first stage blanker is called the 'clever' blanker. It models the noise and fits to each pulse a 'standard' pulse with amplitude, phase, fractional position, and polarization all calculated to match the actual noise pulse as closely as possible. The standard pulses are then subtracted from the signal waveform, reducing the noise pulses by approximately 30 dB. The 'dumb' second stage blanker then removes all data points for which the total power is above a given threshold. This reduces the noise by approximately 40 dB. The results achieved by this two-stage noise blanker are phenomenal! If the receiver has not been calibrated, then just the "dumb' noise blanker is available, but even this does a very good job because it operates on the time domain signal that has had the strong signals removed. Without these strong signals present it can operate much more effectively than a conventional noise blanker that must operate with these signals present in the passband. After noise blanking is done, the second FFT is performed. This produces the waterfall display, and after the polarization control algorithm is applied, the high resolution display is generated. Automatic Frequency Control is derived from the results of the second FFT. A second reverse FFT is performed, using a decimation filter (sampling only part of the second FFT spectrum). This is equivalent to using a mixer followed by a filter and then resampling the signal. A third FFT provides the baseband display. It is then multiplied by the user-selected filter and then another reverse FFT returns the signal to the time domain for final signal processing. Audio is then sent from the soundcard to either audio amplifier, speakers, or headphones for the final step of human decoding. All of this is explained in very detailed fashion on Leif's website. In addition, Linrad's source code is there.

IX. A Discussion of Linrad's Main Receiver Screen and its Operation

I have found that Linrad does an absolutely superb job of allowing me to hear the desired weak signal hidden in the midst of the all the noise and clutter present at my urban QTH. It does this better than any other receiving system I have ever tried. I generally use it with the filter set at 20-25 Hz. The best way to describe Linrad's operation and features is to discuss a series of screen grabs I made using data recorded at SM5FRH on 144 MHz during the 2001 ARRL EME contest. Across the top of Figure 1 you see the frequency scale in Hz. Here Linrad is set up to cover 90 kHz, which is a reasonable spread for 2 meter EME. With it set up like this, one can see everything that is going on in a 90 kHz slice of the band. The small arrows near the left and right corners at the top of the screen allow adjustment of the frequency width and the center frequency of the waterfall and main spectrum displays (which track together in this regard),.

Below the frequency scale at the top of the screen is the waterfall display, showing signal intensity as a function of frequency horizontally, and as a function of time, vertically. Earlier times are nearer the bottom, most recent times at the top. Decimal minutes are displayed along the left vertical axis. The time display reads 00.00 because this screen was taken using pre-recorded data (this is a minor bug). You can see numerous vertical dashed lines on the waterfall display. Each of these represents an EME signal. A quick count suggests that more than 40 EME signals are seen on the waterfall. Just below the waterfall on the screen is the real-time <u>main spectrum display</u>. Signal strength is the vertical axis and frequency is the horizontal axis, corresponding to the same locations on the waterfall and the frequency calibration at the top of the graph. The little up/down arrows at the bottom left and middle right of this display allow you to adjust the range and center point (baseline), respectively, of the spectrum amplitudes displayed, so that the signals are the right vertical size for best viewing, and centered as you wish on the display. It is much more difficult to pick out weak signals on this display than on the waterfall, and I don't use the spectral display very much. Leif notes that the main purpose of the main spectral display is to aid in noise blanker level adjustments and to show very strong signals that saturate the waterfall display. If the second FFT and AFC are deselected, the main spectral display becomes an excellent spectrum analyzer. On this screen, at about 50.400 KHz on the spectral display you can see a strong signal and a cursor over top of it. This is I3DLI's signal.

Below this on the left as displayed from top to bottom are the boxes to set (by clicking on the box and then typing in the desired values): the number of FFT1 averages per displayed point of the spectrum, the number of FFT2 averages per line of the waterfall, the zero point of the waterfall display, the gain of the waterfall display, the number of averages per displayed point of the baseband window spectrum (this is the display with green horizontal lines), and the number of averages per displayed point of the high resolution spectrum (this is the display with red horizontal lines).

To the right of the lowest two of these boxes is the small <u>coherence graph and signal amplitude box</u>. The coherence graph shows that I3DLI's signal has good phase coherence for automatic CW copying, and his signal amplitude as received here is 47.22 dB averaged over both key up and key down times, 52.57 dB current key down level, and 57.51 dB peak key down level since signal selection by the operator. The colored dots in the upper half of this box show the statistics of the complex amplitude of the baseband signal, using the same color scale as the waterfall display. The distance from the center of the crosshairs is proportional to the signal amplitude. Zero phase angle is to the left. I is along the x axis, and Q is along the Y axis. If the operator has selected a large coherence ratio (here the ratio equals 8; see description in the paragraph describing the <u>baseband display</u> below) and there is no signal (white noise), then the points will scatter evenly in all quadrants since there is no correlation between the phase of the carrier and the instantaneous phase of the total signal in the filter passband. As a result a round area at the crosshair center will be formed. If a perfect signal with no QSB is present, the round area will move a distance along the x-axis that corresponds to the signal amplitude. A perfect CW signal in white noise will produce two circular areas, one at the center corresponding to key up, and one displaced to the right at a distance corresponding to the signal amplitude. If a signal has some chirp and constant amplitude, then the signal will form an arc with constant radius, with the phase drifting symmetrically around zero during the key down period. This can be seen with I3DLI's signal on this screen, for which the phase drifts within approximately +/- 20 degrees while the amplitude is saturated. The horizontal bar below the crosshairs box shows the time duration of CW dashes in relation to the duration that would be optimum for the selected baseband filter. The dots should be near the center of the display if the filter width is set optimally.

Below and slightly to the right of the coherence graph and signal amplitude box is the adaptive polarization control. The software receiver can be set up to receive two channels of data. In this case, one is the signal from the vertical elements of the EME array, and the other is the signal from the horizontal elements. By rotating the line with the mouse you can select any desired receive polarization angle. Or, you can leave this set to automatic or ' adaptive' mode and then the software constantly optimizes the polarization angle and phase. Moving the line on the horizontal bar (green when you can see the colors) changes the polarization from linear to elliptical to circular. I usually leave the polarization control set to 'adapt' and let the computer do the work. The little blue "receive polarization" display that is a part of this control shows that the received polarization angle for I3DLI is 28 degrees, and that Linrad is in the adaptive polarization mode, where it follows the polarization angle and phase of the received signal automatically.

To the left of the adaptive polarization control is the EME Window. Once the EME Window is set up by typing "M" on the main menu and the database files dir.skd, eme.dta, and allcalls.dta are placed in the /home/emedir directory, Linrad will provide data on DX EME stations. I3DLI has been typed into one of the boxes, and the EME window gives his data in green. Because this screen was made using a previously recorded file, the data does not reflect the actual conditions when I3DLI was heard, but rather it reflects the conditions when the file was replayed and this screen recorded. At the time of the playback of the recording of the EME contest, the terrestrial azimuth from W3SZ to I3DLI was 53 degrees. The terrestrial distance to I3DLI's grid of JN65bi was 6825 km. I3DLI's azimuth to the moon was 264.5 degrees, and his elevation -30.1 degrees. W3SZ's azimuth to the moon at the time of the playback was 199.9 degrees, and the elevation was 22.2 degrees. Given that the receive polarization angle of I3DLI was 28 degrees, the calculated optimal transmit polarization angle for W3SZ was 95 degrees.

To the right of the polarization control box is the high resolution display. Here is the important and, really, incredible part of Linrad. By clicking with the mouse cursor at any point on the waterfall (or the main spectrum) you cause that portion of the spectrum to be placed in the high resolution spectrum box and DSP-processed. That is, that portion of the spectrum is DSP-filtered, noise-blanked, and converted to audio frequency so that it appears in your headphones or on your speakers. IT IS POINT AND CLICK RECEIVING!!! Because of the excellent DSP, this is an incredible experience. If you are not clicked on a signal, the receiver is quiet. When you click on a peak, the signal pops into your headphones. To fine tune, you click on the peak in the high resolution spectrum, if need be, to touch up the tuning. On the high resolution display, there is excellent resolution of the signals. You can see that I3DLI is just above (144,)050,400 Hz. The green vertical cursor at the bottom of the high resolution display marks the frequency of I3DLI as tracked by the AFC circuit. On the high resolution display there are centered above this cursor a larger, green peak and a smaller, purple peak. The larger, green peak represents the selected polarization component of the received signal, and the purple peak represents the smaller, orthogonal polarization component. Optimal polarization matching of the received signal is achieved when the purple signal is minimized and the green signal maximized. The narrow gray cursor extending vertically across the high resolution display at 50400 Hz represents the frequency of I3DLI at the time his signal was selected by clicking on it, and the distance between this cursor and the smaller green cursor represents the drift or change in Doppler shift of I3DLI's signal between the current time and the time that the signal was selected by clicking on it. The two tiny "A"s (yellow and blue when you can see the colors) at the bottom left of the high resolution window are for setting the mode by which the levels of the 'dumb' and 'smart' digital noise blankers are adjusted. You have your choice of [-] (no noise blanker), [A]utomatic, or [M]anual for these blanker settings. Automatic means that the blanker level follows the noise floor automatically, but the operator is responsible for setting the level above the noise floor in a way that fits the hardware he is using. Manual means that the blanker level is fixed. The tiny "o" at the right bottom of this display turns on the oscilloscope function that shows the time domain signals at the inputs to the summation devices just before the second FFT is performed (see figure 4, top line, just to the right of center of the figure). The signals are presented showing first the real power spectrum of the signal, and then the real and imaginary components of each polarity of both the weak and strong components of the signal, giving a total of 9 different 'oscilloscope' tracings for each time point. With such a display you can really tell what the blankers are doing and gain lots of other useful information. This is explained in detail on Leif's website, e.g. http://nitehawk.com/sm5bsz/linuxdsp/timf2/timf2.htm gives an example of what can be done with this display.

To the right of the high resolution display, on top, is the baseband display. In the baseband window you can see that I3DLI's signal is nicely centered in a 25 Hz bandwidth filter. The line and 'hump' or inverted 'U' (yellow when you can see the colors) in the baseband display show the filter center frequency, bandwidth, and shape factor in graphical form. If you want a different filter bandwidth or shape factor, you just take the mouse over to the baseband display, and drag the filter curve wider or narrower, and the filter adjusts graphically. THIS REALLY WORKS!! On this display you can see both the center of I3DLI's signal and the keying sidebands within the yellow outline of the filter band pass curve. There are several controls in the baseband window. As we just noted, by dragging the yellow lines with the mouse you can set the filter width and shape factor. There is a red horizontal bar at the left of the window that does not really show up with grayscale reproduction. This is the level or volume control and it is adjusted by clicking it or dragging it with the mouse. Above it is a very bright red 'dot' (actually a short, horizontal line) that indicates the received signal level. It is 'pinned' at the top of the scale, commensurate with I3DLI's usually excellent signal strength. The fact that this is red indicates that I3DLI's signal is so strong that the audio channel has saturated with the selected audio gain level. Reducing the gain will cause this dot to become white in color and to fall below the top of the graph. There are three red vertical bars on the left of the window that are the BFO controls. You can vary the pitch of the received signal without taking it out of the filter pass band or moving it in the display by dragging one of these three bars. The upper bar represents the true BFO frequency. With the expanded frequency scale it is actually far outside the window and therefore cannot be used to set the desired pitch. The lower bars have the frequency scale of the BFO frequency offset contracted by 10 and 100 times respectively, so that at least one control will always remain in the window and be available to set the BFO frequency. There are other controls at the bottom of the baseband display for turning on either an amplitude limiter or expander, for choosing the coherent processing mode, adjusting coherent receive parameters, and altering how the program handles the two signals in a dual polarity receive system. The leftmost of these controls [Exp] indicates that the amplitude expander is turned on. The next box gives a choice of 4 operating modes: normal [off]; binaural CW (one ear delayed) [coh1]; coherent with signal (I signal) in one ear and noise (Q signal) in the other ear [coh2]; and signal (I) to both ears, and noise discarded [coh3] ('coh3' selected in this screen). If the signal is not quite stable enough for coh3, then using coh2 instead may be of some help. I find that running with [Exp] and [coh3] works very well. The next box [Rat 3] sets the ratio between the baseband filter width and the width of a subfilter used to extract carrier information for the coherent processing. The last three boxes at the bottom of the baseband window are very difficult to see in a grayscale image, as they are actually dark blue on a black background. The fourth box [off] or [del] ('off' selected in this screen) toggles on or off the signal delay between the ears. The delay can be activated only when 'coh' and 'X+Y' are not selected. The fifth box [8] is the value of the delay if selected in the previous box. The last box [off] or [X+Y] ('off' selected in this screen) when set to [X+Y] sends one of the received signals to one ear and the other to the other ear.

Below the baseband display and immediately to the right of the high resolution display is the automatic frequency control box. This displays time along the horizontal axis and the received frequency vertically. The upper (yellow) trace is the signal-to-noise ratio of the received signal, and the lower trace (actually two traces, superimposed) is the frequency of the received signal, in green, and the frequency of the DSP LO in white. The boxes at the bottom of this display allow you to set the averaging parameters for the AFC circuit.

Linrad has context-sensitive help screens. If you place the mouse cursor over a control or text field and press the 'F1' key, context-sensitive help will pop up. You can press any key to get back to the Main Receiver screen. If you place the mouse cursor over some "empty space" and press 'F1' you will see the control fields highlighted. You can exit Linrad at any time by pressing the 'escape' key.

If you don't have a receiver with a wide (20-90 kHz) IF bandwidth you can still gain experience with Leif's receiver running in wideband mode. On his website Leif has lots of 90 KHz wide files recorded using Linrad at SM5FRH during the 2001 ARRL EME contest that you can download from http://ham.te.hik.se/~sm5bsz/arrl2001/index.htm . These files are large, ranging from about 90 to about 450 MB, compressed in bzip2 format. FRH1135 is particularly nice and is the first one listed on Leif's page. Once you have his receiver running on your computer you can put the names of these files (once they are uncompressed using bunzip2 or equivalent) in a text 'adfile' that you create in the Linrad directory. This file will direct Linrad to these data files and when you start Linrad you can type 'h' or 'j' and the program will run just as if you were actually receiving this data via your own antennas. You can click on the various signals, and even play with the receive polarization control. It is truly amazing to do this! Leif's main Linrad radio page has links to his many useful pages of explanation, diagrams, screenshots, etc. His website is a real treasure trove.

X. Linrad Performance

I have extensively used Linrad for 144 MHz and higher frequency weak-signal work for more than one year and have gained great admiration for it during that time. I frequently have pulse noise here that can read S7 to S9 on the conventional receivers. The conventional receivers' noise b lankers are not sufficiently good for me to do weak signal or especially EME work when this noise is present, even if I run other DSP programs such as DSP-Blaster (which is very good) with LMS Noise reduction and DSP filters from the receiver headphone output. But Linrad's two noise blankers do an extremely effective job of removing the noise as long as the receiver is not overloaded. In addition, the digital filters used in Linrad are very effective and have very little ringing. I generally run Linrad at 20-30 Hz filter width (usually nearer 20 Hz) with excellent results. With Linrad's AFC turned on one does not experience the problem of the signal drifting out of the very narrow filter passband. The other major improvement I have found with Linrad for EME work is always having the correct receive polarity. Because Linrad is constantly receiving both the horizontal and vertical polarizations and computing the actual received signal polarization and setting its polarization to match, the problem of Faraday Rotation disappears for all practical purposes. Thus one vexing variable is removed from the difficult EME equation. For these reasons I just don't use anything but Linrad for 144 MHz weak signal work anymore. I will sometimes again set up the FT1000MP Mk V or Elecraft K2 and LT2S Mk II at the beginning of an EME contest to compare with Linrad, but Linrad is so clearly superior that I quickly return to using Linrad only. And there is the distinct advantage with Linrad of seeing everything that is going on over the entire segment from 144.010 thru 144.100 MHz, nearly the entire EME band, all at once. Because many of the EME stations use consistent frequencies, I can pretty much tell who is transmitting at any time. And when a new station comes on a frequency, I can QSY there to work him with the click of a mouse. All of this makes using a conventional receiver seem like a terrible step backwards. Linrad has met or exceeded all of my expectations. Operations here at W3SZ during the 2002 ARRL EME contest were typical of my experience with Linrad. Using Linrad as the receiver and running 1.5 kW output with low loss 7/8 inch hard-line from the transmitter to the base of my antenna support structure, and LMR600 from there to the 2 x 2 M2 2MXP20 array, I found that exactly 50% of the stations that I called after I had received Q5 copy of their CQ and callsigns using Linrad as my 144 MHz receiver were able to copy my call. The other 50% could only send QRZ, and were not able to copy my call even though I called for extended periods of time, and varied my transmit polarization periodically to mitigate the possible effects of Faraday rotation. Because I used automated keying, I could not explain this on the basis of my poor 'fist'. So I believe it reflects the superiority of Linrad as a weak signal receiver for 144 MHz EME use as compared to the receivers used elsewhere.

XI. Summary

I encourage you to get started with Linux and try out Linrad. It is a step into the future of radio, and particularly amateur radio, communications. If you do weak signal work, you will find it to be a tremendous step forward. For further information on Linrad and other DSP techniques for weak signal use refer to the links provided in this brief article, or refer to the links given on the web page " http://www.qsl.net/w3sz/start.htm ", or subscribe to the Linrad reflector. To subscribe to the Linrad mailing list, send email to: majordomo@antennspecialisten.se with the following text on the first line of the body of the text: "**subscribe Linrad**". The Linrad mailing list is archived at http://www.antennspecialisten.com/linrad-archive/ . I thank Kohjin Yamada JR1EDE, Joe Kraft DL8HCZ, and especially Leif Asbrink SM5BSZ for their great help with this manuscript, and in Leif's case also for his great patience with me and my many questions as I began using Linrad over the past two years.

Bibliography

[1] The ARRL Handbook for Radio Communications, 2003 Edition. Newington, CT, USA. Chapter 18.

[2] Bob Larkin, "The DSP-10: AN All-Mode 2-Meter Transceiver Using a DSP IF and PC-Controlled Front Panel", --Part 1, Sep 1999 QST, pp 33-41; --Part 2, Oct 1999 QST, pp 34-40; --Part 3, Nov 1999 QST, pp 42-45.

[3] Gerald Youngblood, "A Software-Defined Radio for the Masses", --Part 1, Jul/Aug 2002 QEX, pp 13-21; --Part 2, Sep/Oct QEX, pp 10-18; --Part 3, Nov/Dec QEX, pp 27-36; --Part 4, QEX, in press.

[4] James Scarlett, "A High-Performance Digital-Transceiver Design", --Part 1, Jul/Aug 2002 QEX, pp 35-44.

[5] John B Stephensen, "Software-Defined Hardware for Software-Defined Radios", Sep/Oct 2002 QEX, pp 41-50.

[6] Leif Asbrink, "Linrad: New Possibilities for the Communications Experimenter, Part 1", Nov/Dec 2002 QEX, pp 37-41.

[7] Doug Smith, "Signals, Samples, and Stuff: A DSP Tutorial", --Part 1, Mar/Apr 1998 QEX, pp 3-16; --Part 2, May/Jun 1998 QEX, pp 22-37; --Part 3, Jul/Aug 1998 QEX, pp 13-27; --Part 4, Sep/Oct 1998 QEX, pp 19-29.

[8] See Leif Asbrink's Linrad Home Page at sk7do.te.hik.se/~sm5bsz/linuxdsp/linrad.htm

[9] See Bob Larkin's DSP-10 Home Page at www.proaxis.com/~boblark/dsp10.htm

[10] See my DSP Starter Page at www.qsl.net/w3sz/start.htm

[11] Matt Ettus, "Software Defined Radio", http://www.ettus.com/sdr/sdr_seminar_2002.pdf

[12] See Joe Taylor's WSJT Home Page at pulsar.princeton.edu/~joe/K1JT/

[13] Richard Lyons, "Quadrature Signals: Complex, But Not Complicated", http://www.dspguru.com/info/tutor/QuadSignals.pdf

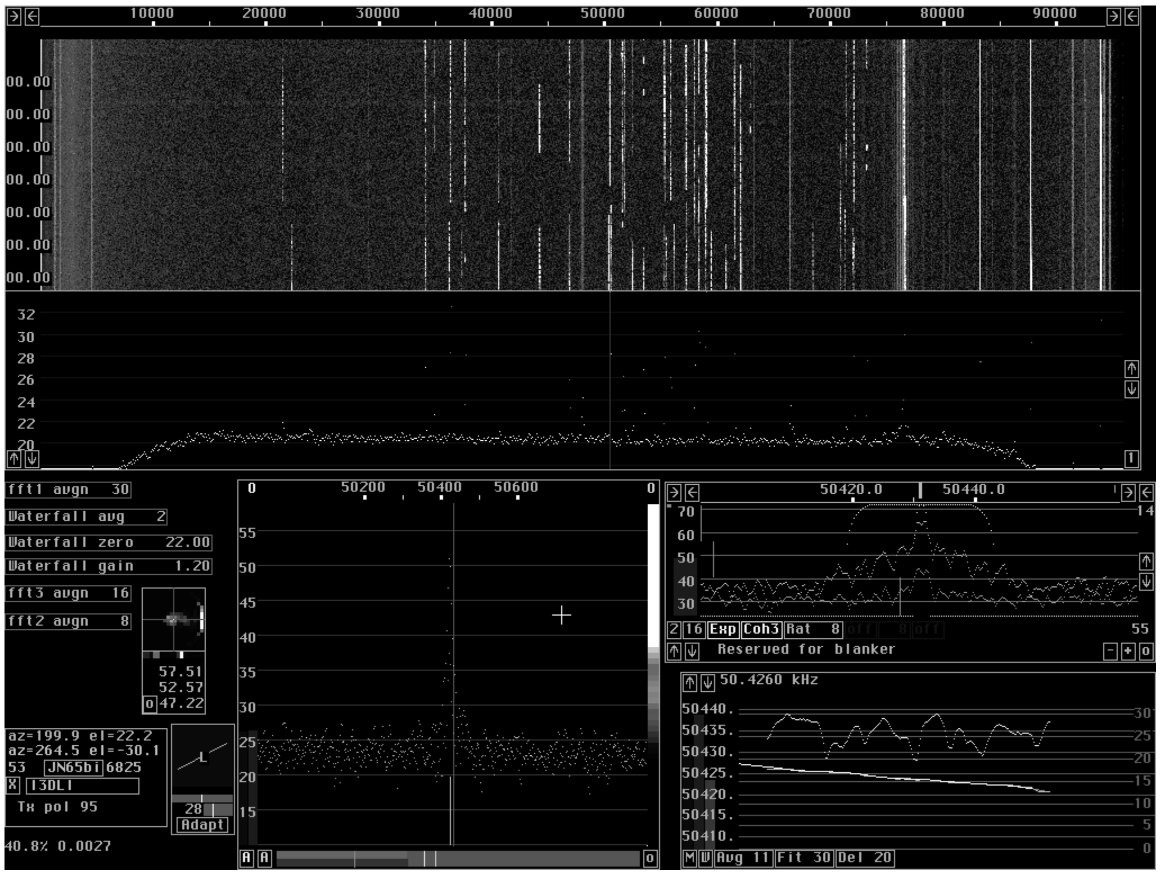

Figure 1. Linrad screen showing a 90 kHz portion of the 2 meter band as received at SM5FRH during the ARRL 2001 EME Contest. On the waterfall display at the top of the screen you can see at least 40 vertical dashed lines; each one of these is an EME station's signal as received by Linrad, SM5BSZ's software receiver. See the text for further details.

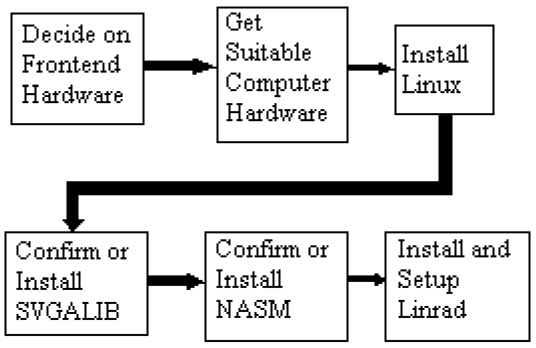
Decide on Frontend Hardware
Get Suitable Computer Hardware
Install Linux
Confirm or Install SVGALIB
Confirm or Install NASM
Install and Setup Linrad

Figure 2. This is a block diagram of the simple steps required to get a Linrad installation up and running. See text for details.

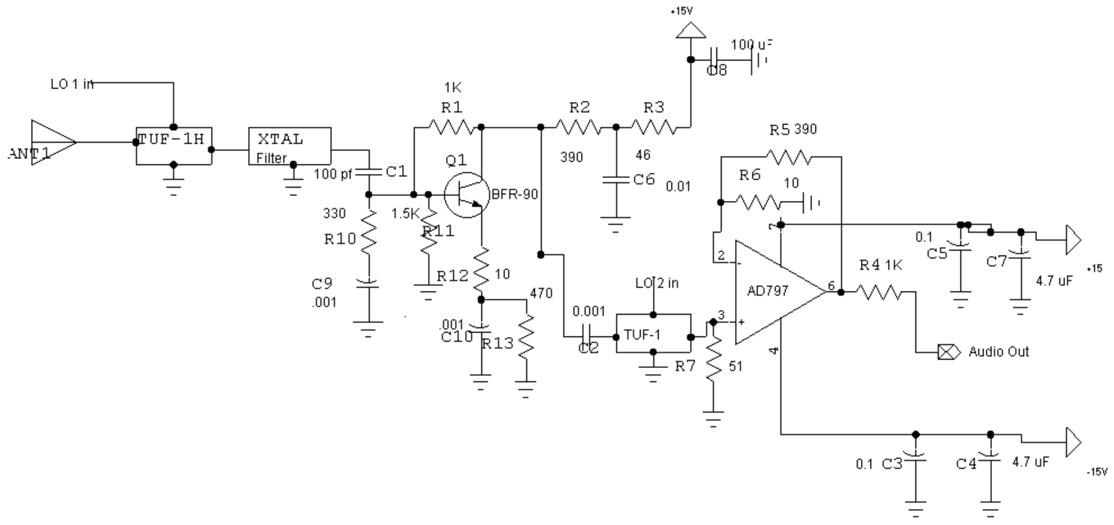

Figure 3. This is a schematic of the front-end I use with the Linrad software receiver. It is a simple configuration, but works well for me. The combination of this front-end and Linrad outperforms the conventional receiver combinations I have tried.

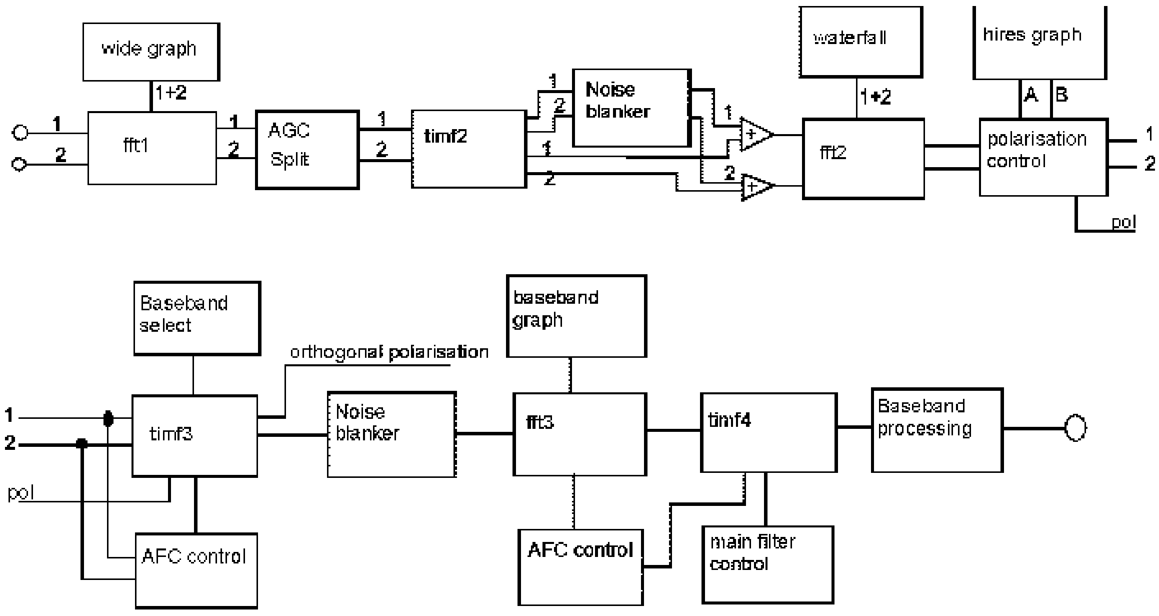

Figure 4. This is Leif SM5BSZ's block diagram of the Linrad Linux PC Receiver. Input is at upper left corner, and audio output is at lower right corner.

Bibliography

[1] The ARRL Handbook for Radio Communications, 2003 Edition. Newington, CT, USA Chapter 18.

[2] Bob Larkin, "The DSP-10: AN All-Mode 2-Meter Transceiver Using a DSP IF and PC-Controlled Front Panel", --Part 1, Sep 1999 QST, pp 33-41; --Part 2, Oct 1999 QST, pp 34-40; --Part 3, Nov 1999 QST, pp 42-45.

[3] Gerald Youngblood, "A Software-Defined Radio for the Masses", --Part 1, Jul/Aug 2002 QEX, pp 13-21; --Part 2, Sep/Oct QEX, pp 10-18; --Part 3, Nov/Dec QEX, pp 27-36; --Part 4, QEX, in press.

[4] James Scarlett, "A High-Performance Digital-Transceiver Design", --Part 1, Jul/Aug 2002 QEX, pp 35-44.

[5] John B Stephensen, "Software-Defined Hardware for Software-Defined Radios", Sep/Oct 2002 QEX, pp 41-50.

[6] Leif Asbrink, "Linrad: New Possibilities for the Communications Experimenter, Part 1", Nov/Dec 2002 QEX, pp 37-41.

[7] Doug Smith, "Signals, Samples, and Stuff: A DSP Tutorial", --Part 1, Mar/Apr 1998 QEX, pp 3-16; --Part 2, May/Jun 1998 QEX, pp 22-37; --Part 3, Jul/Aug 1998 QEX, pp 13-27; --Part 4, Sep/Oct 1998 QEX, pp 19-29.

[8] See Leif Asbrink's Linrad Home Page at sk7do.te.hik.se/~sm5bsz/linuxdsp/linrad.htm

[9] See Bob Larkin's DSP-10 Home Page at www.proaxis.com/~boblark/dsp10.htm

[10] See my DSP Starter Page at www.qsl.net/w3sz/start.htm

[11] Matt Ettus, "Software Defined Radio" http://www.ettus.com/sdr/sdr_seminar_2002.pdf

[12] See Joe Taylor's WSJT Home Page at pulsar.princeton.edu/~joe/K1JT/

[13] Richard Lyons, "Quadrature Signals: Complex, But Not Complicated", http://www.dspguru.com/info/tutor/QuadSignals.pdf