LINRAD

by Leif Asbrink, SM5BSZ

This is a “quick and dirty” copy & paste from Leif’s website to get a printable kind of manual, but far from being complete.
By Ernst, DK1VI (dk1vi@darc.de)

Anyone who likes to improve this is very welcome, but please make it available to the HAM community.

Installing LINRAD (by SM5BSZ) under Microsoft Windows

Make a directory for linrad (default = C:\linrad) to which you unpack the zip file containing linrad.exe, errors.lir and help.lir. That is all, provided that your soundcard(s) are properly installed for Windows. Linrad reads and writes its parameter files on the currently logged directory so you may have several different directories with different names.

In case you have WSE converters or want to use the parallel port to control some other kind of hardware (There have been descriptions of DDS frequency synthesizers with parallel port control in e.g. QST.) you have to make inpout32.dll (32768 bytes) available to Linrad by placing it in the linrad directory or wherever Microsoft conventions suggest it should reside.

If you want to use the SDR-14 or SDR-IQ hardware from Rfspace the file ftd2xx.dll has to be available to Linrad and the SDR-14 or SDR-IQ drive routine has to be installed. In case you already have installed SpectraVue on your computer, everything is already done, but if you do not want to install SpectraVue now you can download this file ftd2xx.zip (266881 bytes), unpack it to a suitable place and instruct the Windows hardware installation tool to search for the drive routines for "SDR-14" or "SDR-IQ" there. You may place a copy of the dll file in the Linrad directory.

If you want to use the Perseus hardware http://www.microtelecom.it the file perseususb.dll has to be available to Linrad and the Perseus hardware has to be installed. Make sure that perseus.exe runs properly before trying Linrad. (One has to use the CD supplied with the Perseus to install the USB driver.) Linrad also needs the sbs files of Perseus. You may place copies of the dll file and the sbs files in the Linrad directory. These files are protected by copyright. All rights belong to http://www.microtelecom.it Each version of Linrad is compiled for a specific version number on the sbs files for Perseus. Go to this page if you want to use Perseus with Linrad and need specific version of the sbs files: sbs files for using Linrad with Perseus Also see **Appendix 5: Perseus with LINRAD**.

To compile linrad.exe from source code under Windows, make a directory C:\linrad and unpack the contents of lirxx-xx.zip into it. Download MinGW (Free software under the terms of the GNU General Public License). MinGW-3.1.0-1.exe is contained in this zip file: mingw.zip (15156728 bytes) Unpack and run the mingw.exe file to install MinGW on your computer. Allow it to install at the default location C:\MinGW Download nasm from http://nasm.sourceforge.net or get nasm.exe from here: nasm.zip (186137 bytes) Unpack and copy nasm.exe to the folder C:\MinGW\bin. Finally log into the C:\linrad directory and run configure.exe to generate the file Makefile (from Makefile.in.) Finally type make to execute the command in make.bat.

Radio hardware

Linrad can receive a radio signal in two formats. As a real signal or as a complex signal.

A complex signal is a pair of two signals I and Q which are obtained from two frequency mixers with a phase shift of 90 degrees between them. To use the complex format two audio channels are needed for one RF signal. It is possible to convert directly from radio frequency to I and Q. No filters are required, the anti alias filters of the sound board will provide the selectivity before the actual A/D conversion. Using complex signals provides twice the bandwidth compared to real signals sampled at the same sampling rate. The double bandwidth is no magic - it is because two audio channels are used for a single RF signal.

A real signal is just an ordinary audio signal such as the one that is fed to the terminals of the loudspeaker of an ordinary SSB receiver.

Look here for some examples of analog hardware to use with Linrad and ordinary sound boards.

The radio hardware needed for Linrad is just a linear receiver. A combination of amplifiers, frequency mixers and filters. An ordinary SSB receiver is one example of a linear receiver. A general discussion on radio receivers might shed more light (or confusion) about what is required to prepare the signal before it is converted to digital form.

Linrad uses the parallel port to control external hardware. One pin is clock, one is data and the remaining pins can be used to select what device the data is for. This hardware control is intended to control frequency synthesizers, antenna rotors and any other hardware one might like to have under computer control.

Making the PC quiet. The standard PC computer radiates at VHF frequencies. This link shows some images of how I made my PC computer quiet by putting it into a box with filters on all wires that pass through the wall of the box.

Blanker performance and calibration using RX2500 With linrad00-50 some improvements are made in the noise blanker. This link shows blanker performance on a calibrated system. RX2500 users having no pulse generator can download a calibration function from here to get started with the smart blanker.

Hardware related parameters

When you start Linrad for the first time you are prompted to select screen, mouse speed, A/D and D/A devices and speeds etc.

By pressing W your will create the file dsp_uiparm and get the same settings automatically when you start Linrad next time. In case you have got your dsp_uiparm from some other system it will work on your computer if the hardware is compatible.

Parameters that depend on receive mode

There are several distinctly different modes of operation for Linrad. Today (July 2001) only the weak signal cw mode is partly implemented. It is possible to set weak signal cw parameters "seriously incorrect" and get a system that is reasonably well adapted for SSB or normal CW.

The mode-dependent parameters define fft sizes and related things. Since the fft's are used as filters they define the size of the FIR filters used when data is decimated. The FFT window functions are the coefficients used by the processing FIR filters so better windows give greater dynamic range (suppression of spurious signals) but they do use more processing time.

A= Weak signal CW A mode for extremely weak signals with modest QSB, primarily EME. This mode has a processing delay of several seconds.

Setup for Weak signal CW mode

Use second fft or not

The second fft is intimately related to the noise blanker. In case you do not need the noise blanker there is usually no need to introduce all the complexity with the second fft, just disable it!

If your computer does not support MMX instructions the second fft is not very efficient and you should probably disable it.

If your computer supports MMX instructions and if you have access to a pulse generator for calibration, and particularly if your analog hardware has a lot of bandwidth you should definitely enable the second fft if you ever have impulse noise problems. Power line interference, car ignition spikes and alike.

The processing that comes after the first frequency conversion is the same regardless of whether the second fft is enabled or not. It is described here: [baseband processing in weak signal CW mode]

Linrad in weak signal CW mode without second fft

Fig. 1 shows the block diagram of linrad in two channel mode when second fft is disabled.

The first fourier transform, fft1 is the dominating processing task. The output of fft1 is used for several purposes:

1: The averaged power spectrum: This is a graph where one can measure signal levels over the full dynamic range of the system.
2: The waterfall graph: This is a graph where it is easy to find weak signals.
3: AFC control: The history of how the power spectra develop over time is used to select the frequency required to downconvert the selected signal so it will be centered in the narrow baseband filter.
4: Polarisation control: The history of how the power spectra develop over time is used to find the optimum combination of the two antenna signals 1 and 2 that will place the signal in output signal A, leaving only noise in output signal B. For an X-yagi receiving a linearly polarised wave the process is equivalent to a mechanical rotation that brings one polarisation in the plane of the incoming wave so the orthogonal set of elements will receive no signal.
5: Decimation filter timf3 is a back transformation of fft1 that uses only a part of the fft1 spectrum. The process is equivalent to a frequency mixing followed by a filter and resampling. Read the data sheet for Analog devices AD6644 or Graychip GC4014 / GC4016 for other ways of doing the same thing (in hardware).
Since the fft1 output is already in place for other purposes it is efficient to use it also for the decimation filter. It is possible to handle many different signals in parallel because timf3 operates with a small fraction of fft1 only producing output at a low sampling speed. This way it will be possible to decode many signals simultaneously and print all the transmissions in ascii on the screen.

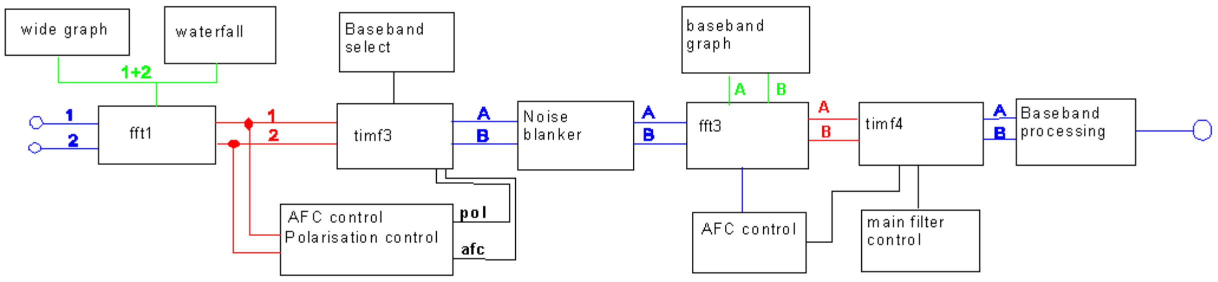
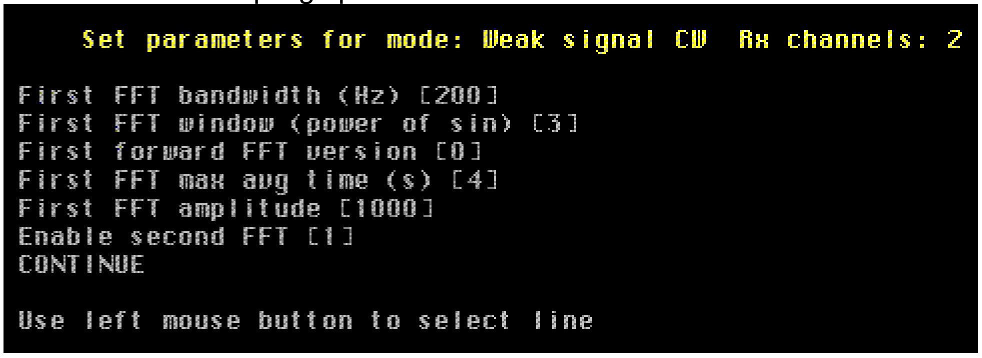


***Fig 1.** Linrad in two channel mode without second fft*

The setup screen for parameters related to the first fft is shown in fig.2. The bandwidth and window parameters affect the size of the first fft. The length of the transform is 1/bandwidth if no window is used and it increases with the power of sin used for the window. The minimum processing delay is of course the time it takes to collect all the points needed in the transform so narrow bandwidth in fft1 causes a noticeable processing delay. Table 1 shows some typical values for bandwidth and window parameters and the transform sizes they give at some different sampling speeds.



```
     Set parameters for mode: Weak signal CW  Rx channels: 2

First FFT bandwidth (Hz) [200]
First FFT window (power of sin) [3]
First forward FFT version [0]
First FFT max avg time (s) [4]
First FFT amplitude [1000]
Enable second FFT [1]
CONTINUE

Use left mouse button to select line
```

Fig 2. *The fft1 parameter setup screen.*

The actual bandwidth of each bin in the first fft is not exactly what you asked for with the bandwidth parameter. What you actually get is something approximately within a factor of two. The transform sizes have to be a power of two.
It is always nice to have a short processing delay so do not ask for narrower bandwidth than you really can use. A bandwidth of 10Hz is sufficiently narrow to find signals that are so weak that it is impossible to copy them.
What window to select depends on the dynamic range requirements and the data decimation ratio.

Selected values		real @ 6000Hz			real @ 44100Hz			complex @ 44100Hz		
Bandwidth parameter	Window parameter	Size	Bw	Delay	Size	Bw	Delay	Size	Bw	Delay
200	3	128	56	0.04	512	103	0.02	1024	103	0.02
20	2	512	12	0.17	4096	11	0.18	8192	11	0.18
5	1	1024	4	0.34	8192	4	0.37	16384	4	0.37
1	1	8192	0.6	1.36	16384	2	0.74	65536	1	1.48

Table 1. *Typical fft sizes, bandwidths and processing delays for different combinations of fft1 parameters. The bandwidth is calculated as 1/t where t is the time between the 6 dB points of the window function.*

When narrow bandwidths are used (large fft1 size) a low order window is sufficient but if the first fft bandwidth is large the window should be of higher order.
The data decimation means picking a part of the spectrum, then transforming that part only backwards. Spurs are produced if there is a signal that has part of its energy within the selected point and parts outside. The spurs will mainly fall at very low and very high frequencies in the back transform so they will to a large extent be filtered out in the next filtering process. Some of the spur energy will fall at the frequency where the desired signal is and those spurs may limit the dynamic range if unsuitable parameters are selected.
The whole thing is quite simple. Select the parameters that make a strong signal as narrow as you want it in order to find weak signals close to it on the main spectrum. The limit is not often linrad, it is the sidebands of strong interfering signals that limit how close you can go before interference is a problem.
To get an idea, check the screen dumps showing the resampling spurs from fft1 back transformation with some different parameter choices. These graphs clearly show that resampling spurs is not a limiting factor for system performance.
When you have decided what bandwidth and window to use, try the different fft versions and select the one that works fastest on your computer. In case you use real input you may select an approximate routine for conversion from real to complex signals. This may save some time but it will also produce a spur which is about 80dB below the main signal, growing towards the spectrum end points. For real input you may use fft1 version 2 to avoid this spur.
Some more information about approximate real to complex conversion can be found here:
Speed and accuracy of approximate real FFT
The parameter First FFT max avg time is used to determine how much memory to reserve for first FFT spectra. This parameter also determines the maximum time the AFC and adaptive polarisation may base the signal tracking on.
First FFT amplitude is a scale factor, 10 times here is 20dB. This scale factor will shift the signal level everywhere within linrad. When second fft is disabled, moving the y scale does not affect processing since floating point arithmetics is used everywhere.
Linrad in weak signal CW mode with second fft enabled.
When the second fft is enabled the first fft is used primarily to split the incoming signal into a weak signals part and a strong signals part.
The first fft produces a full dynamic range power spectrum on the screen and points which are considered strong come in red colour. Red and white points are grouped together in two

transforms. The one with strong signals is reduced in amplitude to fit in 16 bit words while weak signals automatically fit in 16 bits.
Weak and strong signals are back transformed separately to provide two time functions, one of which contains only noise and weak signals on which the noise blanker operates.
The second fft then makes fourier transforms of the sum of the weak and the strong signals.
Figure 3 illustrates linrad running with two rx channels and with second fft enabled. When the second fft is enabled it is essential to set gain levels correctly all the way through linrad. timf2 and fft2 use 16 bit only to take advantage of the very fast 16bit MMX instructions.

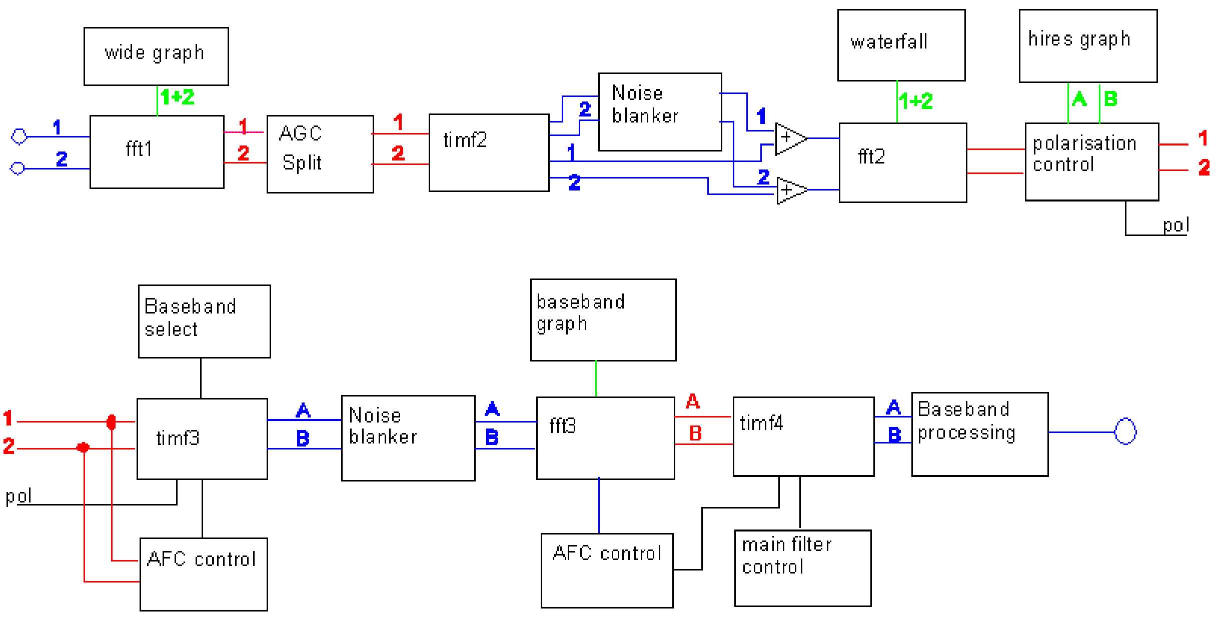
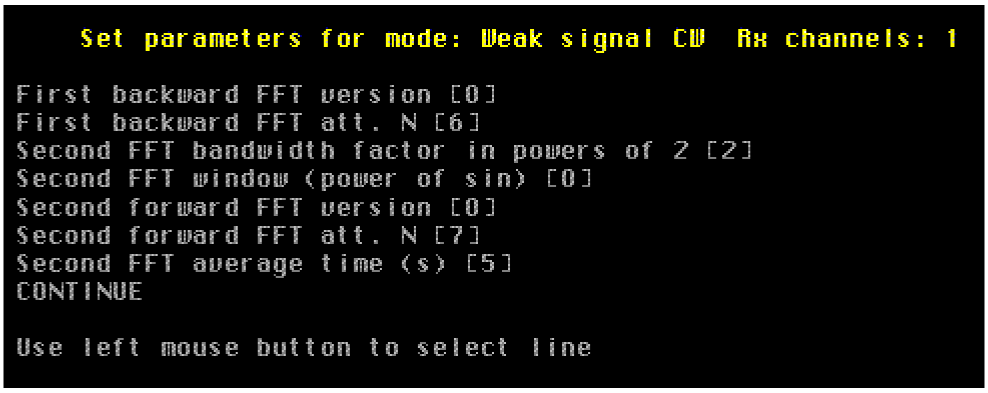


Fig 3. Linrad in two channel mode with second fft enabled

Setting up the first fft When the second fft is enabled, the purpose of the first fft is only to remove strong narrowband signals that would destroy the noise blanker performance if they were left in place. There is no reason to waist computer time doing the first fft too well unless there are very strong signals of very high quality within the passband. Some screen dumps are presented to illustrate this. Typical bandwidth to use for the first fft is 100Hz with a sin to power 3 window. In case the second fft is made very large somewhat narrower bandwidth for the first fft may be useful becayse the ratio between the transform sizes affects the signal level at which a signal is regarded as strong. With a very large ratio between transform sizes some of the desired signals will be strong enough to be routed together with the strong signals and then they will suffer from poor noise blanker performance.
When the second fft is enabled the first FFT amplitude parameter that changes the signal level has to be adjusted together with other parameters to place the signal correctly in the 16 bit format of timf2 and fft2. For details, look at set digital signal levels correctly
The second fft is used for:
1: The waterfall graph This is a graph where it is easy to find weak signals.
2: The high resolution graph This graph is complementary to the waterfall graph. It is an averaged power spectrum and with long averaging times it becomes extremely sensitive. This graph presents the spectrum with one pixel per point while the waterfall only has 1024 points and often has a large number of data points behind each pixel which causes loss of S/N.
3: AFC control The history of how the power spectra develop over time is used to select the frequency required to downconvert the selected signal so it will be cerntered in the narrow baseband filter.
4: Polarisation control The history of how the power spectra develop over time is used to find the optimum combination of the two antenna signals 1 and 2 that will place the signal in output signal A, leaving only noise in output signal B. For an X-yagi receiving a linearly polarised wave the process is equivalent to a mechanical rotation that brings one polarisation in the plane of the incoming wave so the orthogonal set of elements will receive no signal.

5: Decimation filter timf3 is a back transformation of fft2 that uses only a part of the fft2 spectrum. The process is equivalent to a frequency mixing followed by a filter and resampling. Read the data sheet for Analog devices AD6644 or Graychip GC4014 / GC4016 for other ways of doing the same thing (in hardware).



```
    Set parameters for mode: Weak signal CW  Rx channels: 1

First backward FFT version [0]
First backward FFT att. N [6]
Second FFT bandwidth factor in powers of 2 [2]
Second FFT window (power of sin) [0]
Second forward FFT version [0]
Second forward FFT att. N [7]
Second FFT average time (s) [5]
CONTINUE

Use left mouse button to select line
```

Fig 4. *The fft2 parameter setup screen.*

The second fft is never smaller than the first fft. How much bigger it is is set with the bandwidth factor, see fig. 4. The same rules apply for spur generation when fft2 is back transformed as when fft1 is back transformed. To get an idea, check the screen dumps showing the resampling spurs from fft1 back transformation with some different parameter choices. These graphs clearly show that resampling spurs is not a limiting factor for system performance. For the second fft, select a window 0 to 2 depending on second fft size. The second fft does not have any strong signals ever so spurs is less of a problem.
The first backward FFT version is very important. If your computer has MMX, select 1 here. It has a significant effect on processor load. The second forward FFT version is equally important. If you have MMX, select 2 here. The att N parameters control how the signal levels grow while the transforms are computed. These parameters have to be set properly to not saturate the 16 bits or to loos S/N ratio due to quantisation noise. For details, look at set digital signal levels correctly
The parameter Second FFT average time is used to determine how much memory to reserve for second FFT spectra. This parameter also determines the maximum time the AFC and adaptive polarisation may base the signal tracking on.

B= Normal CW Like a "normal" radio. The processing delay is one hundred or a few hundred milliseconds. (no more info available)

C= Meteor scatter A mode for short bursts of very high speed CW. (no more info available)

D= SSB Like a "normal" radio. The processing delay is one hundred or a few hundred milliseconds. (no more info available)

D= FM It should be possible to improve drastically on the S/N one gets from a conventional FM radio. This mode has low priority. (no more info available)

Dynamic range considerations

The dynamic range of the A/D converter is limited. It is essential for the dynamic range of the whole system to set the analog signal level and A/D board gain correctly. (see text directly below)

Do not waste precious dynamic range by incorrect signal levels.

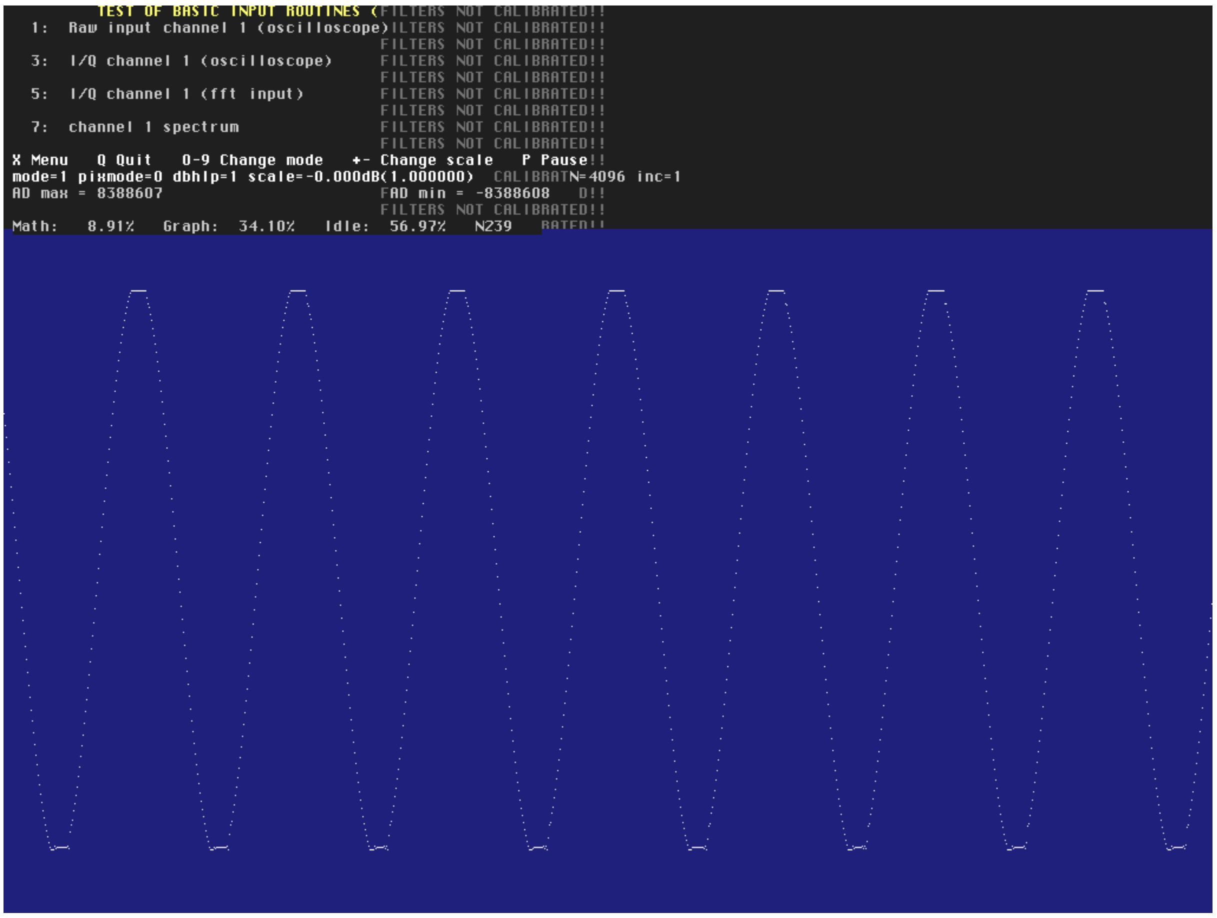
The audio board

Most A/D boards (every board I am familiar to) has computer controlled switches that can be set for different gain levels. If the A/D board sensitivity is set high, the amplifier that precedes the A/D converter on the A/D board will add some noise. Different inputs may behave differently; normally the microphone input is much noisier than the line input.

Once you have successfully got linrad running, start with any selection of parameters that your system accepts and look at the noise floor on the main spectrum (blue dB scale) while running in "Weak CW mode". Remove all signal wires to the board and leave all inputs open. Set all volume controls to minimum and try to select different inputs (line, mic, CD ....) if the board allows you any choice. Note if there is any difference in noise level.

Select the best (lowest noise) input in case there is a difference. Then bring up the volume until you see the noise floor go up by 1dB. Then reduce the voltage gain by a factor of two (6dB) to make the noise floor degradation caused by the amplifier 0.25dB.

Once you have selected an input and the maximum volume that you can use with it, connect a signal source to the input and run linrad in the F=Hardware test mode. Increase the signal level until you see saturation on the screen. It is best to use a sine wave like source, but any signal will do. Fig. 1. shows the screen you will get.



```
            TEST OF BASIC INPUT ROUTINES (FILTERS NOT CALIBRATED!!
  1:  Raw input channel 1 (oscilloscope)ILTERS NOT CALIBRATED!!
                                        FILTERS NOT CALIBRATED!!
  3:  I/Q channel 1 (oscilloscope)      FILTERS NOT CALIBRATED!!
                                        FILTERS NOT CALIBRATED!!
  5:  I/Q channel 1 (fft input)         FILTERS NOT CALIBRATED!!
                                        FILTERS NOT CALIBRATED!!
  7:  channel 1 spectrum                FILTERS NOT CALIBRATED!!
                                        FILTERS NOT CALIBRATED!!
X Menu   Q Quit   0-9 Change mode   +- Change scale   P Pause!!
mode=1 pixmode=0 dbhlp=1 scale=-0.000dB(1.000000)  CALIBRATN=4096 inc=1
AD max = 8388607                        FAD min = -8388608   D!!
                                        FILTERS NOT CALIBRATED!!
Math:   8.91%   Graph:  34.10%   Idle:  56.97%   N239   RATED!!
```

***Fig 1.** Linrad in "Hardware test mode".*

What you should look for is AD max and AD min and the numbers that follow. With a 24 bit A/D board you should read 8388607 and -8388608 as in fig.1. With a 16 bit board you should read 32767 and -32768.

In the not unlikely case you get lower readings, saturation does not happen in the A/D converter but somewhere else. It could be your signal source but it could also be the amplifier on the sound board.

If it is the signal source, just get another one, if it is not the signal source, increase the volume switch setting on the sound board by 6dB and loose 1dB instead of 0.25dB of the dynamic range. If saturation still happens in the sound board audio amplifier, try the other inputs if their noise floor is equally good or get a better sound board!

Once you have set the sound board for the maximum gain it can have without loss of dynamic range and made sure the D/A can be saturated, measure the signal level required at the A/D board input for the boart to just saturate. For the Delta44 board one needs about 11V p-p or 3.9V RMS.

The ideal analog hardware

The analog hardware has to be able to supply the required voltage with very low distortion.

With ideal analog hardware, the noise floor does not raise at all when it is connected to the audio board if the antenna and first amplifier is disconnected

With ideal hardware the only contributions to the noise floor one can see on the screen should come from the antenna, the first amplifier and the audio board. Ideally the noise floor should increase by 20dB when the first amplifier and the antenna is connected. Then 99% of the noise comes from antenna/preamp and 1% from the A/D board and system noise figure is degraded by only 0.05dB due to the sound board.

In difficult situations it could be enough to allow the noise floor to raise by only 10dB when antenna/preamp is connected. Then 90% of the noise is from the front end and 10% from the A/D converter. The loss of system noise figure is 0.4dB due to the sound board so a 10 dB increase of dynamic range can be traded for a S/N loss of 0.35dB.

In really difficult cases the gain can be reduced by 20dB from ideal. Then 50% of the noise will come from the sound board and 50% from the antenna/preamp. The associated loss of NF will be 3dB but that is more or less ok in this situation. With signal levels high enough to require 20dB below optimum gain, the sideband noise from the troublesome signal will degrade sensitivity anyway - and probably by more than 3dB.

Using non-ideal analog hardware

If the analog hardware is a "normal" receiver, disable AGC or turn down the RF volume to make sure AGC does not act under normal operating. Set the volume controls for the noise floor to be placed 6dB above the sound board noise floor when antenna and the first amplifier is disconnected.

Random noise adds by power. If the sound board noise level is P, placing the noise floor with preamp and antenna disconnected 6dB higher means that the power level then is 4 * P which means that 25% of the noise comes from the sound board and 75% from the analog hardware. In this case, the analog hardware to which the preamp will be connected has 33% more noise compared to its normal use directly off the loudspeaker due to the contribution from the sound board. The output of the preamplifier will go to a receiver with a noise temperature that is degraded by 33% or 1.25dB. In case the noise floor does not rise by at least 16dB when antenna and preamp is connected you need more preamp gain (or lower losses) or a better second amplifier. In case the noise floor rises by more than 20dB your system is good and you should reduce the gain until the preamp does not add more than 20dB.

Check the dynamic range

Once the noise floors are correctly placed, check the performance of your system by listening to a strong signal from the antenna input. Tune the analog hardware to place the signal at the low side of the spectrum. It is essential that the frequency is low enough so three times the audio frequency is within the audio passband. Now, increase the signal gradually and watch the second harmonic. Very probably your analog hardware is not even nearly as good as the sound board so you may see the second and/or third harmonic grow above a reasonable level (-40dB below the fundamental) long before the system is saturated.

Saturation may well be in the analog hardware and not in the soundboard which you should check with the "Hardware test mode" as described above.

For more details and some examples, check: Checking the dynamic range

In case you are using a normal SSB radio, the bandwidth is limited to about 2.5 kHz so a very limited dynamic range is no problem. Strong signals have to be kept outside the 2.5 kHz passband and the analog filters of the radio is responsible for rejecting them. In case the strongest signal the analog hardware can handle is far below sound board saturation, the sound board is not used to its full capacity but that is no problem at all. To process weak signals in noise 8 bit A/D converters would be perfectly ok! More bits are needed in order to process strong signals correctly.

If you are using a wideband analog system, regardless whether you have wide filters (real signals) or a complex baseband signal I and Q (direct conversion radio) the problem is the same. *It is difficult to to make analog hardware that takes full advantage of the dynamic range of a good sound board*. On the other hand *It is very easy to build the analog hardware for a wideband radio with linrad if dynamic range requirements are modest.*

Assuming that -40dB is the maximum acceptable level of the strongest spur, finding what maximum level a signal may have before this limit is exceeded will tell if the hardware is acceptable. If the maximum level as defined this way is many dB below saturation of the sound board, better analog hardware is required.

With really good hardware it is possible to get extremely good performance, the dynamic range of a modified Delta44 can be preserved, which allows quite good performance, comparable to what conventional receivers/transceivers can give.

=====

The Linrad program itself is just another linear receiver (with some additions to it). For processing speed the dynamic range of the digital signal is limited by the use of 16-bit arithmetics and it is essential to set digital signal levels correctly in case the second fft is enabled. The digital signal levels depend on fft sizes, total bandwidth and the nature of the signals you receive.

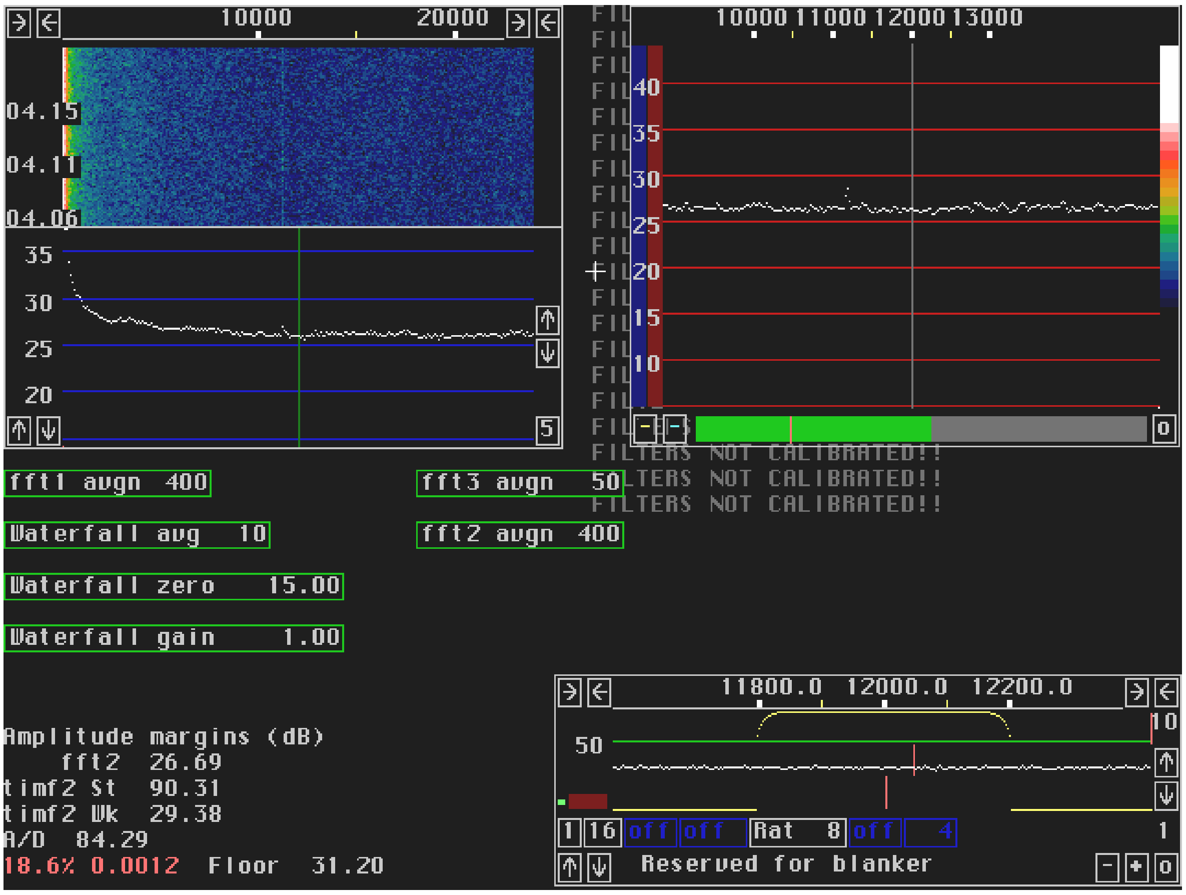
=====

Setting digital signal levels correctly

The zero point of the main fft display

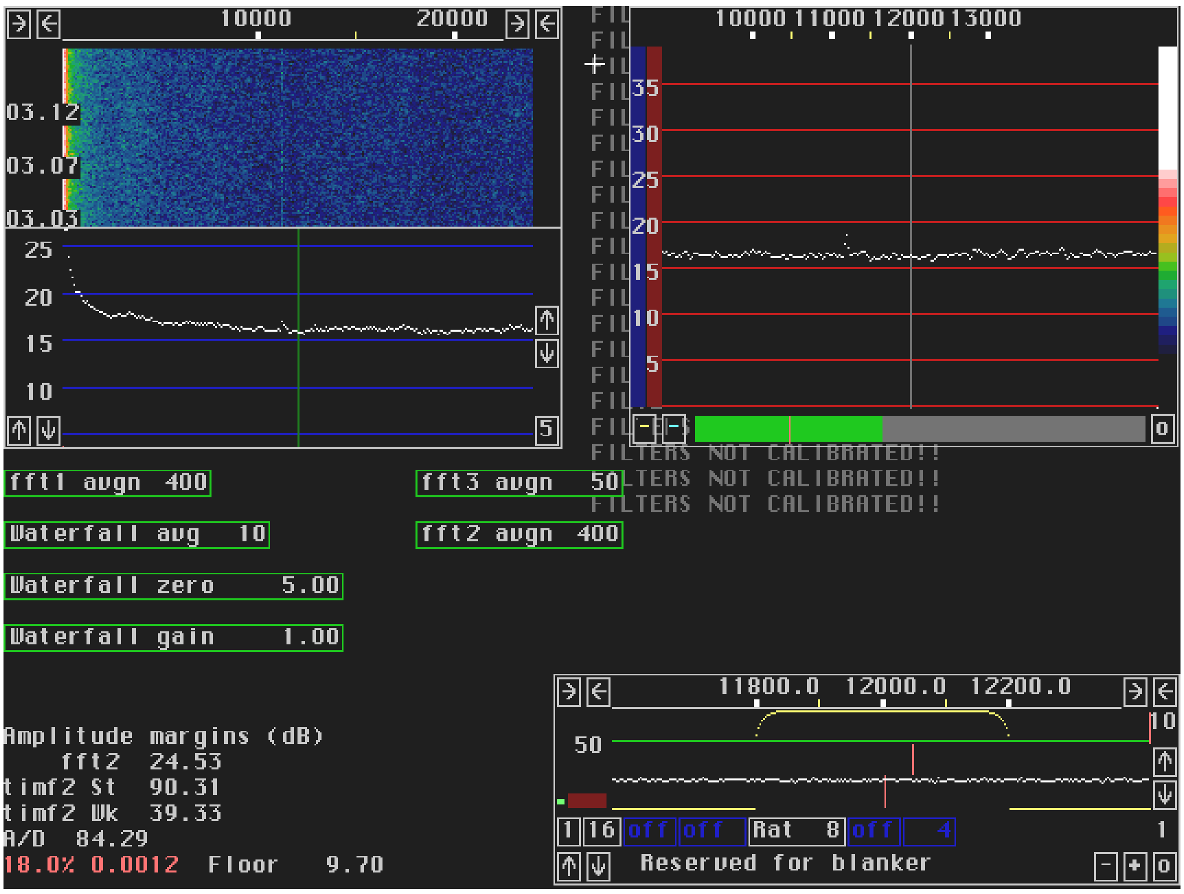
Since the first fft is calculated with floating point numbers it does not matter where the zero point is placed for the transform itself or its display. If the second fft is disabled you can shift the spectrum up and down the dB scale by use of the parameter **First FFT amplitude** which is 1000 by default. You will have to adjust the output audio volume accordingly but processing is not affected.
When the second fft is enabled, the output of the first fft is rounded to integer values and it is essential that the white noise floor is placed high enough above the quantisation noise caused by loosing the floating point decimals.
For the noise blanker to operate it is essential that the noise floor is placed low enough. More about that below.
Compare the figures 1 to 5. These figures are produced from exactly the same input from the A/D converter (stored on the hard disk). A weak tone at 11.2kHz is injected into a Delta44 sampling at 96kHz. The size is 2048 for both fft1 and fft2 with sine to power 4 windows for both. The x-axis is one pixel per bin for both fft1 and fft2. Note that the second fft gain parameters have to be adjusted to keep the processing within 16 bit to avoid introduction of other rounding errors (more about them further down this page).
To set **First FFT amplitude**, first press A to get the amplitude information in the lower left corner. The bottom line gives the noise floor in bits RMS. If you have a nice location with a noise floor that is essentially white noise of constant amplitude, you should find the noise floor somewhere between 10 and 30 dB when using the default parameter value 1000. The Floor RMS value should be in the range 3 to 30. The soundboard itself should be somewhere

between -5 and +5dB and your analog hardware should add between 15 and 25 dB. In most situations there is no reason to do anything, just use the default value of 1000. In case your hardware is not properly adjusted you should change the First FFT amplitude parameter to place your white noise background somewhere in the 10 to 30 dB range.



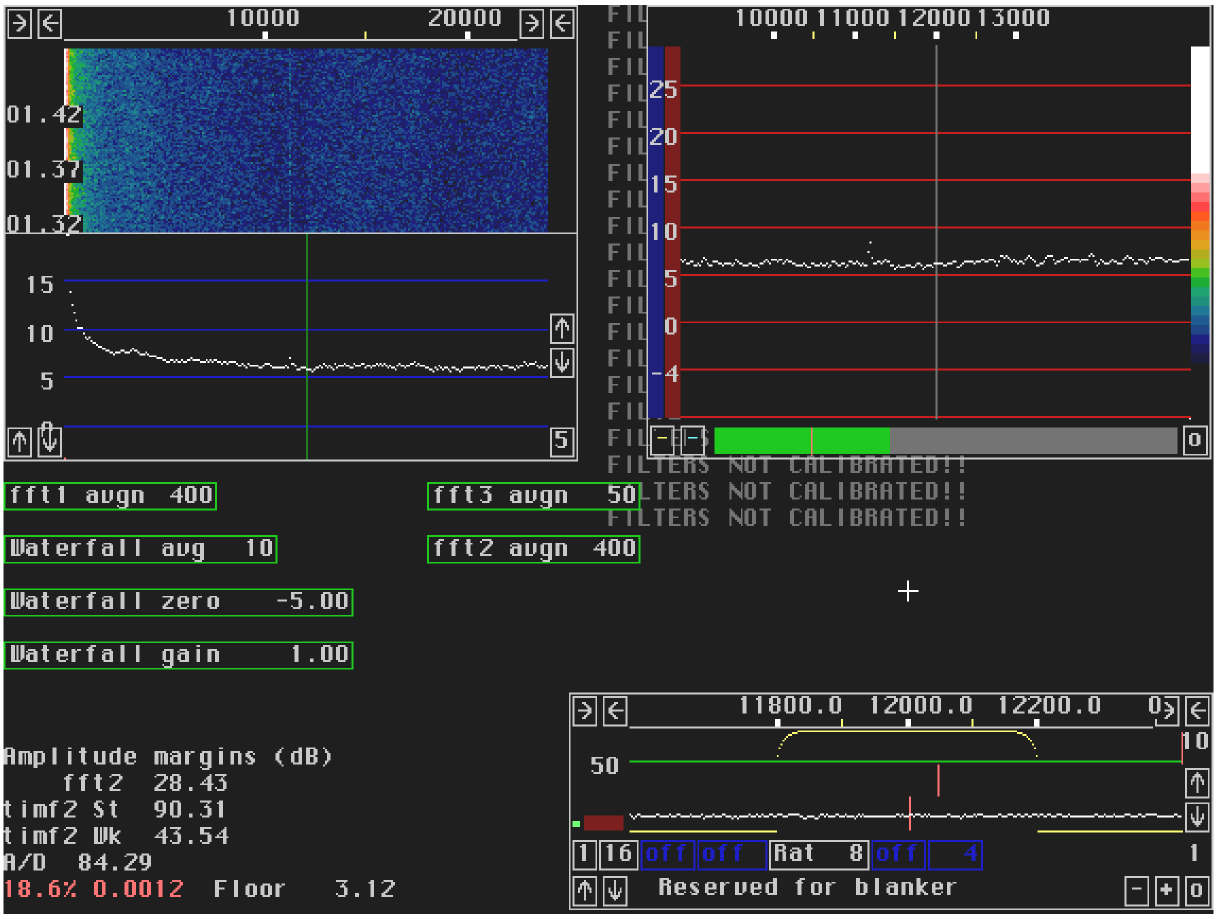
```
fft1 avgn  400                 fft3 avgn    50
Waterfall avg    10            fft2 avgn  400
Waterfall zero     15.00
Waterfall gain      1.00

FILTERS NOT CALIBRATED!!
FILTERS NOT CALIBRATED!!
FILTERS NOT CALIBRATED!!

Amplitude margins (dB)
    fft2   26.69
timf2 St   90.31
timf2 Wk   29.38
A/D   84.29
18.6% 0.0012  Floor   31.20

1 16 off off Rat   8 off    4
Reserved for blanker
```

Fig 1. *First FFT amplitude = 31600. The noise floor is at 31 RMS in units of one bit. Loosing decimals deep down in the noise is insignificant.*



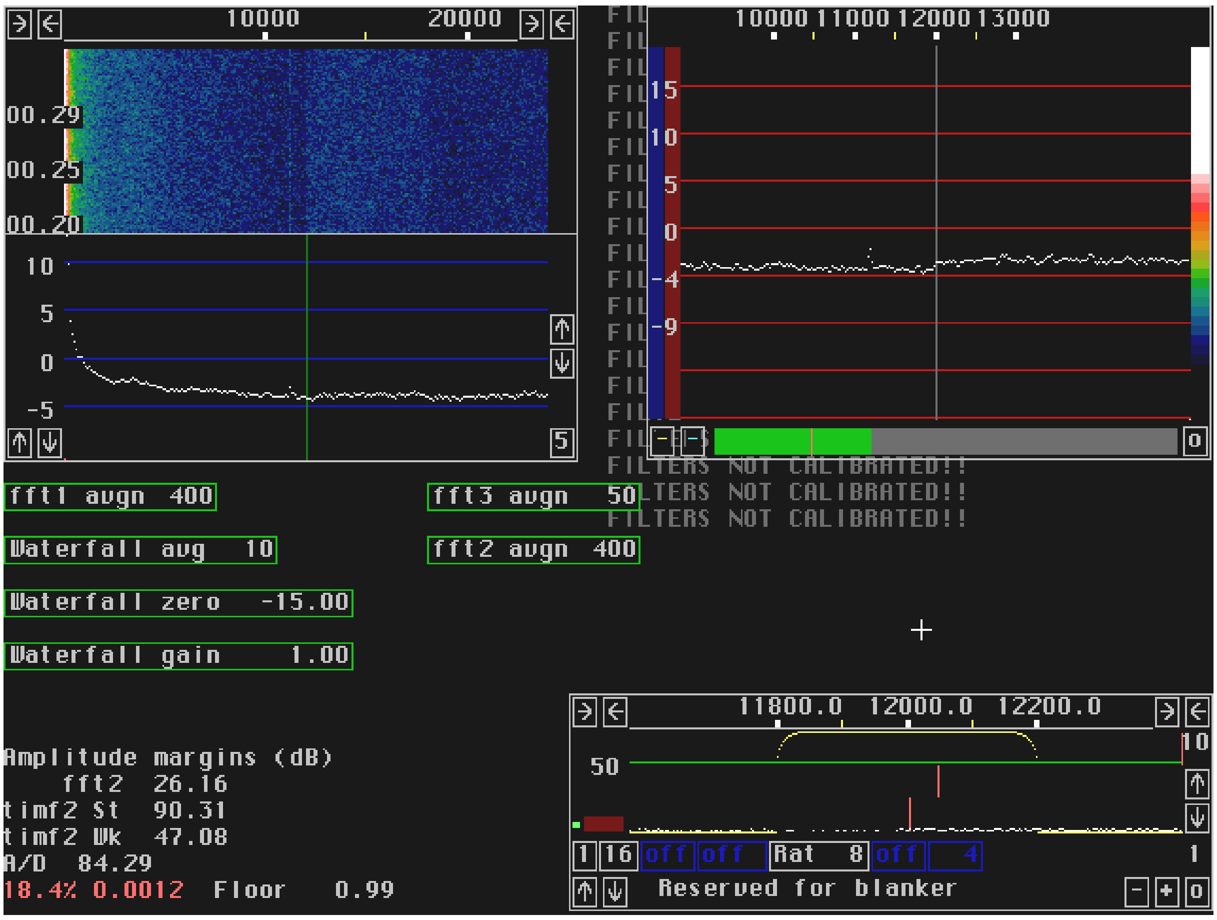
```
fft1 avgn  400                fft3 avgn  50
Waterfall avg   10            fft2 avgn  400
Waterfall zero      5.00
Waterfall gain      1.00

Amplitude margins (dB)
     fft2  24.53
timf2 St   90.31
timf2 Wk   39.33
A/D   84.29
18.0% 0.0012   Floor    9.70
```

***Fig 2.** First FFT amplitude = 10000. The noise floor is at 9.7 RMS in units of one bit, which means that quantisation noise is around 28dB below the noise floor. Note that there is no difference in the waterfall graph or in the high resolution graph compared to fig.1.*



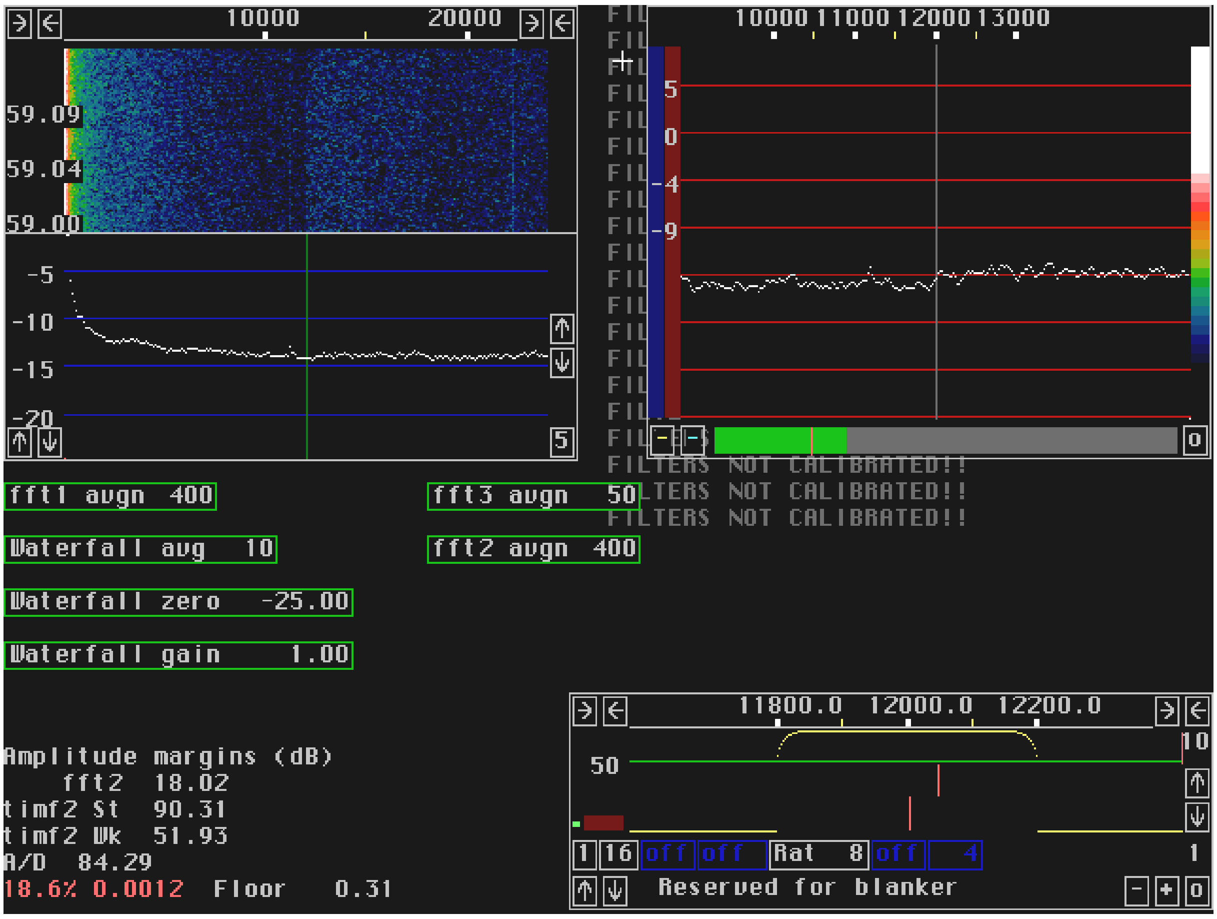
```
fft1 avgn  400                fft3 avgn  50
Waterfall avg   10            fft2 avgn  400
Waterfall zero     -5.00
Waterfall gain      1.00

Amplitude margins (dB)
     fft2  28.43
timf2 St   90.31
timf2 Wk   43.54
A/D   84.29
18.6% 0.0012   Floor    3.12
```

Fig 3. *First FFT amplitude = 3160. The noise floor is at 3.1 RMS in units of one bit, which means that quantisation noise is around 18dB below the noise floor.*



```
FILTERS NOT CALIBRATED!!
FILTERS NOT CALIBRATED!!
FILTERS NOT CALIBRATED!!

fft1 avgn  400          fft3 avgn   50
Waterfall avg   10      fft2 avgn  400
Waterfall zero    -15.00
Waterfall gain      1.00

Amplitude margins (dB)
    fft2  26.16
timf2 St  90.31
timf2 Wk  47.08
A/D   84.29
18.4% 0.0012   Floor    0.99

1 16 off off Rat  8 off  4      1
Reserved for blanker
```

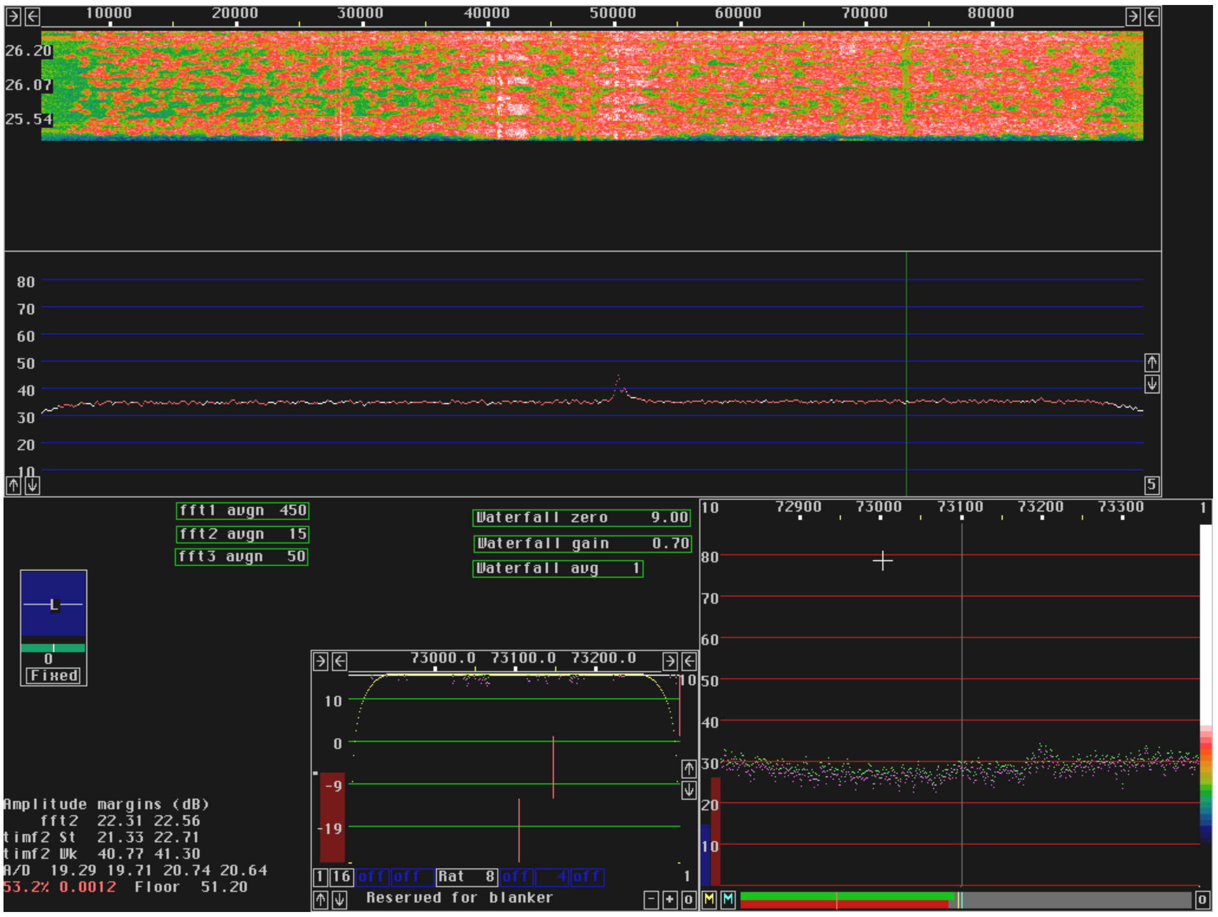
Fig 4. *First FFT amplitude = 1000. The noise floor is at 0.99 RMS in units of one bit which means that quantisation noise is around 8dB below the noise floor. There are subtle differences compared to the previous screens. Easiest to see is the noise floor jump at 12kHz. The quantisation is added to a fourier transform before back transformation. A signal that is a few dB above the noise floor will suffer from a correspondingly smaller degradation due to rounding errors.*



```
fft1 avgn  400            fft3 avgn   50
Waterfall avg   10        fft2 avgn  400
Waterfall zero   -25.00
Waterfall gain     1.00

Amplitude margins (dB)
    fft2   18.02
timf2 St   90.31
timf2 Wk   51.93
A/D   84.29
18.6% 0.0012   Floor    0.31
```

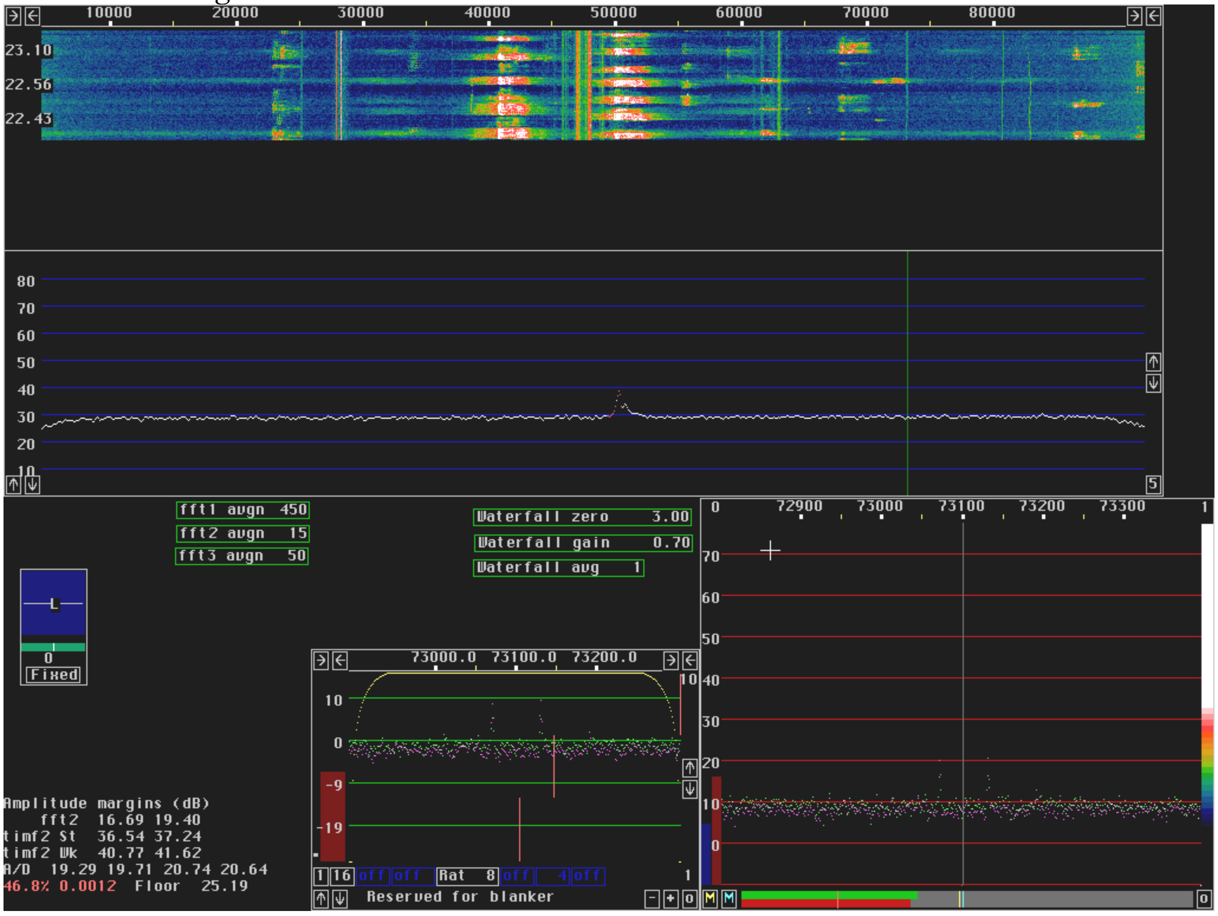
***Fig 5.** First FFT amplitude = 316. The noise floor is at 0.31 RMS in units of one bit, similar to the quantisation noise. The S/N of the signal at 11.2kHz is clearly degraded. There is no reason to place the noise floor low like this.*

The most critical situation is if you want to run a small first fft and a very large second fft in a location with extremely bad powerline noise.
To show what the problem is, the data used to demonstrate the noise blankers was used with extreme fft size parameters. The first fft size is set to 1024 and the second fft is set 65536. Such large size ratios should not be used normally, there is no reason. The large size ratio makes linrad reserve 6 bit for signal growth in the second fft, one bit for sign and one bit for the noise floor being calculated as the average of I and Q for two rx channels. There is also one bit of safety margin so in this case only 7 bits are allowed for weak signals and noise.
The test signal has powerline noise that reaches 25dB above the white noise floor. Since 25dB is nearly 6 bits there is only one bit left for the white noise floor which then comes very close to the quantisation noise at about 0.3 in units of one bit.
When the noise floor (powerline noise) is placed at 35 dB on the main fft display it is already too strong to be considered by linrad to be noise or a weak signal see fig. 6. Far too many frequencies are considered as strong signals, they are coloured red and the noise blanker does not work. Note that the fft1 gain parameter is 20dB smaller than normal because I added 20dB of extra RF gain to overcome hum at the center of the passband. Since no strong signals are present this loss of 20dB of the dynamic range is completely without any adverse effect.
The remaining figures on this page show the screen with the "First FFT amplitude" parameter stepped downwards in 6dB steps. Other parameters are adjusted to keep the signal levels constant further down the signal path. Only the rounding errors at the first fft output affects the S/N in figures 6 to 10.



```
fft1 avgn  450
fft2 avgn   15
fft3 avgn   50

Waterfall zero    9.00
Waterfall gain    0.70
Waterfall avg     1

Amplitude margins (dB)
    fft2   22.31 22.56
timf2 St   21.33 22.71
timf2 Wk   40.77 41.30
A/D   19.29 19.71 20.74 20.64
53.2% 0.0012  Floor   51.20

Reserved for blanker
```

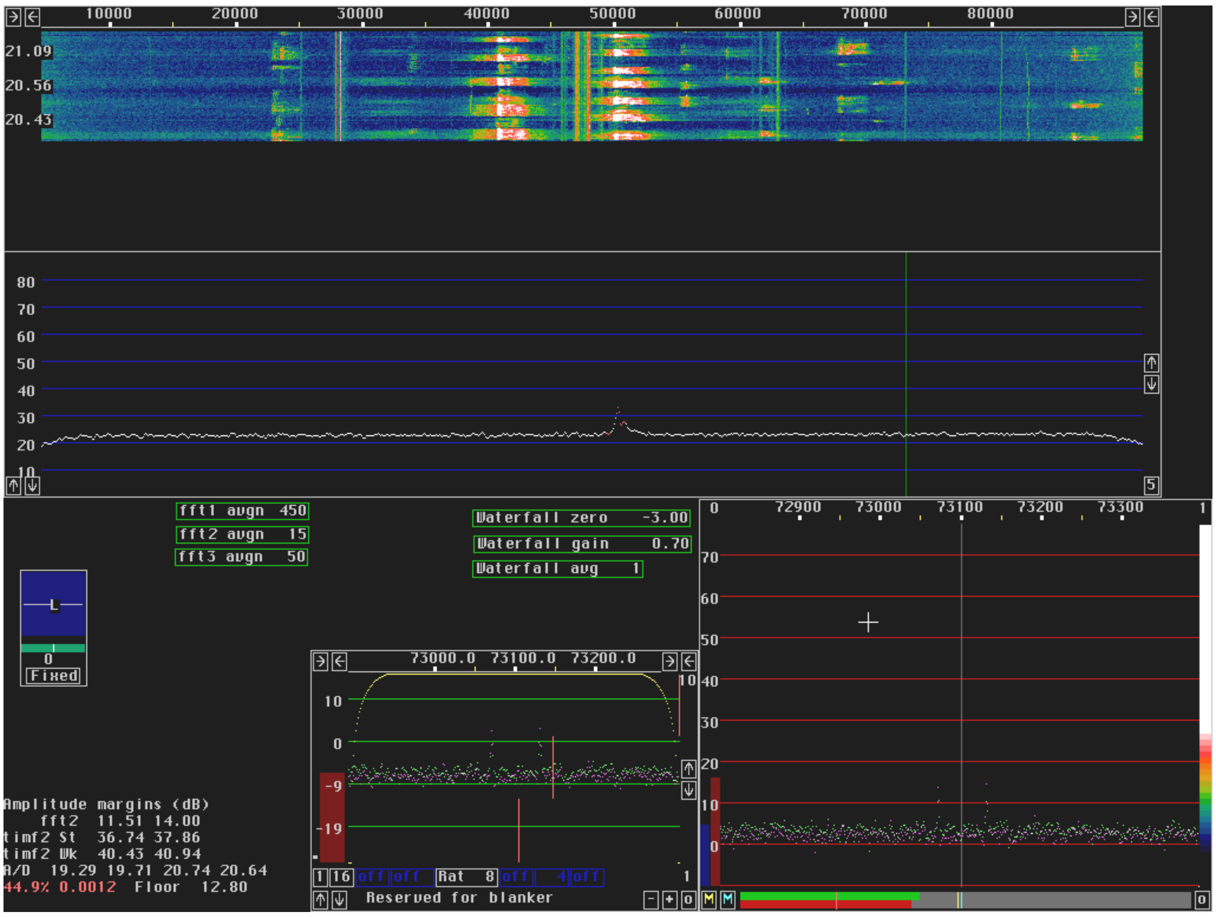
Fig 6. First FFT amplitude = 80. The noise floor is at 51 RMS in units of one bit. This is 35dB above 1 bit or close to 6 bits. With the ridiculously large fft2 to fft1 ratio used the noise level is too strong. The noise blanker does not work.



```
fft1 avgn  450
fft2 avgn   15
fft3 avgn   50

Waterfall zero    3.00
Waterfall gain    0.70
Waterfall avg     1

Amplitude margins (dB)
    fft2   16.69 19.40
timf2 St   36.54 37.24
timf2 Wk   40.77 41.62
A/D   19.29 19.71 20.74 20.64
46.8% 0.0012  Floor   25.19

Reserved for blanker
```

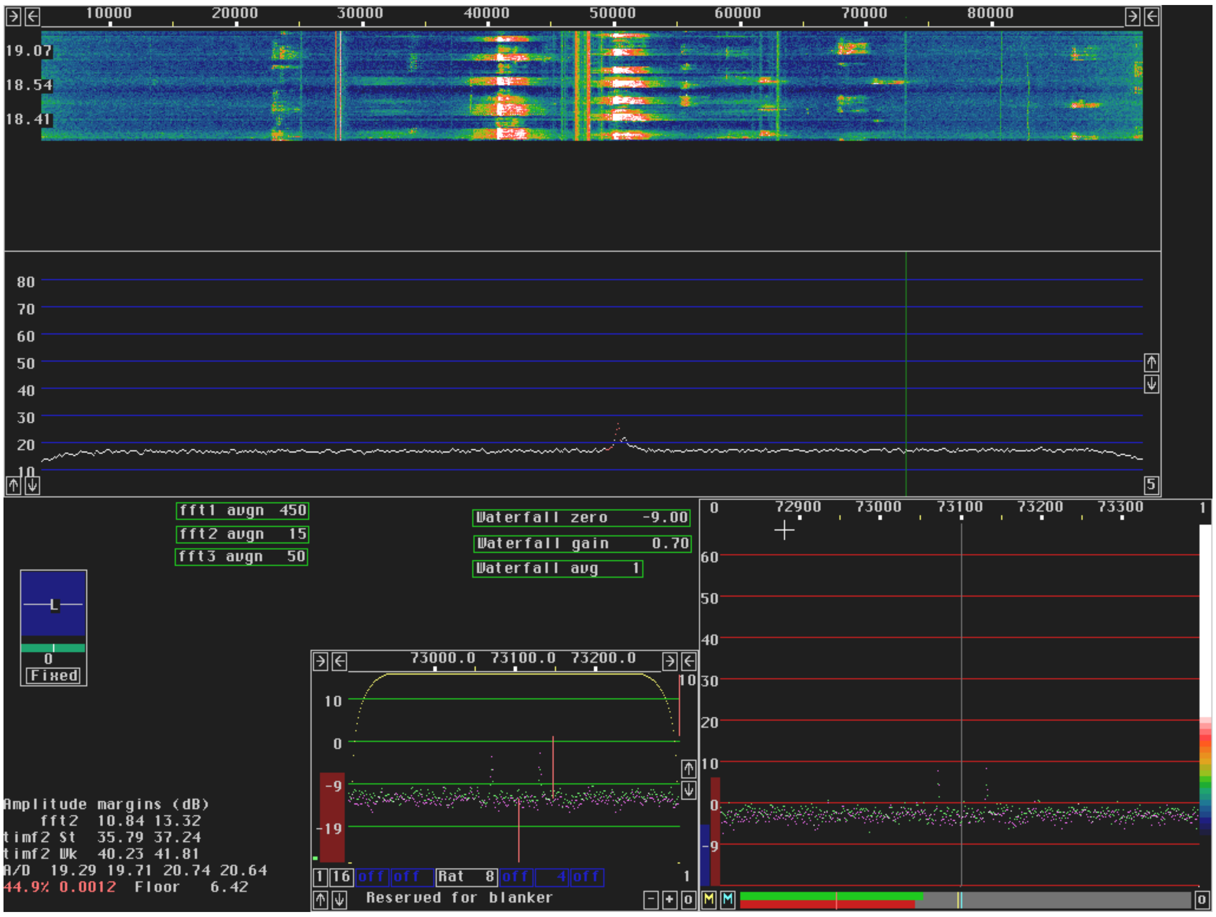
Fig 7. First FFT amplitude = 40. The white noise floor which is close to the noise floor seen in the waterfall graph and in the high resolution graph is at about 3dB in the fft1 spectrum, 25

dB below the powerline noise and 14dB above the quantisation noise. Despite the silly fft2 to fft1 ratio everything works ok.



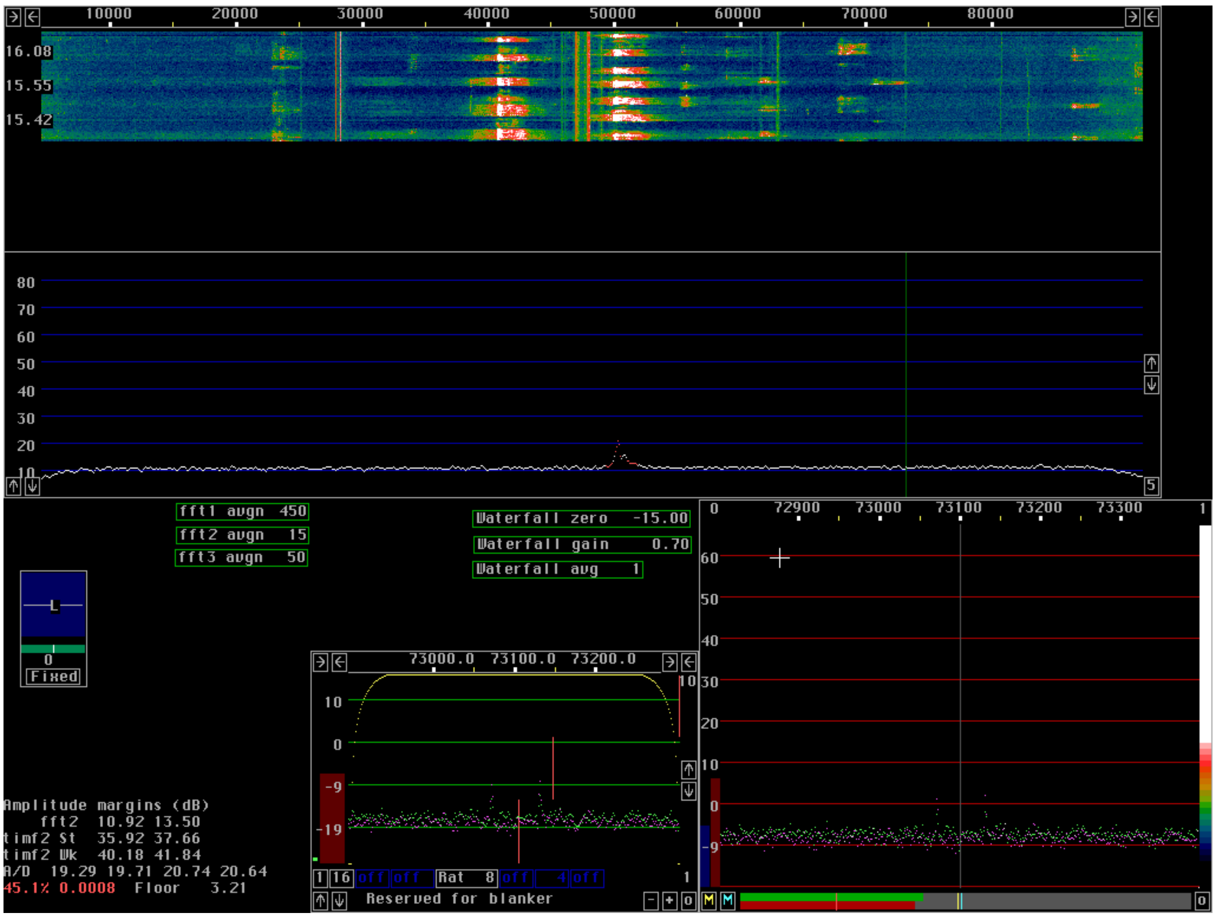
```
fft1 avgn  450
fft2 avgn   15
fft3 avgn   50

Waterfall zero    -3.00
Waterfall gain     0.70
Waterfall avg    1

Amplitude margins (dB)
    fft2  11.51 14.00
timf2 St  36.74 37.86
timf2 Wk  40.43 40.94
A/D  19.29 19.71 20.74 20.64
44.9% 0.0012  Floor  12.80

Reserved for blanker
```

Fig 8. *First FFT amplitude = 20. The white noise floor is at about -3dB in the fft1 spectrum, 25 dB below the powerline noise and 8dB above the quantisation noise. There should be a very small loss of S/N but I can not see it. Everything still works fine.*



```
fft1 avgn  450
fft2 avgn   15
fft3 avgn   50

Waterfall zero    -9.00
Waterfall gain     0.70
Waterfall avg     1

Amplitude margins (dB)
     fft2   10.84 13.32
timf2 St    35.79 37.24
timf2 Wk    40.23 41.81
A/D   19.29 19.71 20.74 20.64
44.9% 0.0012   Floor    6.42

Reserved for blanker
```

Fig 9. *First FFT amplitude = 10. The white noise floor is at about -9dB in the fft1 spectrum, 25 dB below the powerline noise and 4dB below the quantisation noise. Subtle differences can be seen in the noise floor of the high resolution graph.*

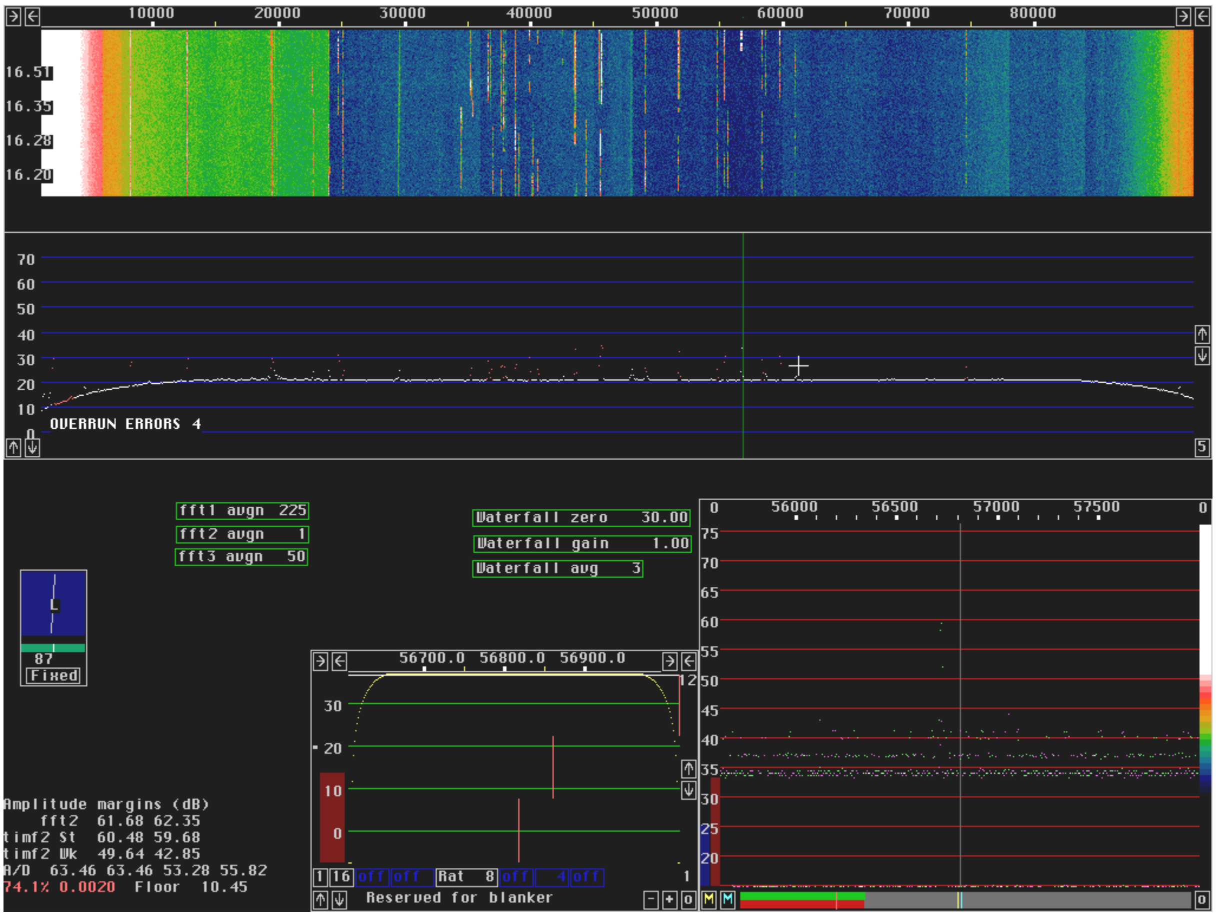


```
fft1 avgn  450
fft2 avgn   15
fft3 avgn   50

Waterfall zero   -15.00
Waterfall gain     0.70
Waterfall avg     1

Amplitude margins (dB)
     fft2   10.92 13.50
timf2 St    35.92 37.66
timf2 Wk    40.18 41.84
A/D   19.29 19.71 20.74 20.64
45.1% 0.0008   Floor    3.21

Reserved for blanker
```

Fig 10. *First FFT amplitude = 5. The white noise floor is at about -15dB in the fft1 spectrum, 25 dB below the powerline noise and 10dB below the quantisation noise. Surprisingly enough, the weak signals in the high resolution graph have lost only 2dB of their S/N.*

The fft1 back transform gain
The output of the first fft is limited to fit with a good margin in 16 bit. That is in the frequency domain. When the back transform is taken, any pulses present may reach large amplitudes. Saturation should be avoided. Press A to see the amplitude margin information. Use the parameter **First backward FFT att N** to set the timf2 St and Timf2 Wk margins. This parameter determines how many loops in the back transform that are run with a shift instruction that divides by two to stop further growth of the time domain signal. If the parameter is too large the margin will become very large and not enough bits will be used for the timf2 signals.
The noise floor that remains with the weak signals is displayed as horisontal bars in the high resolution graph. The red vertical line is at 20dB over a single bit. There is no reason to place the noise floor higher than that although it makes no harm as long as there is a margin shown in the table you get by pressing A.

The second fft gain
The second fft gain is controlled in the same way as the first backwards fft. Set the parameter "**Second forward FFT att. N**" to get a reasonable margin for fft2, the first line in the data obtained by pressing A on the normal screen.
Fig.11 shows what happens if the attenuation is far too low. The fft2 margin is 62 dB. The high resolution graph shows that the power at many frequencies is zero so all information on those frequencies is lost. Loosing bits is a rounding error and it goes different ways above and below 25% of the Nyqvist frequency. If the noise floor is at 1.5bit, rounding up or down will make a difference of 6dB!!!
The maximum amplitude that can occur in the second fft is limited by the selective limiter. There is never any reason to have a big margin here. 30dB is normally enough to get the noise floor with many enough bits but there is nothing wrong with a smaller margin as long as it is above zero.



```
fft1 avgn  225
fft2 avgn    1
fft3 avgn   50

Waterfall zero    30.00
Waterfall gain     1.00
Waterfall avg   3

Rat   8

Amplitude margins (dB)
    fft2   61.68 62.35
timf2 St   60.48 59.68
timf2 Wk   49.64 42.85
A/D   63.46 63.46 53.28 55.82
74.1% 0.0020   Floor   10.45
```

***Fig 11.** "Second forward FFT att. N" set far too large. The fft2 margin is 62dB. Should be in the range 5 to 25dB*

=========

Checking the dynamic range of your hardware
In case Linrad is used to receive large bandwidths it is essential that the analog hardware has extremely good linearity.

If one strong signal enters the computer at 5 kHz while the desired signal is at 10 kHz any non-linearity in the hardware that produces second order harmonics will be disastrous.

Good linearity is also required for the intelligent noise blanker that assumes noise pulses to be shaped by the filters independently of amplitude. Poor linearity also leads to incorrect calibration.

Checking the dynamic range is very easy. You need a single strong signal only. Errors show up as extra signals.

Calibration: Linrad has routines that compensate for "errors" in the analog hardware.
Linrad is working happily in an uncalibrated state but you will not be able to use the intelligent noise blanker.
Calibrating amplitude and phase of the entire filter chain used by the system provides a very flat background on top of which very small signals can be seen. The spectrum becomes extremely flat even if the analog hardware uses filters with a very irregular frequency response. Filter characteristics of the analog hardware only affect the dynamic range on a calibrated system.

Calibration of filter responses in Linrad
Linrad needs to know the frequency response of the hardware it is connected to in order to produce optimum processing. There are two ways of obtaining the frequency response of a filter. The obvious one is to send a sine-wave through the filter and measure the attenuation at different frequencies. That way the amplitude vs frequency curve can easily be obtained. The phase vs frequency curve is obtained by measurement of the phase shift from input to output at different frequencies.
When the "filter" is a radio receiver run in linear mode (CW or SSB without AGC) the amplitude measurement by stepping a signal generator is straightforward and often used, but the phase measurement is difficult because one has to extract all local oscillators from the receiver and mix the test signal with them to get a signal at the loudspeaker output frequency to use as the reference phase. The reference signal must have a known phase shift over the frequency range of interest, typically it is constant and can be neglected since receivers generally have a narrow bandwidth in linear mode.

The measurement in the frequency domain with a signal generator allows a very large dynamic range because the attenuator on the signal generator can be adjusted for the loudspeaker output to not saturate, but the accuracy of each data point is limited. Reading amplitude levels to better than a few percent is difficult and when using an attenuator the uncertainty is typically 0.5 dB (6% amplitude.)

The other, perhaps less obvious way of measuring a filter is to do it in the time domain by sending a wideband pulse through the filter and then analyze the pulse response. This is done by use of a pulse generator.
Pulses have wide spectra that can be assumed to be totally flat within a narrow frequency range. The spectrum of an infinitely short pulse is totally flat and infinitely wide, but it contains an infinitely small amount of energy because the amplitude must be within the range for which the test object has a linear (non-saturated) response. A limited amplitude during an infinitely short time means an infinitely small power.

A low frequency square wave with 50% duty has a spectrum containing odd overtones that falls off by 6 dB each time the frequency is doubled up to a limit that is set by the rise and fall times. A rectangular pulse, a square wave with a duty cycle far from 50% has more energy in each pulse as compared to a square wave in the mid frequency range but the spectrum has an oscillatory behaviour at high frequencies.

Figure 1 shows the spectrum of a 1 microsecond pulse with a repeat frequency of 20 kHz. There is a minimum at 1 MHz where the pulse duration matches one period of the frequency. This overtone spectrum contains odd and even overtones at nearly the same amplitude. When the duty is near 0% odd and even overtones have the same amplitude. Figure 2 shows the spectrum of a 10 kHz square wave. This spectrum contains odd harmonics only so the separation between the (strong) overtones is the same as in figure 1 reflecting the fact that the pulse repetition frequency as seen through a band pass filter at RF frequencies is the same, 20 kHz, in both cases. When the signal is routed through a filter that is wide enough for the pulses to be well separated in the time domain the spectrum of each individual pulse is the same in both cases as long as the bandwidth is small enough for all overtones within the passband to have equal amplitudes in figures 1 or 2.

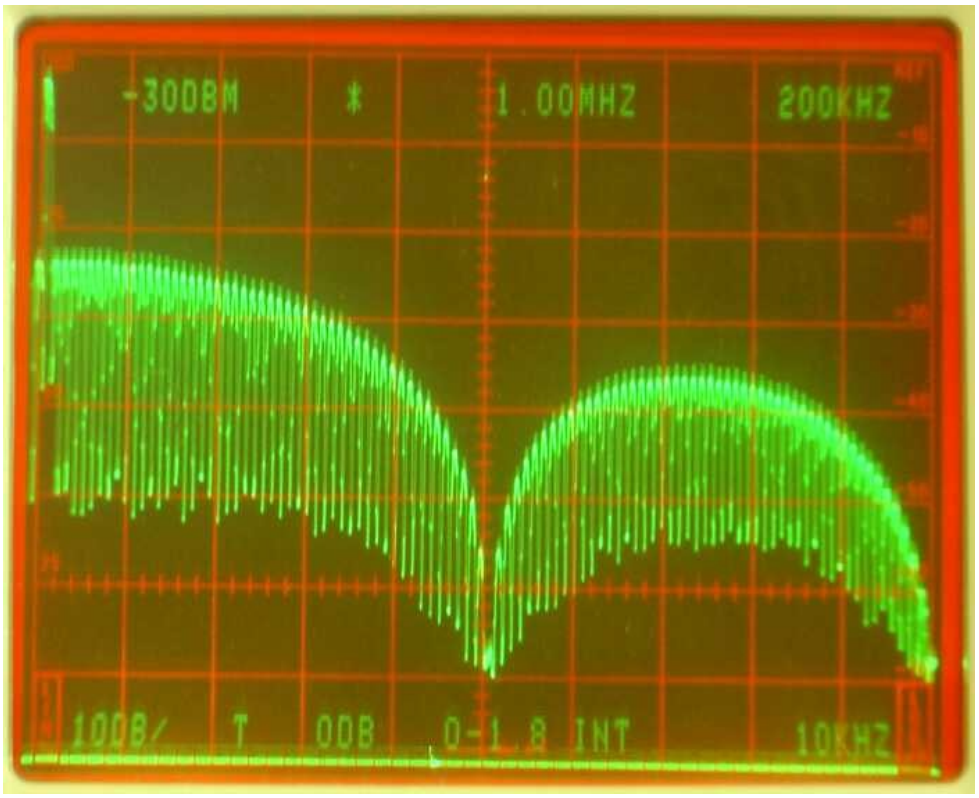


Fig 1. *The spectrum of a 1 microsecond pulse having a repeat rate of 20 kHz.*

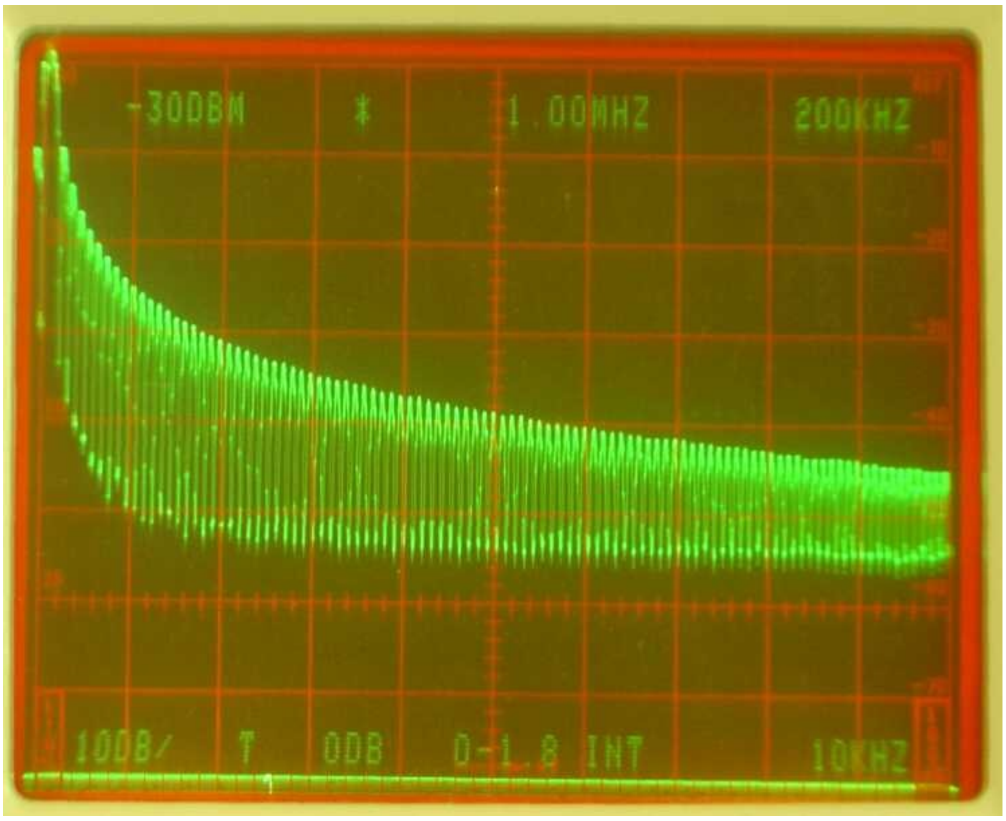


Fig 2. *The spectrum of a 10 kHz square wave.*

Figure 1 and 2 are obtained with a function generator (Escort EGC-2230) set for an amplitude of 27 mV into the spectrum analyzer so the power levels in the two figures can be compared directly. What waveform to usedepends on the frequency and bandwidth of the filter one wants to investigate. When a squarewave is used, the bandwidth must be much smaller than the frequency of operation since the pulse amplitude always falls like 1/F which means that F has to be nearly the same at both ends of the passband. The 1 us pulse has a flat spectrum up to 100 kHz and can be used for a bandwidth of about 5 kHz at 500 kHz.

A pulse length of 20 ns gives a flat spectrum up to about 5MHz with a fall-off that is slow enough to make it suitable for calibration at a bandwidth of up to 1 MHz up to 25 MHz. Shorter pulses can be used at higher frequencies, but the energy of each pulse becomes smaller. A pulse length of 5 ns is suitable at 144 MHz, but it may be difficult to get a good S/N for such a pulse because being 5 times shorter and spread out over a 5 times wider frequency range the power density (dBm/Hz) is 25 times smaller than for the 20 ns pulse. For calibration at 144 MHz one should use an LC bandpass filter with a flat bandwidth of about 5 times more than the desired bandwidth. This way the pulse is smeared out in time to a dampes oscillation with a smaller amplitude allowing a higher level of the original 5 ns pulse. For more details look here A pulse generator for receiver calibration

At the optimum frequency for a short pulse, about 500 kHz in figure 1, the signal level of the pulses produced by a square wave is about 6 dB weaker. Both the rising and the falling edges add their energy into the pulse at this frequency. If the square wave has low noise this does not matter and Linrad can be calibrated by use of square waves as well as by use of short pulses.

Requirements on a pulse generator for Linrad calibration.

1. The spectrum of the pulses has to be flat over the frequency range of interest. Linrad provides no help - this is the operators responsibility.

2. The pulse repetition frequency must be low enough for the pulses to be completely resolved in time.
A filter converts a short pulse to a damped oscillation that oscillates for a time related to the Q of the filterwhich in turn is related to the skirt steepness. The first screen of the calibration sequence helps for this.

3. The pulse repetition frequency must be low enough for Linrad to evaluate the lowest frequency of interest. The pulse repetition frequency should not be higher than half the lowest frequency of interest. When a normal radio is used half the frequency of the -10 dB point could be suitable. Setting a high pitch is a good idea since this requirement then becomes fullfilled by the previous requirement while the overtones of the mains frequency come outside the passband.

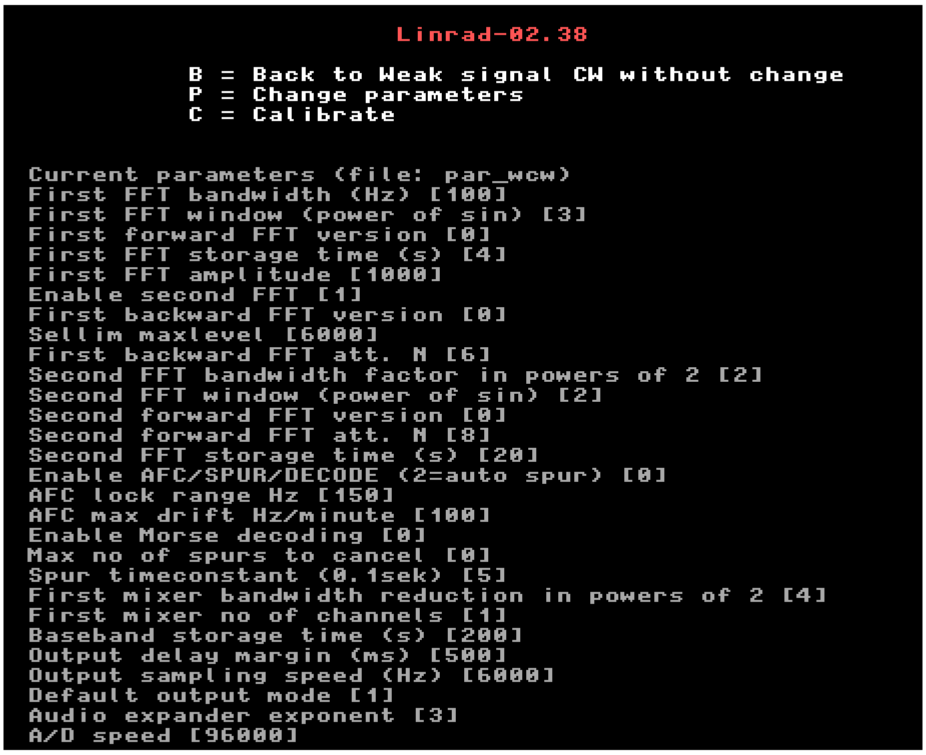
When direct conversion or digital hardware is used to provide I and Q, the lowest frequency of interest depends on how wide center notches one can interpolate across with good enough accuracy.

Linrad provides no help - this is the operators responsibility - but generally a low repetition frequency is better as long as S/N does not become too low or the time to calibrate becomes too long.

4. The signal to noise ratio must be good enough. A square wave that uses a digital IC to switch between VCC and GND should have very low noise, but only if VCC is well decoupled and the GND voltage is properly referenced to the output ground. Spurs and the mains frequency and its overtones must have low levels compared to the pulse amplitude to allow correct calibration.

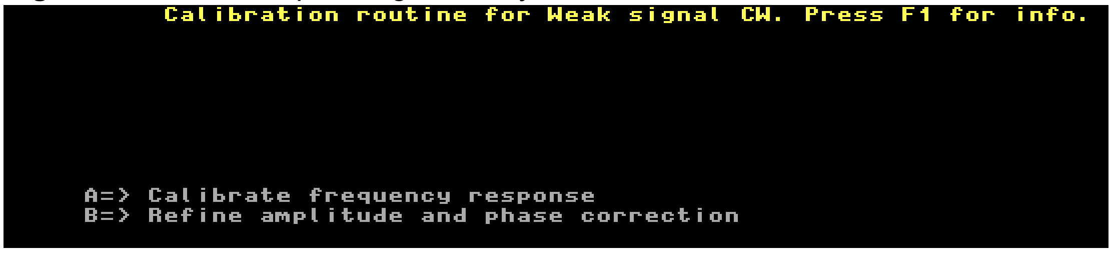
The calibration procedure.

It is advantageous to use the lowest possible frequency for calibration. When converters are used to bring the antenna signal to lower frequencies, For example when using a 144 to 28 MHz converter followed by a direct conversion receiver at 28 MHz it is better to inject the pulses at 28 MHz assuming that the converter has a flat frequency response over the frequency response of interest. Likewise it is better to calibrate a WSE converter chain at 10.7 MHz rather than at 144 MHz. Using the 2.5 MHz input is not a good idea since the frequency response of the 2.5 MHz filters inside the RX2500 will depend slightly of the input impedance seen by the RX2500 so it is better to allow it to see the actual output impedance of the RX10700 to which it will be connected in actual usage.
To reach the calibration procedure, press **X** while Linrad is running in one of its receive modes. The screen showed in fig.3 will then appear. Choose **C** to get into the calibration menu which will look like figures 4 or 5 depending on whether a direct conversion hardware is used or not.



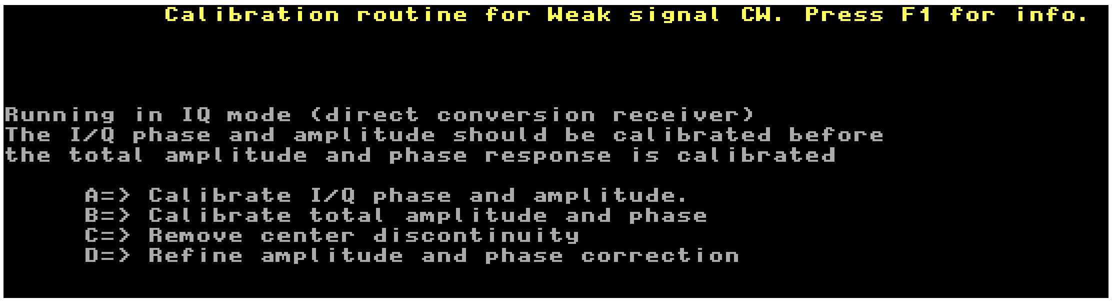
```
                Linrad-02.38

       B = Back to Weak signal CW without change
       P = Change parameters
       C = Calibrate


Current parameters (file: par_wcw)
First FFT bandwidth (Hz) [100]
First FFT window (power of sin) [3]
First forward FFT version [0]
First FFT storage time (s) [4]
First FFT amplitude [1000]
Enable second FFT [1]
First backward FFT version [0]
Sellim maxlevel [6000]
First backward FFT att. N [6]
Second FFT bandwidth factor in powers of 2 [2]
Second FFT window (power of sin) [2]
Second forward FFT version [0]
Second forward FFT att. N [8]
Second FFT storage time (s) [20]
Enable AFC/SPUR/DECODE (2=auto spur) [0]
AFC lock range Hz [150]
AFC max drift Hz/minute [100]
Enable Morse decoding [0]
Max no of spurs to cancel [0]
Spur timeconstant (0.1sek) [5]
First mixer bandwidth reduction in powers of 2 [4]
First mixer no of channels [1]
Baseband storage time (s) [200]
Output delay margin (ms) [500]
Output sampling speed (Hz) [6000]
Default output mode [1]
Audio expander exponent [3]
A/D speed [96000]
```

Fig 3. *The menu after pressing 'X' in any receive mode.*



```
          Calibration routine for Weak signal CW. Press F1 for info.






     A=> Calibrate frequency response
     B=> Refine amplitude and phase correction
```

Fig 4. *The menu after pressing 'C' on the menu of figure 3 when a real-valued signal is produced by the hardware or if digital hardware that automatically has perfect balance between I and Q is in use.*



```
          Calibration routine for Weak signal CW. Press F1 for info.



Running in IQ mode (direct conversion receiver)
The I/Q phase and amplitude should be calibrated before
the total amplitude and phase response is calibrated

     A=> Calibrate I/Q phase and amplitude.
     B=> Calibrate total amplitude and phase
     C=> Remove center discontinuity
     D=> Refine amplitude and phase correction
```

Fig 5. *The menu after pressing 'C' on the menu of figure 3 when direct conversion hardware is in use.*

In IQ mode, the phase and amplitude balance between I and Q should be have been done already, as described here: Calibrate for phase and amplitude errors between the I and Q signals in a direct conversion receiver.

To enter the filter calibration, press the key for **Calibrate total amplitude and phase** The links below give several examples of the calibration on different hardwares with files containing the actual calibration pulses as wav files or as files in Linrads own format. These files will allow you to do the calibration without any hardware at all and to verify that you understand what to do and why.

Two channels with real-valued signals shows calibration with a 50 Hz square wave on a two channel receiver with crystal filters. The focus is *Calibrate total amplitude and phase*. **Check this link before you look at the ones below in case you are not already familiar with Linrad.**

One channel with direct conversion hardware shows calibration on 3.6 MHz with the WSE converters using square waves. The focus is *Remove center discontinuity*.

Calibration with poor S/N. Here an antenna was connected to simulate leak-through of RF signals. The focus is *Refine amplitude and phase correction*

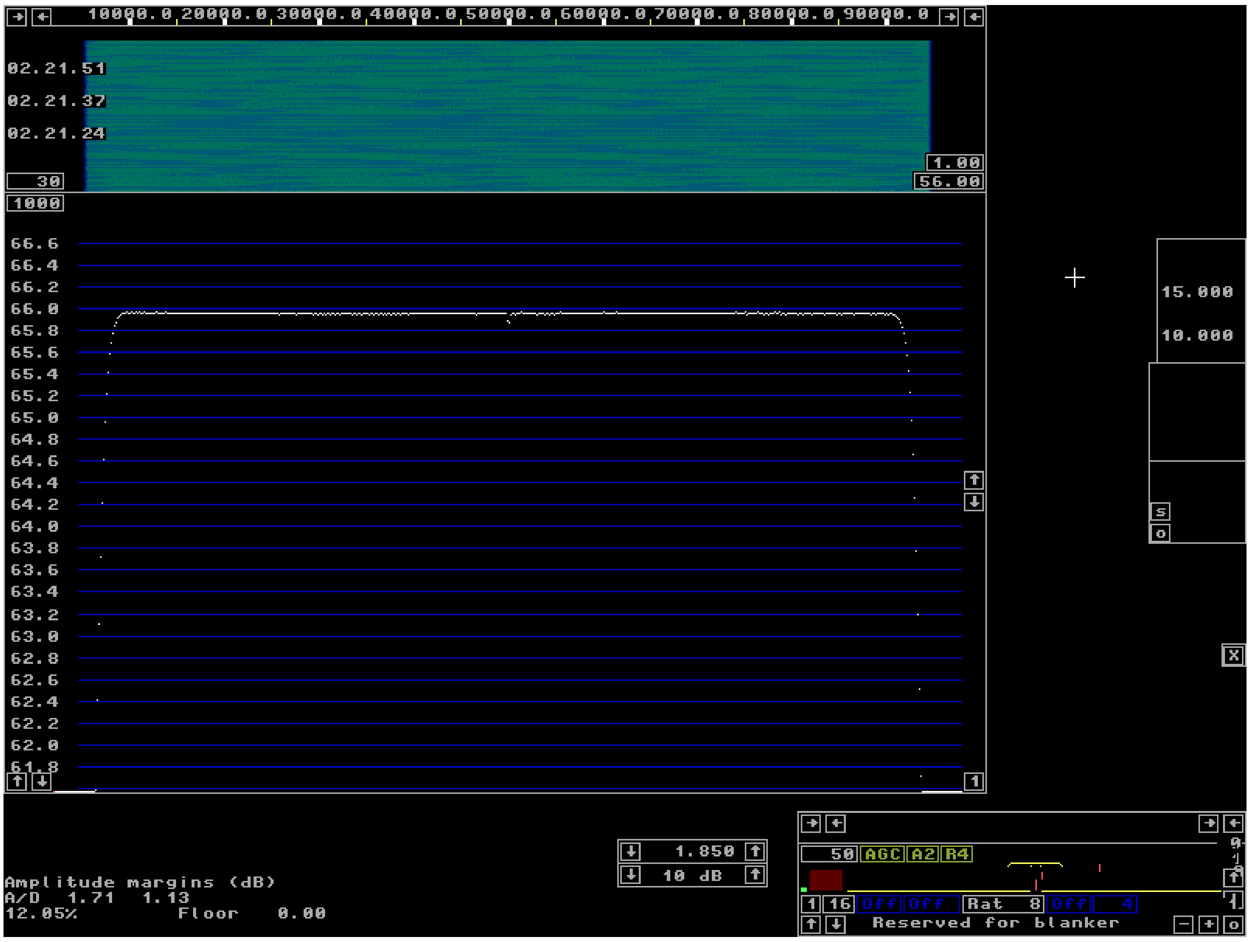
Hardware linearity.

It is normal for a wideband pulse to saturate the wideband amplifiers of the receiver front end. Not only ignition pulses, powerline noise and other truely wideband pulses may saturate a front end. Also radar systems near microwave bands may cause saturation.

Saturation at a wide bandwidth is completely harmless as long as no narrowband signals come close to the saturation level and cause intermodulation. When a pulse saturates, its amplitude becomes lower and its shape becomes different. If the wideband pulse is still much shorter than the pulse response of the bandwidth defining filters even after the pulse amplitude is limited by wideband amplifier saturation, the calibration will still correct. One only has to worry about the recovery time. The wideband amplifiers have to recover very fast after saturation because if they do not, all strong narrowband signals will have a too low amplitude for a while after the pulse. A big capacitor together with a high DC impedance on the gate of a FET can cause a recovery time of milliseconds and that would cause severe keying clicks from all strong signals.

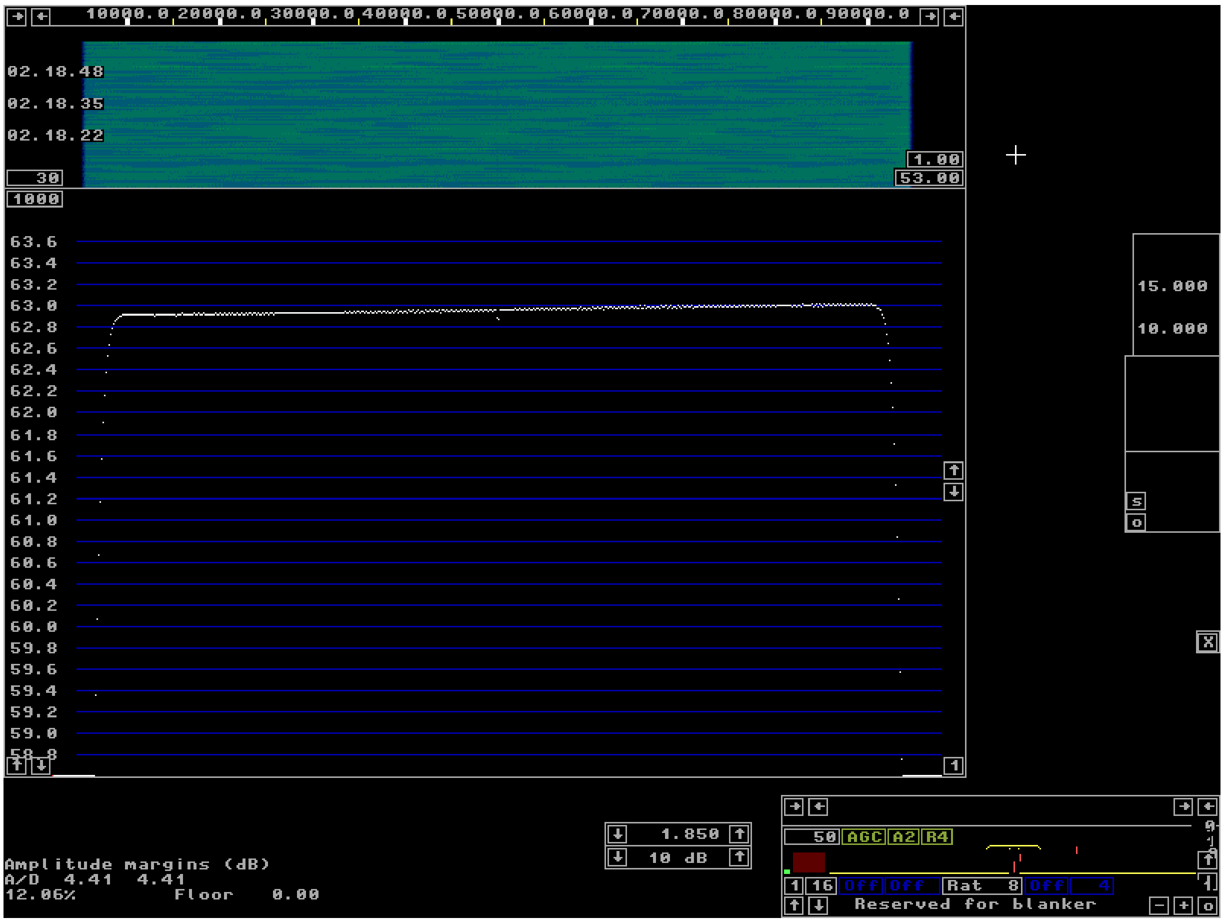
Saturation in amplifiers that operate in a limited bandwidth will change the pulse response that Linrad records and thereby cause an incorrect frequency response. Once you have calibrated, verify the calibration by running Linrad on the calibration signal with averaging on the main spectrum and waterfall graph.
The spectrum you see should be very flat with deviations of only a small fraction of a dB from the ideal response. This response should not change when you insert an attenuator. If it does, your hardware has linearity problems and you should calibrate at a lower pulse level in order to get a flat frequency response for the normal noise floor.

Figures 6 to 9 show averages over several seconds for the WSE converters when calibrated with very strong pulses, only 1 dB below A/D converter saturation. The calibration signal is a 50 Hz square wave that produces 2.7 V across 50 ohms. The calibration is not quite correct for other signals because of the phenomenon illustrated in figure 2, but this small error is independent of the amplitude.
The calibration square wave was fed into the RXHFA unit at 1.85 MHz with maximum gain, +10 dB. The output from the RXHFA is a pulse with an RMS amplitude of 1.9 volts across 50 ohms (+19 dBm) and a duration of about 1.5 microseconds.
Figure 6 shows the pulses at the same signal level as where the calibration was performed. As expected, the spectrum is very flat. With a 3 dB attenuator, figure 7, the spectrum is not quite flat any more. But with 20 and with 40 dB attenuation the spectrum shape is identical as shown in figures 8 and 9. The conclusion is that +19 dBm is a bit too much power output from a RXHFA. Not very surprising because input IP3 is +19 dBm and the gain is 5 dB for an output IP3 of +24 dBm. The calibration pulse is actually 3 dB above the 1 dB compression point of the RXHFA and the reason the calibration error is so small is that the bandwidth of the RXHFA is big enough compared to the visible bandwidth of 93 kHz.



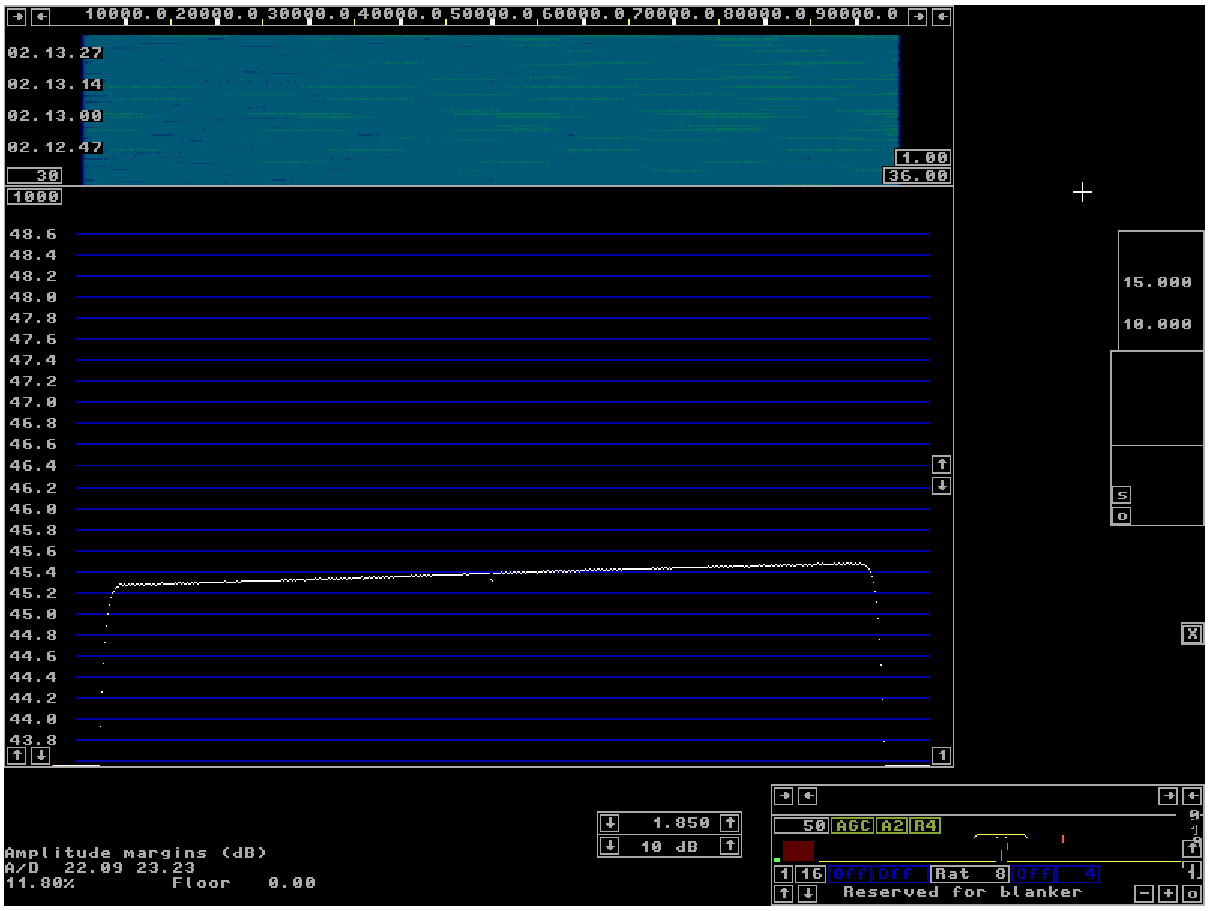
```
Amplitude margins (dB)
A/D   1.71   1.13
12.05%          Floor    0.00
```

Fig 6. *Very strong pulses averaged over a long time on a system calibrated with the same pulses.*



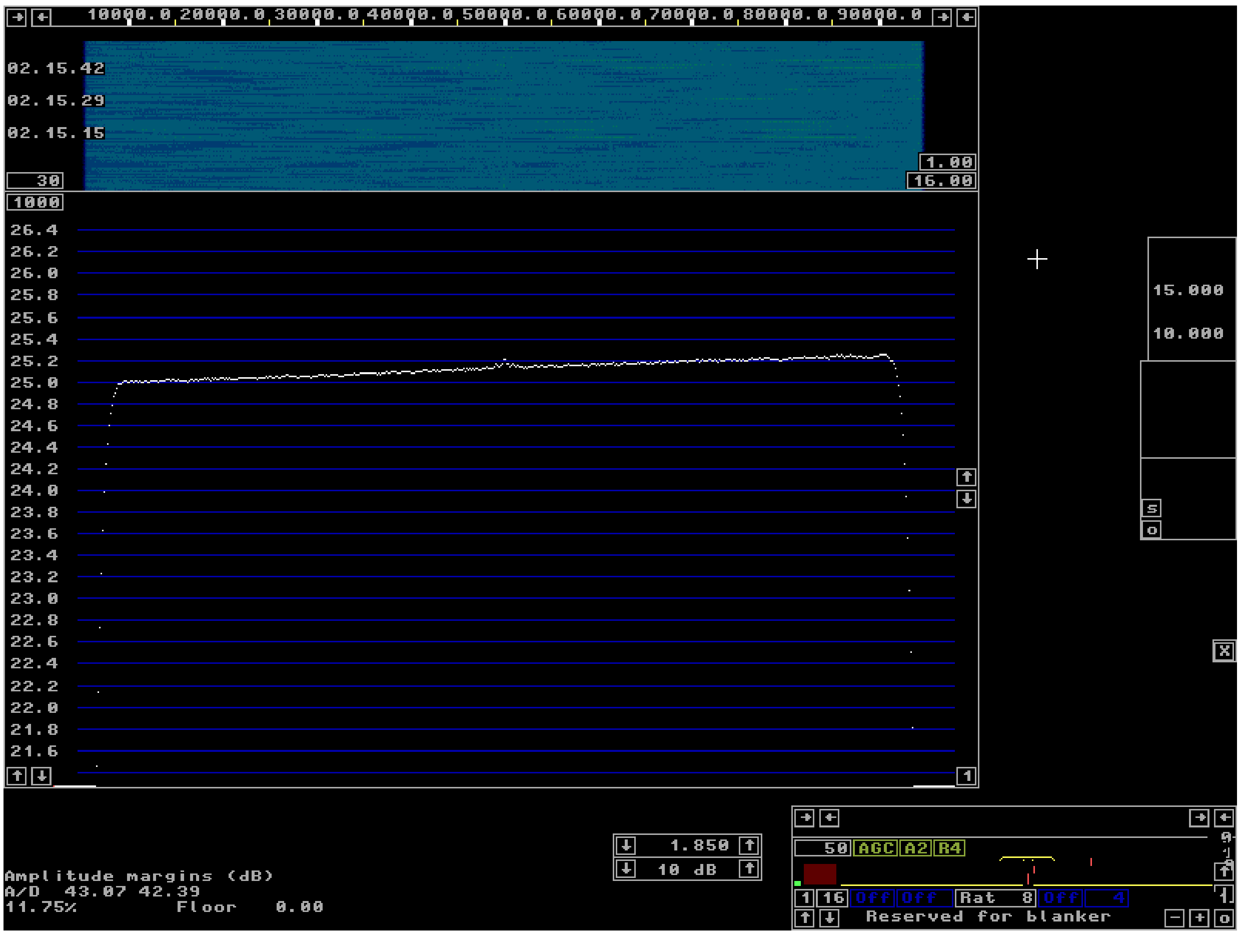
```
Amplitude margins (dB)
A/D   4.41   4.41
12.06%          Floor    0.00
```

Fig 7. *The pulses of figure 6 with a 3dB attenuator.*



```
Amplitude margins (dB)
A/D  22.09 23.23
11.80%         Floor    0.00
```

Fig 8. *The pulses of figure 6 with a 20dB attenuator.*



```
Amplitude margins (dB)
A/D  43.07 42.39
11.75%         Floor    0.00
```

Fig 9. *The pulses of figure 6 with a 40dB attenuator.*

Impedance

The response of an LC filter depends on its source and load impedances. With WSE units on 1.8 MHz, a 300 ohm resistor across the RXHFA input will cause a change of the noise floor that is similar to the difference between figures 6 and 8. This is the change for pulses well within the linear range of the system.
The WSE system is easiest to calibrate at the RX10700 input or through the RXHFA at 7 MHz or so. Calibrating at the RX2500 input is not recommended because that would require a pulse source that has an output impedance that is very similar to the particular RX10700 in use.

In case the input signal is in complex format Linrad has routines to correct amplitude and phase for complex input signals These routines operate in the frequency domain and can absorb frequency dependent phase and amplitude errors that are introduced by differences in amplifiers and filters used between the I/Q mixers and the audio board. The only requirement (non trivial) is that amplitude and phase errors are independent of amplitude, time and temperature.

Correct for amplitude and phase errors in a direct conversion receiver.

(Jan 20 2008)

Complex signals, I and Q pairs.

A complex voltage is two simultaneous voltages obtained from a quadrature mixer, also called an I,Q pair. A complex voltage sampled at a sampling frequency F can represent voltages with frequencies between -F/2 and +F/2. The Nyquist theorem tells us that a (single) signal sampled at the frequency F contains frequencies from zero (DC) to F/2. With two channels I and Q for the same signal, twice as many data points and twice as much information, we should expect the bandwidth to be twice as large compared to the Nyquist frequency for each of the signals individually.

The quadrature mixer which is actually a pair of mixers operated with phases between RF and LO that differ by 90 degrees is difficult to realize in analog circuits. The amplitudes may differ slightly and the phase might not be shifted exactly by 90 degrees. Small amplitude or phase errors in the mixers, amplifiers and filters as well as the A/D converters themselves cause poor separation between positive and negative frequencies.

If there is an amplitude error of 1%, with amplitudes of I=A and Q=1.01A, the complex signal will not only represent the true signal at frequency Fs with an amplitude of 2.01A. It will also represent a signal at the frequency -Fs, the mirror frequency, with an amplitude of 0.01A. The power ratio would be 46 dB and that is not a good enough suppression of unwanted signals in a receiver. Yet we can not hope for better even when using very good components.

It is possible to minimize the mirror frequencies by use of trimmers in the hardware to balance the amplitudes and to set the phase excactly. Unfortunately, in real life, this will only work perfectly for a single frequency. If there are no filters and timing for the two 90 degree LO signals as well as for the A/D sampling clock are controlled by fast digital circuits, it is possible to make the attenuation of the mirror frequency very good over a wide bandwidth by hand tuning the hardware. One can also do the same thing in software. It is obvious that tuning the amplitude is just to multiply one signal with a suitable value that makes the I and Q amplitudes equal. Phase tuning is obtained by adding or subtracting a small fraction of I from Q. Whether in software or hardware, manipulating amplitudes or phases of the signals themselves is error correction in the time domain.

Correction in the frequency domain

In case analog filters are used to protect the A/D converters from strong signals outside the desired passband, the phase and amplitude errors will change with frequency. In a unit like the WSE2500 where very high Q filters are used this phenomenon makes it impossible to get a reasonable performance by balancing I and Q in the time domain. Linrad therefore uses a procedure to do the mirror image cancellation in the frequency domain.

When a false signal occurs at the mirror frequency due to poor mixer balance, the false signal has a fixed phase and amplitude relation to the true signal. Linrad has a calibration procedure that establishes these relations at a selectable number of frequencies. They are then used to linear combine positive and negative frequencies for the false signals to disappear.

The success of the procedure depends primarily on the stability of the quadrature mixer and associated filters and amplifiers. S/N of the calibration signal may also affect the result.

It should be noted that digital quadrature mixers such as the one in a SDR-14 or a SDR-IQ do not have amplitude or phase errors so they need no calibration.
Check this link digital balancing of amplitude and phase for complementary information to the information below.

Calibrating WSE converters

Figure 1 shows an uncalibrated WSE unit when receiving a signal about 10 kHz from the center frequency. The mirror image is suppressed by 40 dB. Figure 2 shows what happens when the signal is moved near the passband edge, about 42 kHz from the center frequency. Here the image frequency is only suppressed by 24 dB, an effect of tolerances on the high Q analog filters

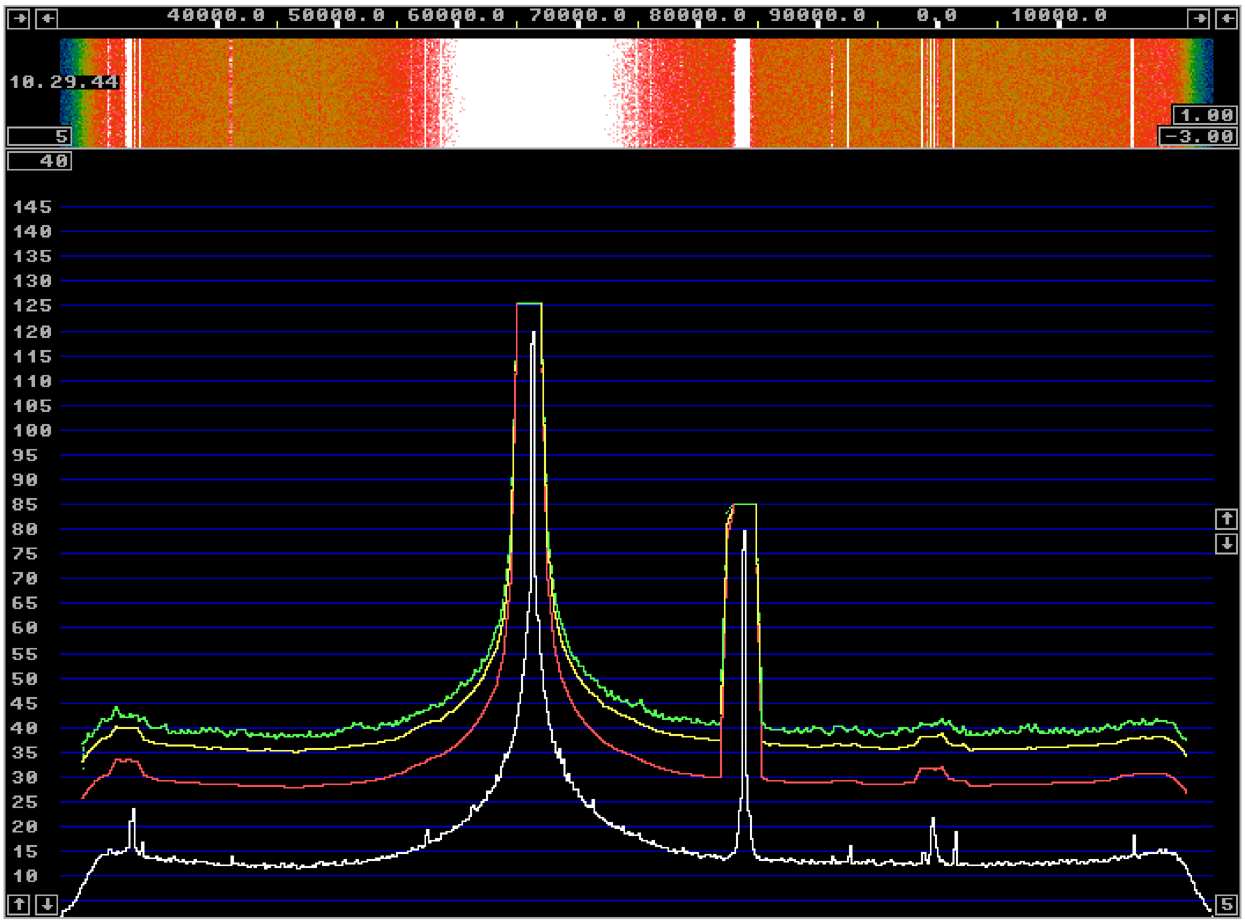
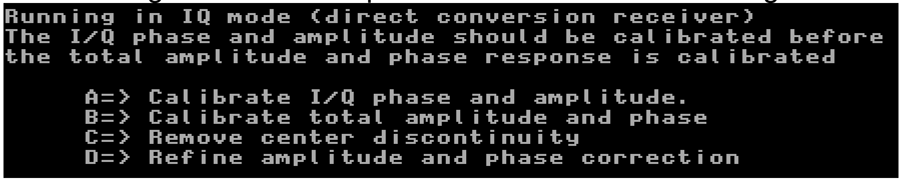


Fig. 2. *An uncalibrated WSE system with a test signal about 42 kHz from the center frequency. At 42 kHz the anti-alias filters cause large errors in the phase and amplitude balance. Everything except the frequency setting of the signal generator is the same as in figure 1.*

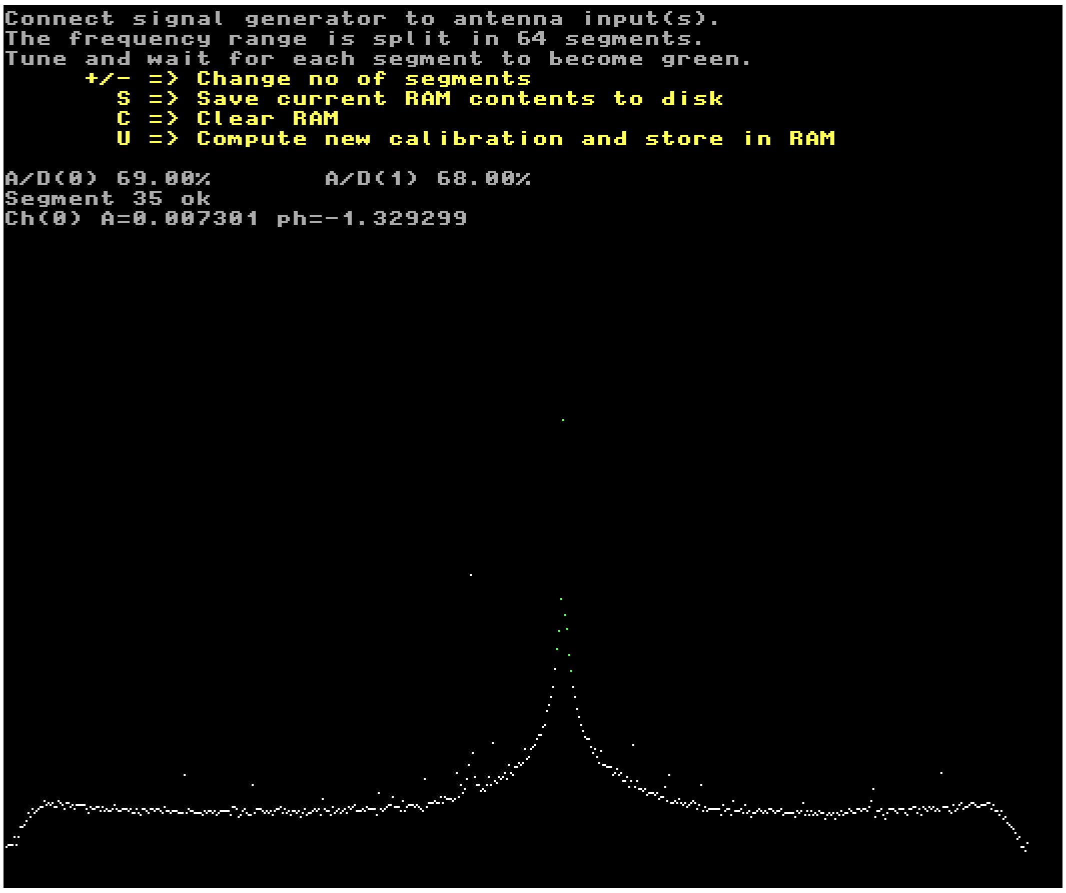
To enter the calibration routine, press X on the normal receive menu, then C for calibrate. That should produce the menu shown in figure 3.
Press A to get to the data input screen which is shown in figures 4 and 5.



```
Running in IQ mode (direct conversion receiver)
The I/Q phase and amplitude should be calibrated before
the total amplitude and phase response is calibrated

      A=> Calibrate I/Q phase and amplitude.
      B=> Calibrate total amplitude and phase
      C=> Remove center discontinuity
      D=> Refine amplitude and phase correction
```

Fig. 3. *The Linrad calibration menu for a system with complex input (I and Q) from a soundcard. With an SDR-14 or SDR-14 the A option is missing.*

Connect a signal generator to the antenna input, feed the same signal into both inputs in case you have enabled two RF channels. Turn the level of the signal generator for about 50% level on the A/D converters. When everything is adjusted to your satisfaction, clear data that might have been collected while you were changing signal levels etc. by pressing 'C'.
In case the system is already calibrated and you only want to refine to absorb effects of ageing or temperature differences, press plus and then minus to clear collected data while keeping the old calibration.



```
Connect signal generator to antenna input(s).
The frequency range is split in 64 segments.
Tune and wait for each segment to become green.
     +/- => Change no of segments
       S => Save current RAM contents to disk
       C => Clear RAM
       U => Compute new calibration and store in RAM

A/D(0) 69.00%         A/D(1) 68.00%
Segment 35 ok
Ch(0) A=0.007301 ph=-1.329299
```

Fig. 4. *The data input screen for the I/Q balance calibration. The magnitude of the error is 0.36% at this point near the center frequency. (The magnitude of the difference between I and Q is 0.73% of the magnitude of the sum.)*

It is convenient to do calibration with a synthesized signal generator that can be set for a step length equal to the segment separation. With 64 segments and 96 kHz sampling, the step size is 1.5 kHz.

The procedure is simple. Step to another frequency. The points should turn from white to red. Then after a while they should turn green. Once the points on the current frequency are green, step to another white region. When all points are green, press 'U' to compute the correction function. You should see a screen like the one in figure 6. When you press enter, the correction will be applied and you will return to the data input screen which might look like figure 7.
After a couple of iterations, the results no longer improve. The correction might look like in figure 8. The image suppression is typically 80 dB as shown in figure 9.

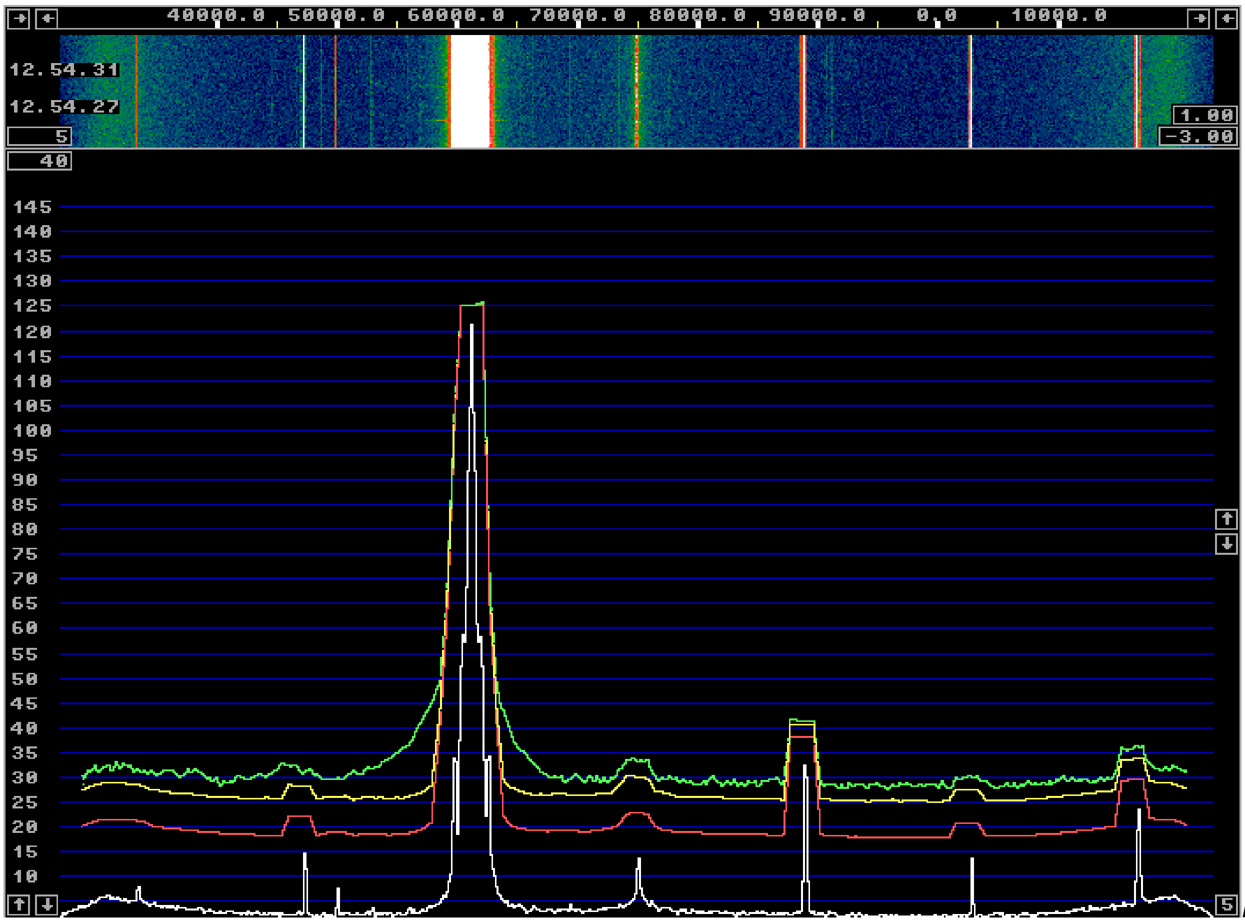


***Fig. 9.** The calibrated WSE system.*

All the calibration was done with a HP8657A signal generator with Linrad in txtest mode (to make easier to see graphs) and with a FFT1 size of 16384. Figure 9 was made with a HP606B which is an old analog generator with about 10 dB less sideband noise. (Compare figure 9 with figure 1.)

By tuning the HP606B slowly over half the screen in txtest mode, one can get the image suppression over half the passband as shown in figure 10. For this figure, calibration was made with an FFT1 size of 65536 and the image suppression is about 90 dB over 50% of the total bandwidth. Narrower FFT bandwidth gives a better S/N for the calibration signal. Such an effort with the calibration is however meaningless with the WSE units. One hour later the image suppression has degraded significantly as can be seen in figure 11. The limitation is the stability of the analog hardware.

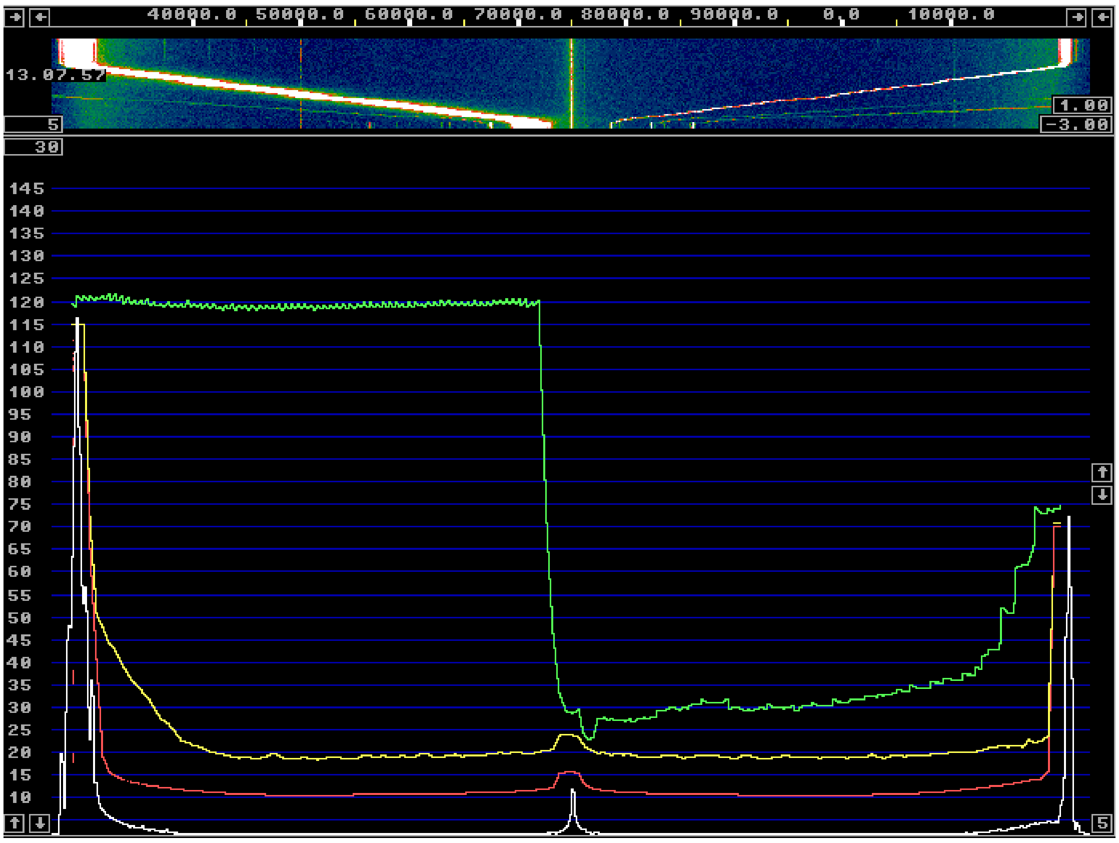


Fig. 10. *A signal swept across half the screen of an accurately calibrated WSE unit immediately after calibration with 65536 transforms for FFT1.*

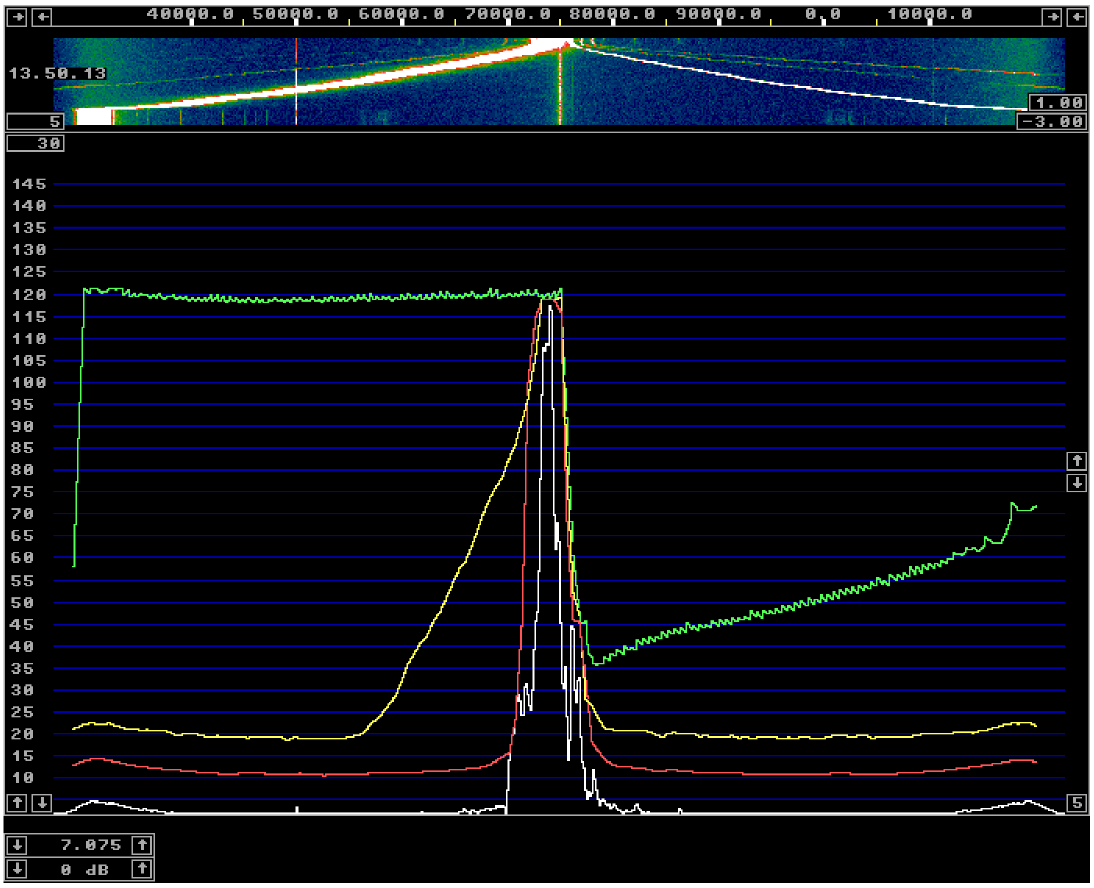


Fig. 11. *The same as figure 10, but one hour later.*

By leaving the signal generator on a fixed frequency and only allowing one segment to turn green, one can rapidly cancel the image on that particular frequency. Figure 12 shows the S-meter for 20 minutes after such a calibration in weak cw mode with two RF channels. Immediately after calibration the image suppression is better than 100 dB, but only for about one minute.

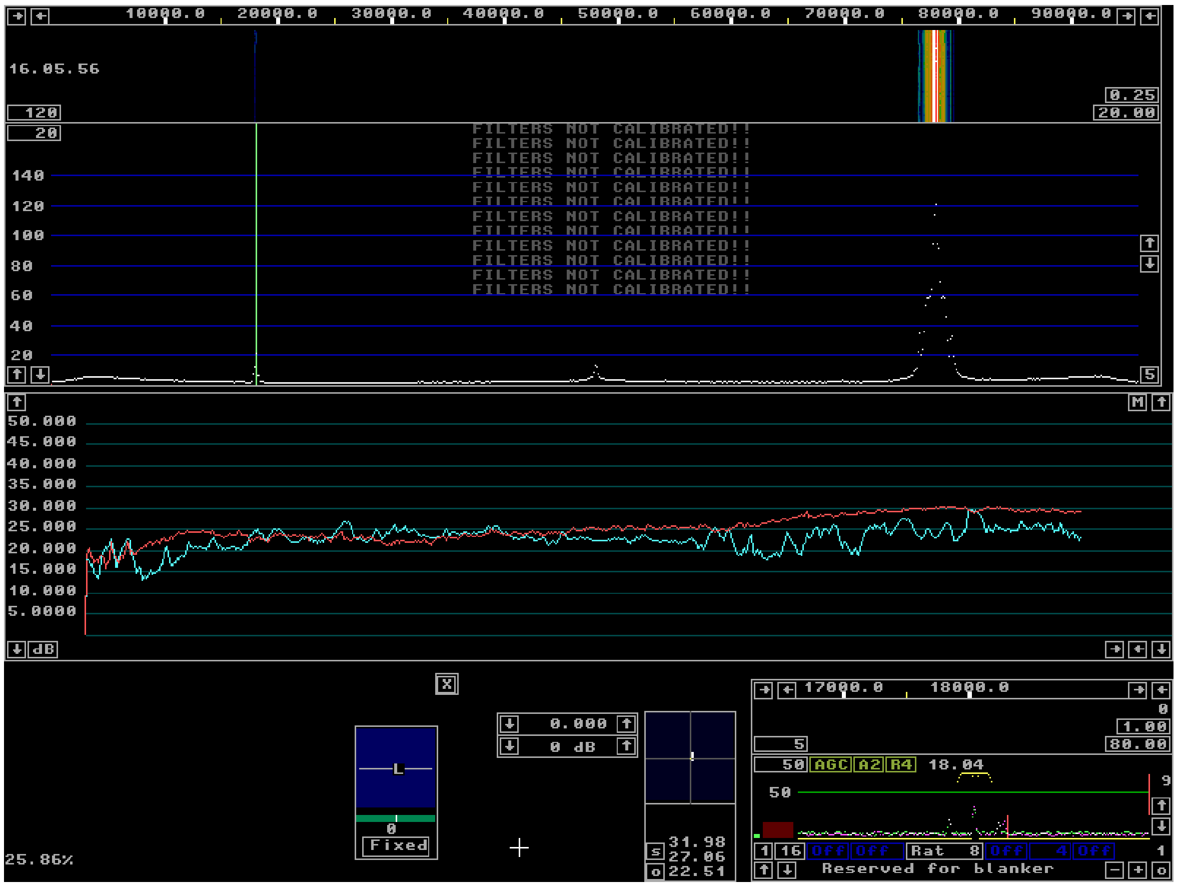


***Fig. 12.** Mirror image suppression during 20 minutes immediately after accurate calibration.*

It is obvious from figure 12 that the two channels are quite different. The blue track, channel 1 (H), seems to suffer from a poor contact somewhere while the red track looks more like thermal drift. The AM(?) noise modulation in channel 1 contributes to the close-in sideband noise of this channel, but it is not visible in the normal sideband noise measurements since the phase noise from the oscillators dominates. When calibrating a few times per year, Linrad with WSE converters should provide a mirror image suppression of at least 70 dB with warmed up units.

Running Linrad

Linrad uses several windows to supply the user with information and to accept commands by mouse clicks. These windows can be moved around and their size can be changed. Left mouse button on the border lines is used.

The wide graph is divided into two parts; a waterfall display and a normal power spectrum. It can show the entire frequency range or some zoomed part of it.

The wide graph.

Fig. 1 shows the wide graph of Linrad and the associated parameter boxes at night while listening on 40 meters.
The main purpose of the wide graph is to locate a signal, to place the mouse cursor on it and press the button to get the signal into the loudspeakers (or better, head-phones).

The wide graph window is split in two parts, the waterfall diagram and the main spectrum. The border line between the two halves can be moved with the mouse.

For real time help on the wide graph as well as on any other Linrad screen object, place the mouse on the object and press F1. To find out what is an object, place the mouse cursor on some empty screen location and press F1.

The waterfall diagram

The numbers at the left is minutes and seconds of the computer clock. The waterfall diagram in fig.1 shows about one minute of a 20kHz wide segment of the CW portion 40 meter band.

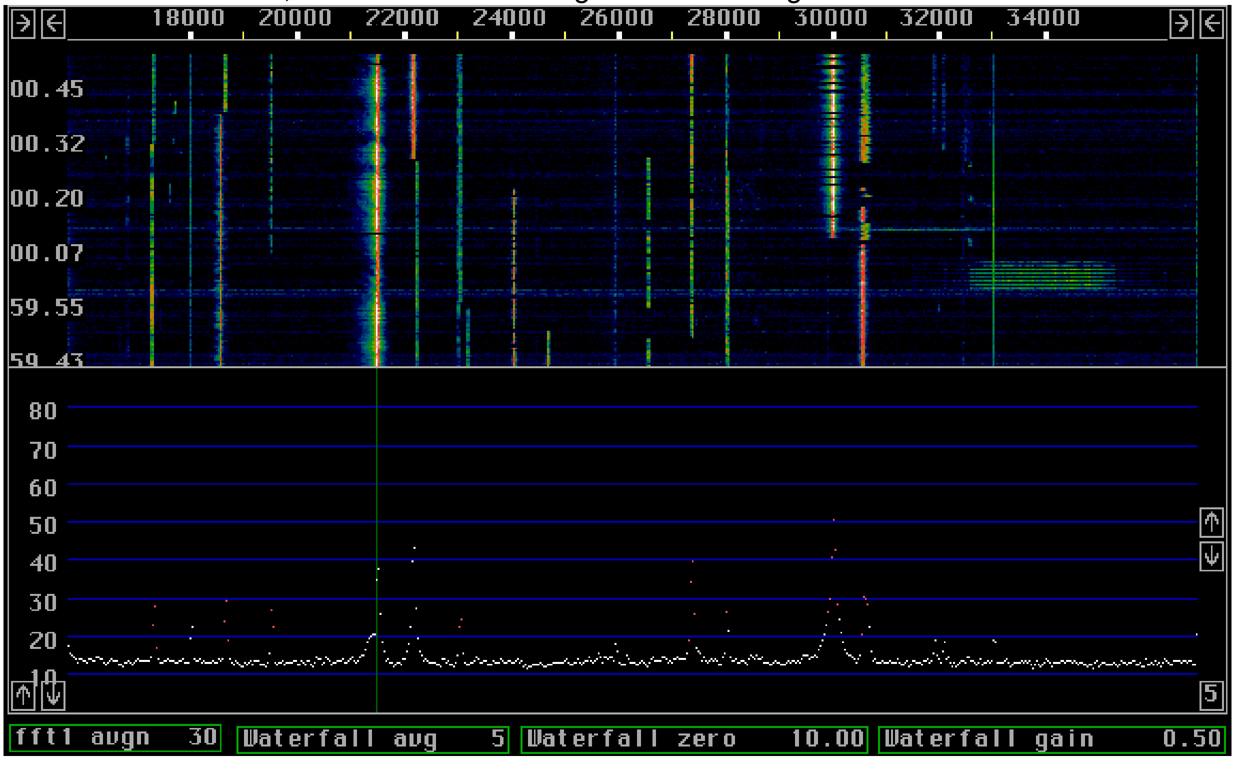
The waterfall diagram is the screen area to watch in order to find the interesting dx signals. What to look for is different on different bands, personally I know well what to look for on 144MHz EME, but I am pretty sure other bands and modes have their particuliarities that one can learn to see on the waterfall diagram. A pileup is very easy to see for example.

If the second fft is enabled, the waterfall shows the output of the second fft which is typically at higher resolution than the main spectrum which is always calculated from the first fft. The waterfall shows the spectrum after the noiseblanker has done its job in case the noise blanker is enabled.

If the second fft is disabled, there is no noiseblanker and the waterfall is calculated from the same transforms as the main spectrum.

The colour scale of the waterfall graph is adjusted by the parameters in the separate parameter boxes **Waterfall zero** and **Waterfall gain**. The time scale is set by **Waterfall avg**

The waterfall diagram is of course affected by the bandwidth and window function parameters selected for fft1 or fft2, whichever is used to generate the diagram.



fft1 avgn 30 | Waterfall avg 5 | Waterfall zero 10.00 | Waterfall gain 0.50

Fig.1. *The wide graph window of Linrad.*

The main spectrum

The main spectrum gives a true picture of what is received. The spectrum is not processed to remove anything. The amplitudes give a true representation of the average power level at each frequency.

When using Linrad for real communication, chasing DX or contesting, the main spectrum is not very interesting so it is then typically compressed to give more space for the waterfall and other interesting things one wants to use the screen area for.

Even in compressed form the main graph will give information about what is going on in frequency segments where the waterfall is saturated.

When using Linrad as a spectrum analyser, the main spectrum is typically magnified to occupy most of the screen. By selecting a high order window the equivalent filter shape of the spectrum analyser get good shape factors. If the second fft is deselected the S-meter in the coherent graph can be used for precision measurements of the power in a well defined bandwidth.

The main spectrum is a tool to monitor the operation of the selective limiter in case second fft is enabled. Those fft bins that have power levels above limit are red in the main spectrum as can clearly be seen in fig. 1. This information is equally or even better seen if the spectrum is compressed. By looking at what points become red one can change the selective limiter levels to give optimum performance for the noise blanker in a particular situation.

The green line at 21.5kHz shows that this frequency is selected and routed to the loudspeakers. Note that the points are all white in the main spectrum. This is the desired signal so it is a good idea to route its frequencies through the noise blanker. Also note that the relatively strong signal at 22.1kHz is white. That is because the selected bandwidth is 1.5kHz for this image. That signal is also within the selected bandwidth.

The vertical scale of the main spectrum is adjusted with the arrowed boxes. The two at the left side expand or contract the scale while the arrows at the right hand side move the zero point up or down.

The main spectrum is averaged over the number of transforms specified in the parameter box **fft1 avgnum.** The number in the box at the lower right hand corner is used to calculate averages of spectra faster. If a low resolution is choosen, the calculation of average power for hundreds of spectra may be useful, but it is very time consuming. To allow faster averages, the main spectrum is averaged in groups as specified by the number in this box. The fft1 avgnum parameter has to be a multiple of this number and if the second fft is disabled, the Waterfall avgnum parameter also has to be a multiple of this parameter.

How to zoom and change the frequency scale

The left and the right side of the wide graph have three controls each. The border line of the graph and two arrow boxes.

The graph can be made smaller by moving a border line inwards and it can be made larger by moving a border line outwards. Moving the border lines does not change the scale, the number of pixels per kHz is unaffected. Moving border lines does not change the start frequency if there is data at the high side to fill the new graph size. If the graph is made bigger than the currently available data allows, the borders will snap back when the mouse button is released.

The boxes are used to expand or contract the frequency scale by moving a section of the graph out from or into the window at the side where the mouse was clicked. If you want to zoom closely to a signal, first move the right border close to it. Then expand the scale by clicking parts of the spectrum out at the left side. Finally pull the right border away from the signal.

The typical usage of the main graph is to display as much bandwidth as possible. Detailed spectra are available in other graphs. The zooming is primarily useful when Linrad is used as a narrowband spectrum analyser to study sideband noise while working on oscillators and transmitters. It is helpful to see the changes in the narrowband waterfall graph when different improvements are tried. One records the narrowband waterfall for a while, presses the **Pause** key and makes the modifications. The waterfall is resumed by pressing any key.

Spur removal and secondary signals.

The wide graph can be used to enable the spur removal for signals visible on the waterfall. Place the mouse cursor on the spur and press **E**.

Although far from completed, Linrad has routines to monitor several stations at the same time. Secondary signals are selected or deselected with the left mouse button, which also can deselect the main signal. The purpose of secondary signals is to decode them and display in ascii what is received to give an idea about the activities of other stations.

The high resolution graph is present only if the second fft is enabled. It shows the second fft power spectrum, always with one point per pixel

The baseband graph shows the spectrum of the signal after it has been shifted in frequency and decimated to a lower sampling speed. This graph also shows what filter is in use for the signal that is routed to the loudspeaker.

The baseband graph in weak signal CW mode

Fig. 1 shows the baseband graph in weak signal cw mode with all buttons artificially enabled. Some of them exclude each other so linrad will always show "off" for some button. The signal is W5UN received with 4 X-yagis. The green spectrum is the signal routed to the loudspeaker or head phones in normal mode. The magenta spectrum is the signal received in the orthogonal polarisation.

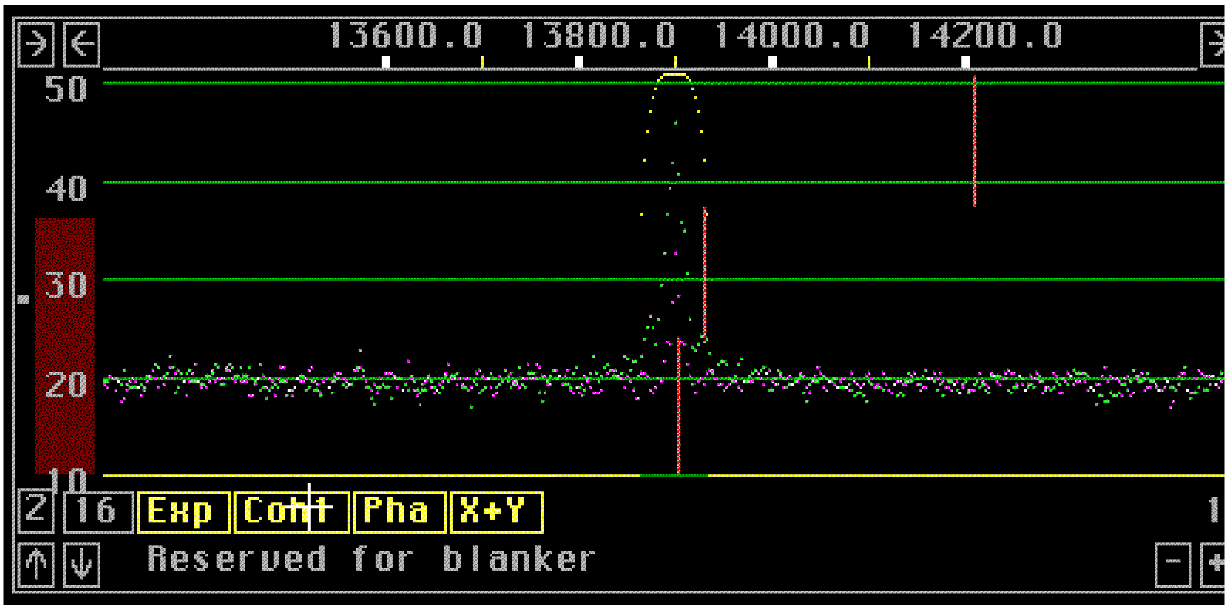


Fig. 1. *The baseband graph in weak signal CW mode.*
The **vertical scale** on the left side, going from 10 to 50, is the signal level in dB. The vertical scale is changed by the vertical arrows in the lower left corner. The zero point is moved with

the vertical arrows at the right side. The red bar under the dB scale is **the volume control** which is changed by a mouse click on the dB scale. The little white dot at the left side of the volume control is the **level indicator** which moves up and down and turns red if the D/A converter is saturated. In weak signal mode there is no AGC, the volume control should be set for the D/A converter to saturate between 10 and 20 dB above the average noise floor. The level indicator shows amplitude in linear scale.

The **frequency scale** can be expanded or contracted in two ways. The arrow boxes at the left side are used to place the fft bins closer or further apart on the screen. The left side boxes do not affect processing at all, only the display. The arrow boxes to the right of the frequency scale change the size of the transform producing the spectrum. The number below shows the current size in powers of 2. The fft size used for fig. 1 is 1024 which is indicated by **10** below the fft size control. The baseband fft size affects the processing delay. Do not use a larger fft than required. If you do not need to see details within the passband it is better to expand the scale with the left side arrows to keep processing delay small.

The red line just below the frequency scale at 14205 Hz is **the BFO control.** It can be moved left or right with the mouse. W5UN produced an audio tone of 300Hz when fig. 1. was produced. The upper BFO control is in the correct place with respect to the frequency scale. The two red lines close to the center of the figure are **BFO controls for zoomed spectra** These controls can be used as an alternative to the BFO control. They are 10 times and 100 times closer to the passband center and allow the BFO to be moved far outside the part of the spectrum shown on the graph.

The yellow dots is the filter currently in use. The **flat passband** at the center can be changed with the mouse in the upper half of the graph. **The shape factor**, is controlled by the width at the bottom of the graph by the mouse in the lower half of the graph.

The number of channels control is the little box containing **2** below the volume control. Some modes produce a stereo signal and if such a mode is selected it is not possible to select a single output channel (mono). For modes that produce a single audio signal only, selecting 1 for number of channels will produce a mono signal which is routed to both loudspeakers by the sound board mixer. If two channels are selected in a mode producing a mono signal, the signal will be sent to the two outputs in opposite phase. Switching the phase now and then may help to prevent fatigue when listening to narrowband noise for a long time.

The **output number of bits control** is the box containing **16** that is placed below the volume control. Clicking on it will toggle between 8 and 16.

The **audio expander switch** is the yellow box containing **Exp** When this box is yellow the output volume is increased for strong signals and decreased for weak signals. The audio expander is intended for use only with really narrow bandwidths, 10% of the selected audio tone and below.

The **coherent CW switch** is the box showing Coh1 in yellow. This box can be switched between **"Off" "Coh1"** and **"Coh2".** Coherent CW means that the phase of the carrier is extracted from the fft bins at the center of the passband. The phase information is used to separate the signal into an I (in phase) component and a Q (quadrature) component. In mode Coh1 the Q signal as well as negative values of the I signal are rejected. The remaining part, the in phase part of the I signal is used to produce a mono signal. In mode Coh2 the I signal and the Q signal are sent to the output as a stereo signal. Coherent modes will work only if the fft size is large enough.

The **binaural phasing switch**, the yellow box containing **Pha** is used to enable generation of an artificial stereo signal from a mono signal. The phase relation between left and right ear is made frequency dependent. This mode is experimental. I do not know if it will improve anything. The effect is that the signal to one ear is delayed with respect to the signal to the other ear. When more bandwidth is used than what is required to include the principal keying sidebands the noise is perhaps less disturbing when artificially spread out spatilaay this way. This mode should be used only with headphones.

The **dual antenna stereo switch**, the yellow box containing **X+Y** is used to send the two orthogonal signals from two antennas to the two output channels. When polarisation rotates rapidly, this is the optimum mode. The two output channels are transformed as specified by the polarisation graph in a way that corresponds to mechanically rotating the antenna. This mode should be used only with headphones.

The **baseband oscilloscope** can be toggled on/off with the little box containing **o** in the lower right corner. The baseband oscilloscope displays the unfiltered baseband signal in the time domain. The vertical scale of the baseband oscilloscope is adjusted with the arrowed boxes next to the **o** box. The baseband oscilloscope is intended to assist adjustment of the baseband noise blanker which is not yet implemented.

The number above the baseband oscilloscope controls is the current **output mode.** (111 in fig. 1) Select this number as the default output mode to get the selected combination of the buttons left of the mode number as the default on starting linrad.

The afc graph is present only if AFC is enabled. It gives information on how successful the AFC is in following the signal.

Automatic frequency control (AFC)

AFC is typically used in FM receivers to compensate for the thermal drift of the receiver local oscillator. In an analog receiver it is accomplished by use of a capacitance diode in the local oscillator that is DC coupled through a low pass filter to the output of the FM detector. When the receiver is incorrectly tuned, the DC voltage at the FM detector output deviates from the ideal value of zero volts. The feedback to the LO frequency via the capacitance diode reduces the error to a level where distortion is not generated. The big advantage of an AFC circuit is that it allows the use of a narrower bandwidth than would otherwise be possible. A 88 to 108 MHz receiver can use 300kHz bandwidth even with a local oscillator that may drift by +/- 150 kHz. Without AFC one would need more than twice the bandwidth with a significant loss of sensitivity as a consequence.

Linrad uses AFC in weak signal CW mode in a similar way. It allows the use of narrower filters than otherwise possible. In CW mode the bandwidth of the signal may be 17Hz and Linrad allows the use of a filter that is only 17 Hz wide. Without AFC, the filter would have to accomodate the combined frequency drift of both transmitter and receiver. Sometimes there may be a varying doppler shift as well. Without AFC the filter will let through something like 2 dB more noise power with a loss of readability as a consequence.

Linrad will also have AFC in SSB mode (some day). The purpose is then different. To get good readability of a SSB signal the receiver has to be tuned to within 50Hz or less which will take some time. The Linrad AFC will save time by automatically tuning the system correctly for the SSB signal on which the mouse is placed. The Linrad AFC in SSB mode will automatically detect if the transmission is made in upper sideband mode or in lower sideband mode and place the BFO accordingly.

AFC in weak signal CW mode

Since the purpose of the AFC is to allow reception of signals that are too weak to be received without any AFC, the AFC has to be capable of finding the frequency of extremely weak signals.

The AFC uses the same spectra as used for the waterfall graph; fft2 if enabled, otherwise fft1.

The optimum bandwidth for the fft is the natural bandwidth of the CW carrier which is typically 0.25Hz for 144MHz EME if both stations have frequency stable equipment. When 0.25Hz bandwidth is used for the fft, the total frequency drift should stay below 0.25 Hz in 8 seconds or 1.9 Hz/min. Few stations are stable enough, a practical bandwidth for the waterfall is 1 to 5 Hz.

Linrad assumes the frequency drift is largely linear with time. When a new signal is selected, the AFC routine forms averages over a large number of transforms to improve the S/N of any signal that may be present. This averaging will typically be over a period of 10 seconds or more which would smear out signals that drift by more than the fft bandwidth over the averaging time. To avoid this problem, Linrad forms several averages with gradually frequency shifted transforms and this way any signal that drifts linearly with time will have the optimum S/N improvement in one of the averaged spectra. The strongest signal found in any of the averages will give the frequency of the signal as a linear function of time.

Once the signal is found, Linrad assumes the frequency drift is not changed so it will try to find the best position for a maximum close to the expected frequency only. Fig. 1 shows the Linrad AFC window. The input signal is the UNKN422.WAV file by AF9Y. The green dots are the average frequencies calculated from a spectrum that is averaged over 11 points symmetrically placed in time around the point in time where the point is placed. The yellow dots are the S/N values calculated from the same average spectrum. The red points is a straight line in 25 points that is fitted to the 25 most recent green points. The fitting is done with the frequency values weighted by the S/N ratio, points in time where the signal is missing does not contribute to the line fitting which is done by the linear least squares fit method.

The white dots are the 17th point of each fitted straight line. The white points are the frequency values to which the final narrow filter is tuned.

The operator may select another number than 11 for the number of transforms to average over. Just place the mouse on the "Avg" box, click and enter a new value. Likewise the length of the straight line and the delay can be changed on the fly. When zero delay is selected, the frequency will be extrapolated from old data and that is of course bad for accuracy - but in some situations avoiding a delay may be more important.

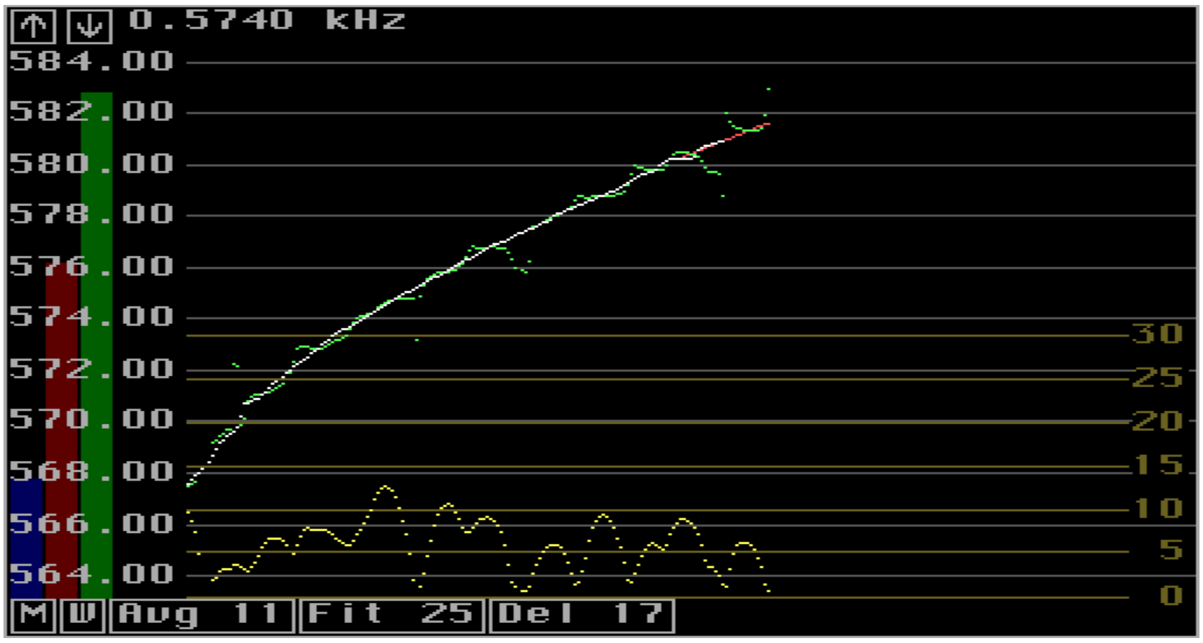


Fig. 1. *The AFC graph with UNKN422 as the input signal. The white dots represent the frequency to which the final narrow filter is tuned.*

The "W" box to the left of the "Avg" box can be used to select or deselect a window function for the formation of the average spectrum. The AFC averaging window produces smoother average frequency curves but the averaging has to be taken over more points for the same S/N improvement.

Fig. 2 shows the waterfall graph produced by the very same transforms as were used to produce fig. 1. There is no averaging in fig. 2, each green dot in fig. 1 is based on the average of 11 transforms but since the window is enabled only the center transform is included with full amplitude, the other transforms are attenuated by a cosine function.

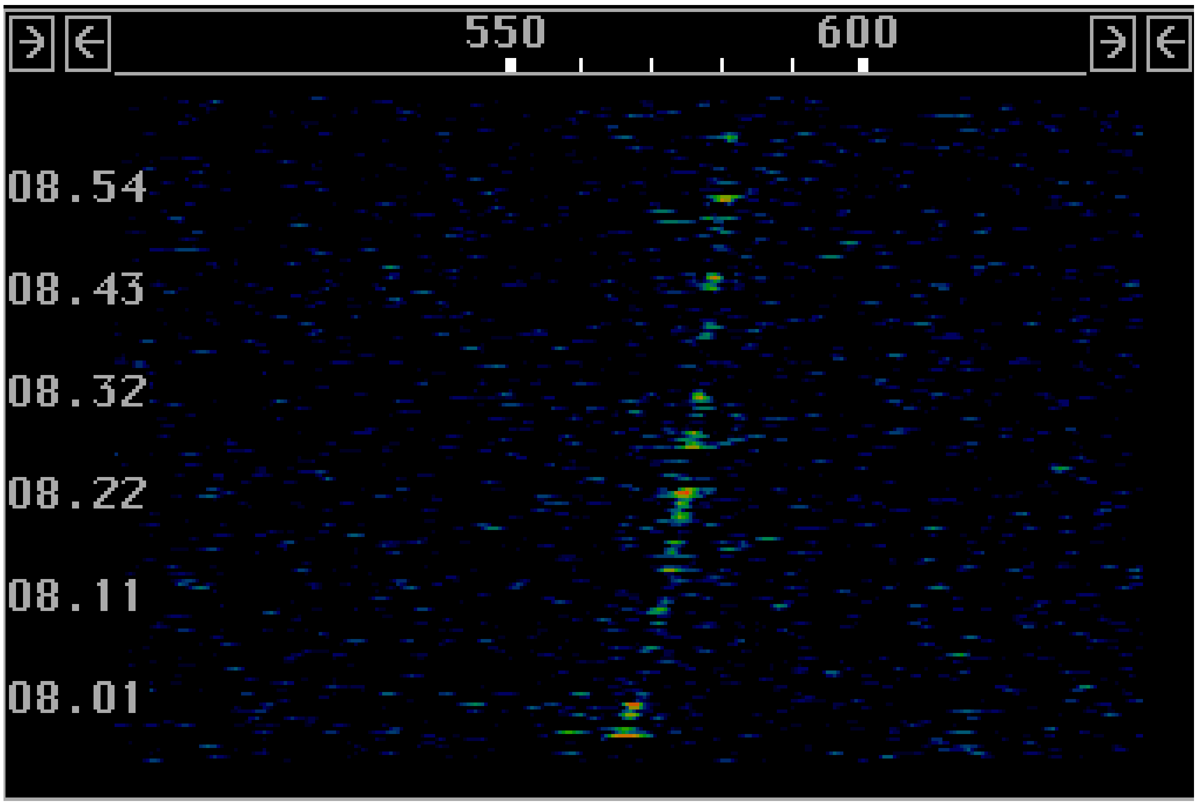


Fig. 2. *Waterfall diagram produced by the very same transforms as those used for figure 1. This waterfall was present on the Linrad screen simultaneously with the afc graph of fig.1.*

Besides the averaging parameters, the bottom row has a box for AFC mode control. The only mode currently available is "M" for manual and "-" for AFC off. Some day there may be an automatic mode....
The three bars at the left side of the AFC graph can be used to control the AFC process. The blue bar sets the minimum S/N required to lock on a signal. If there is no signal present, it is better to stay at a fixed frequency than to track the random maximum frequency of the noise floor. The green bar can be used to limit the search range. The AFC will lock to the strongest signal within the search range and if the operator wants to listen to a weak signal very close to a strong one, this control can be used to make sure that the AFC will search a small enough frequency range for the desired signal to be the strongest one. The red bar can be used to limit the the range over which the search for a maximum is done while the AFC is locked. A signal that drifts continously with time will be very close to where Linrad expects to find it. It is better to flag the frequency as invalid than to search a wider range and possibly get data points based on noise only into the line fitting procedure. Pushing the red bar up to maximum may be helpful for very unstable signals that make occasional frequency jumps. In such cases it is best to set the "Fit" parameter to 1 and to use the smallest "Avg" value compatible with the signal level.

AFC in normal CW mode

Normal CW mode is similar to a conventional receiver with a CW filter. The bandwidth is assumed to be set wide enough to accomodate any frequency drift. A small processing delay is essential but AFC would be of little use.

AFC in meteor scatter CW mode

This mode is identical to the weak signal CW mode. The CW speed is quite different so the bandwidths an timings have to be set accordingly. The frequency drift may be very large, several hundred Hz per second but the propagation path does not broaden the signal so chances are good that the AFC will allow use of coherent averaging, something that would greatly improve the sensitivity for high speed meteor scatter.

AFC in SSB mode

In SSB mode the AFC will be entirely different from the AFC in CW modes. The frequency of the carrier has to be located by an analysis of a large number of low resolution transforms. The human voice is rich in overtones, the corresponding SSB signal therefore has several equidistant peaks that can be extrapolated back to the carrier frequency. On the basis of a single spectrum it is not possible to decide where the carrier is, there will be several possible positions. By looking at several spectra it is possible to find the only position which is consistent with all of them. The frequency of the human voice varies during normal speech. Therefore the separation between the equidistant peaks vary.

The polarization graph is present only if two receiver channels are present. It shows the current receive polarisation which is set automatically or by hand.

Using the noise blanker

The noise blanker in Linrad is quite different from conventional noise blankers. To use it to its full capability, good hardware that allows high bandwidth and good dynamic range is required and it is necessary to have the system properly calibrated.

The signal on which the blanker operates does not contain the entire spectrum that is received by the computer, those parts of the spectrum that contain strong signals are excluded. Averaged spectra are used to determine what frequencies to exclude so averaging parameters and to some extent fft sizes affect blanker performance. The exclusion of strong signals also depends on threshold levels which can be set by the operator depending on the interference situation.

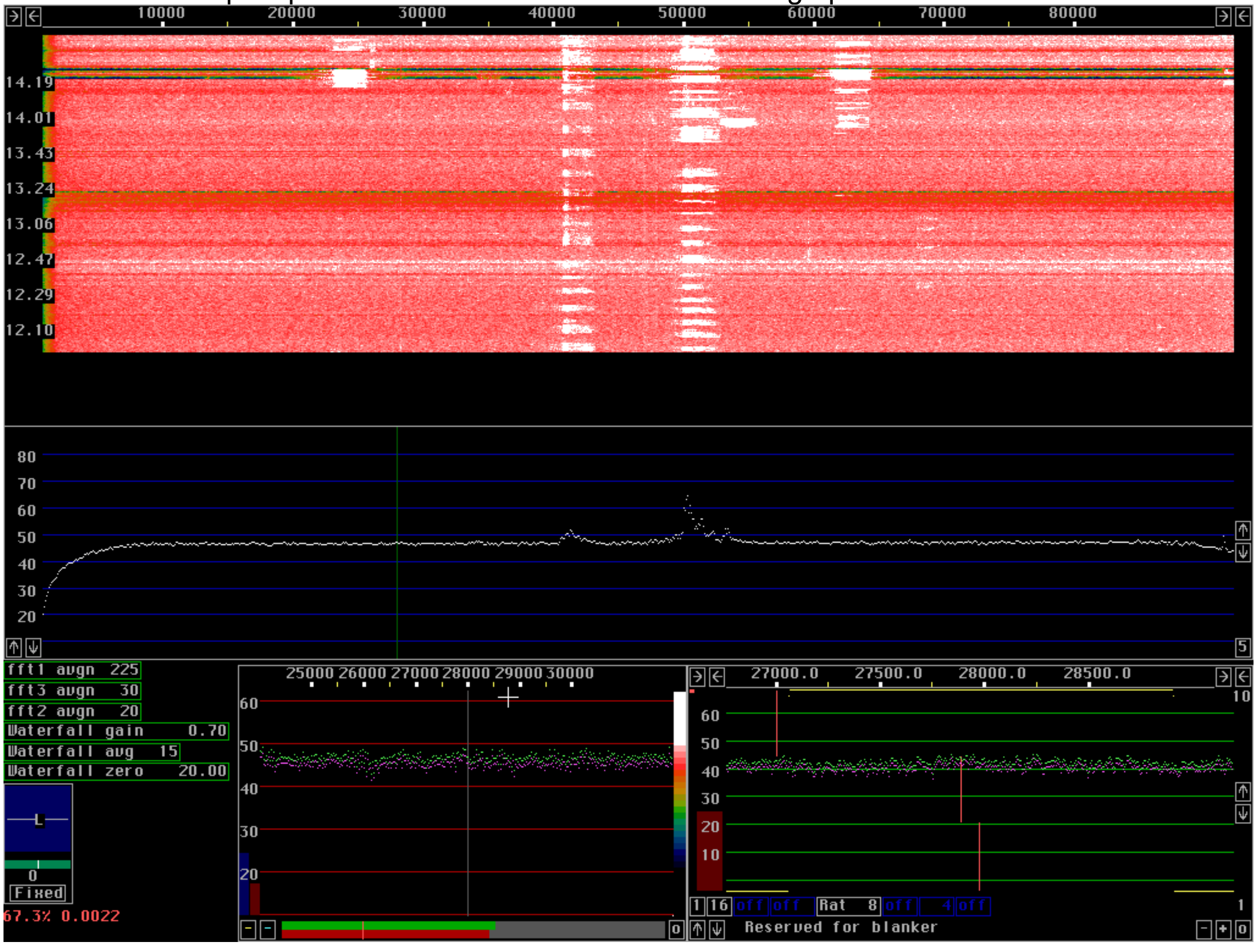
The blanker itself operates in two steps, both of which have threshold levels that decide if a pulse is strong enough to be treated by the blanker.

To get the most out of the blanker a good understanding of its way of operation is required. A real life demo of the Linrad blanker shows what the noise blanker can do to make it possible to receive weak signals under really difficult circumstances. This is a sequence recorded during the leonids meteor shower 2001 with severe powerline noise and several very strong signals in the passband. The sequence is processed in different ways in order to explain the different processing steps.

The Leonids 2001, many signals and severe powerline noise

Fig. 1. shows the screen of linrad when the sequence is processed without any noise blanker. The noise floor is raised by about 25dB so it is approximately at S9 with a conventional S-meter with 3dB for each S-unit.

All screen dumps on this page show the same sequence. For this page linrad00-32 is used. This version has some improvements in the blanker algorithms that make the performance slightly better compared to earlier linrad versions. In fig. 2. the processing is stopped during one of the short quiet periods which is a clear demo of the high power line noise level.



```
fft1 avgn  225
fft3 avgn   30
fft2 avgn   20
Waterfall gain     0.70
Waterfall avg   15
Waterfall zero    20.00
0
Fixed
67.3% 0.0022
1 16 off off Rat 8 off 4 off
Reserved for blanker
```

Fig 1. *No blanker. Note that the powerline noise has short quiet periods.*

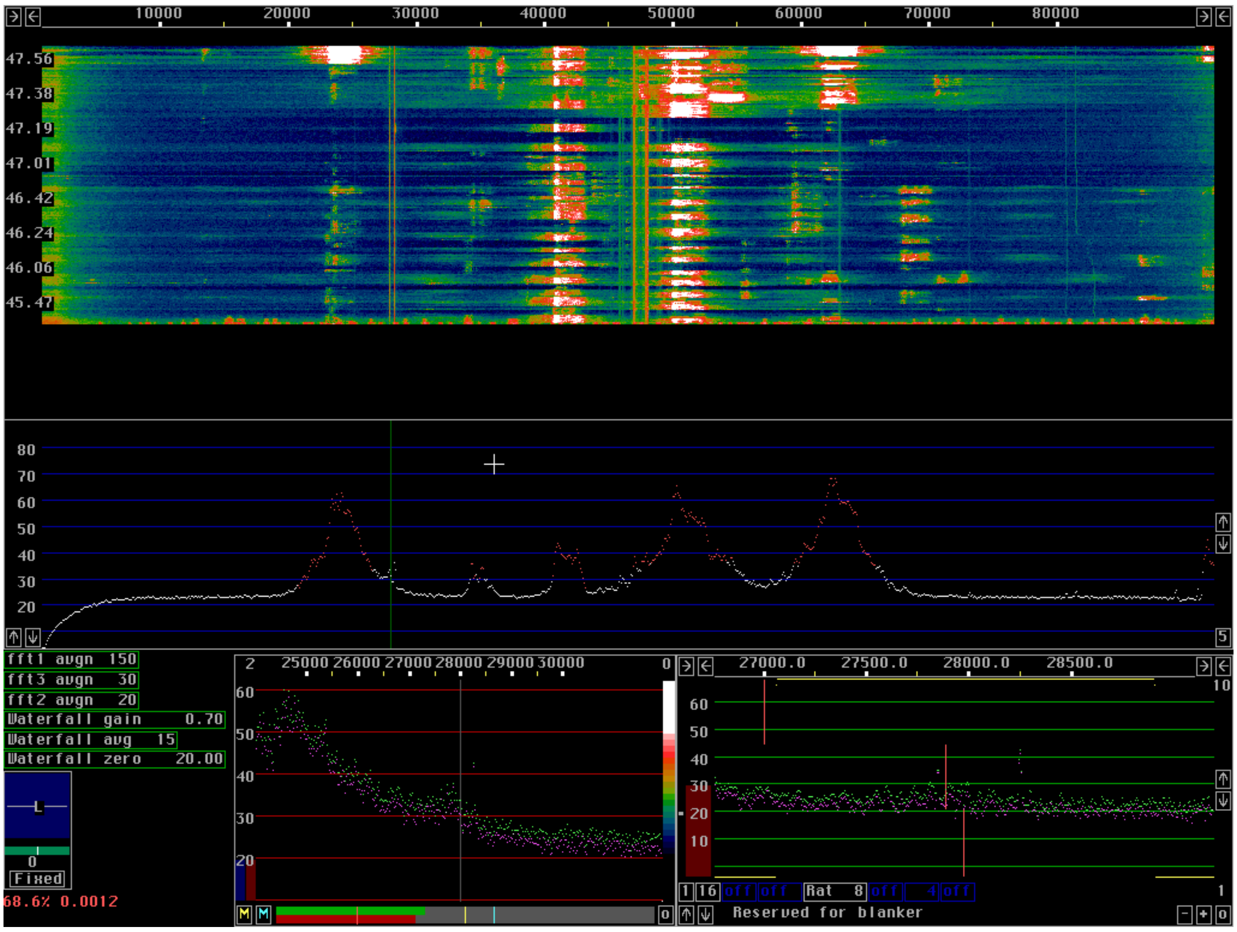


Fig 2. *screen during a short pause in the powerline noise. The fft1 averaging is made somewhat smaller in order to make the spectrum reach its noise free level quicker. This does not affect the noise floor level at all. Note the noise bursts at 31kHz, time 45.50 for example. These bursts are generated by a too low setting of the dumb blanker level, the yellow line at the bottom of the high resolution graph. When what is not excluded from the two strong SSB signals adds up to reach the dumb blanker level, intermodulation between the two is produced.*

Fig. 3. shows the blanker in proper operation with both steps enabled, both the smart blanker and the dumb blanker. I do not have a modern transceiver with a good noise blanker in it so I can not tell how linrad compares to conventional noise blanking, but compared to my old FT221 linrad is much better.

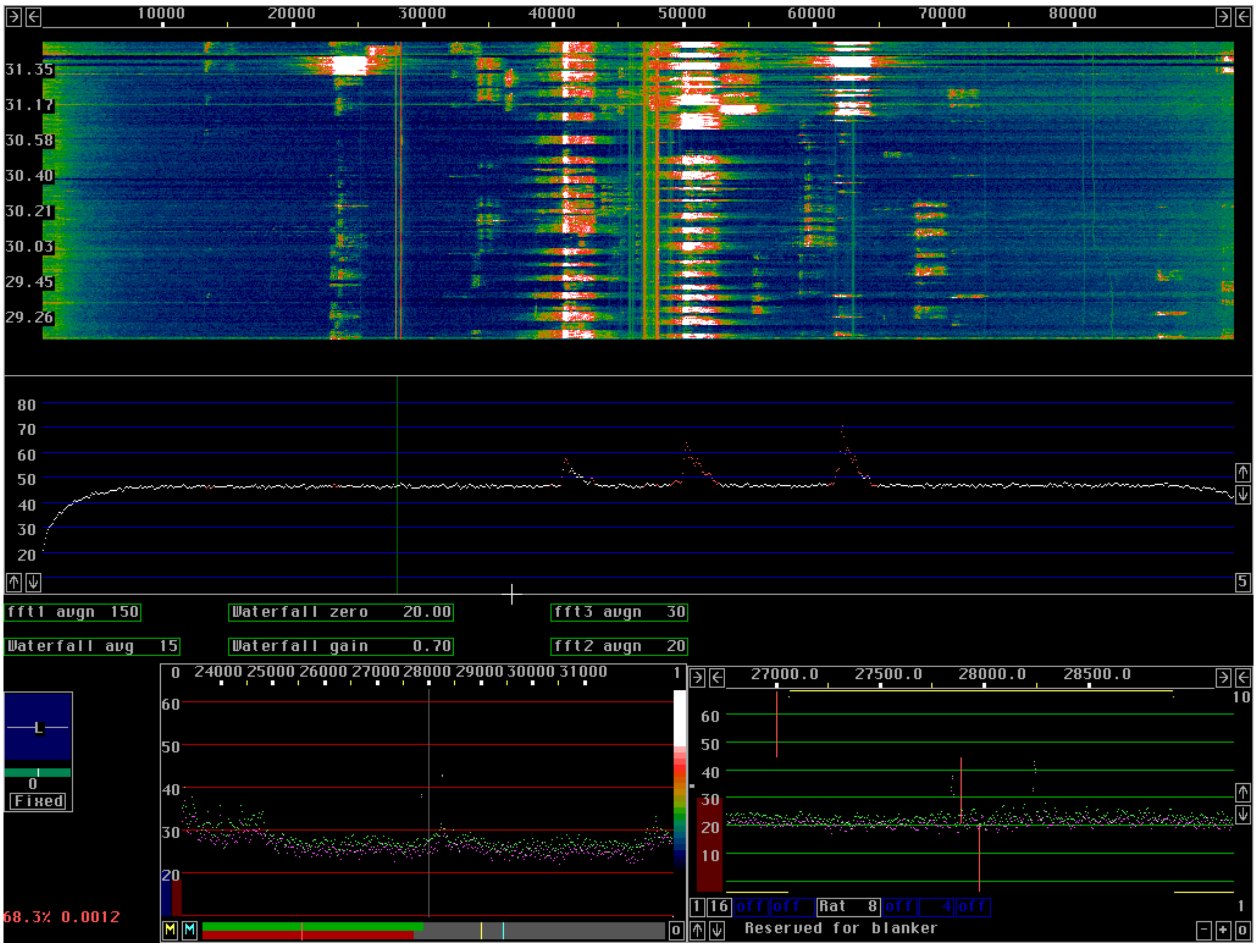


Fig 3. Normal operation of the linrad noise blanker. Both the smart and the dumb blankers are enabled. The noise floor is close to the level it would have been at if the powerline noise had not been present at all. The short periods of no powerline noise are nearly invisible and make a very small difference in the audio output.

Excluding frequencies.

The main spectrum (blue dB scale) is used for a first search for strong signals. This spectrum is searched in two ways, fast and slow. The average is calculated in two steps, first the number of spectra indicated in the little box in the lower right corner is scanned for signals that are strong enough to possibly saturate further processing. For the processing on this page the first fft resolution is 300Hz with a sin to power 3 window so the average of 5 spectra is computed in about 15 milliseconds. The main spectrum averaging number is 150 so it represents the average over about half a second. The slow average is searched each time 5 new spectra are included so signals well above the noise floor are detected much quicker than in 0.5 seconds.

The blue vertical bar at the left side of the high resolution graph sets the S/N level at which signals are excluded. This level can be set lower if long averaging times are used because then the noise floor becomes flatter.

Searching for signals in the main spectrum does not work well in situations with continous powerline noise as one can see from a comparison between fig. 2 and fig. 3. The powerline noise has lifted the noise floor by about 25dB so only signals that are more than about 25dB above the white noise floor can be found by use of the main spectrum.

The most recent line of the waterfall graph is used for a second search for strong signals. Here noise pulses are not present any more so much weaker signals can be found in a difficult situation as the one used for this page.

The S/N level for deciding if a frequency shall be excluded when searching the high resolution spectrum (fft2, the last line of the waterfall) is set by the red vertical bar in the high resolution graph. How to set this level depends on the waterfall averaging chosen.

All spectrum points that are considered to contain strong signals are red in the main spectrum. The red and/or blue bars are set too low if red points occur at frequencies where there is no signal. Exactly how to set these levels for optimum performance is unclear at the moment. I need several more difficult test cases to get some experience. Maybe it is quite different on crowded HF bands compared to nearly quiet VHF bands as one normally has during EME operation.

Excluded frequencies are not rejected, they are just passed outside the noise blanker and added to the blanked signal. In case the amplitude is very large, the excluded frequencies are attenuated to prevent saturation in the high resolution fft which uses 16bit only to take advantage of the very fast MMX (multimedia extension) instructions of modern processors.

Effects of the blanker on excluded frequencies

To show what happens to excluded frequencies, a part of the spectrum from 67kHz to 82 kHz is artificially excluded. Fig. 4 shows the dumb blanker works in this situation. As one would expect, the excluded frequencies contain the powerline noise as if no blanker was present at all, but other frequencies are nearly free from the powerline noise.

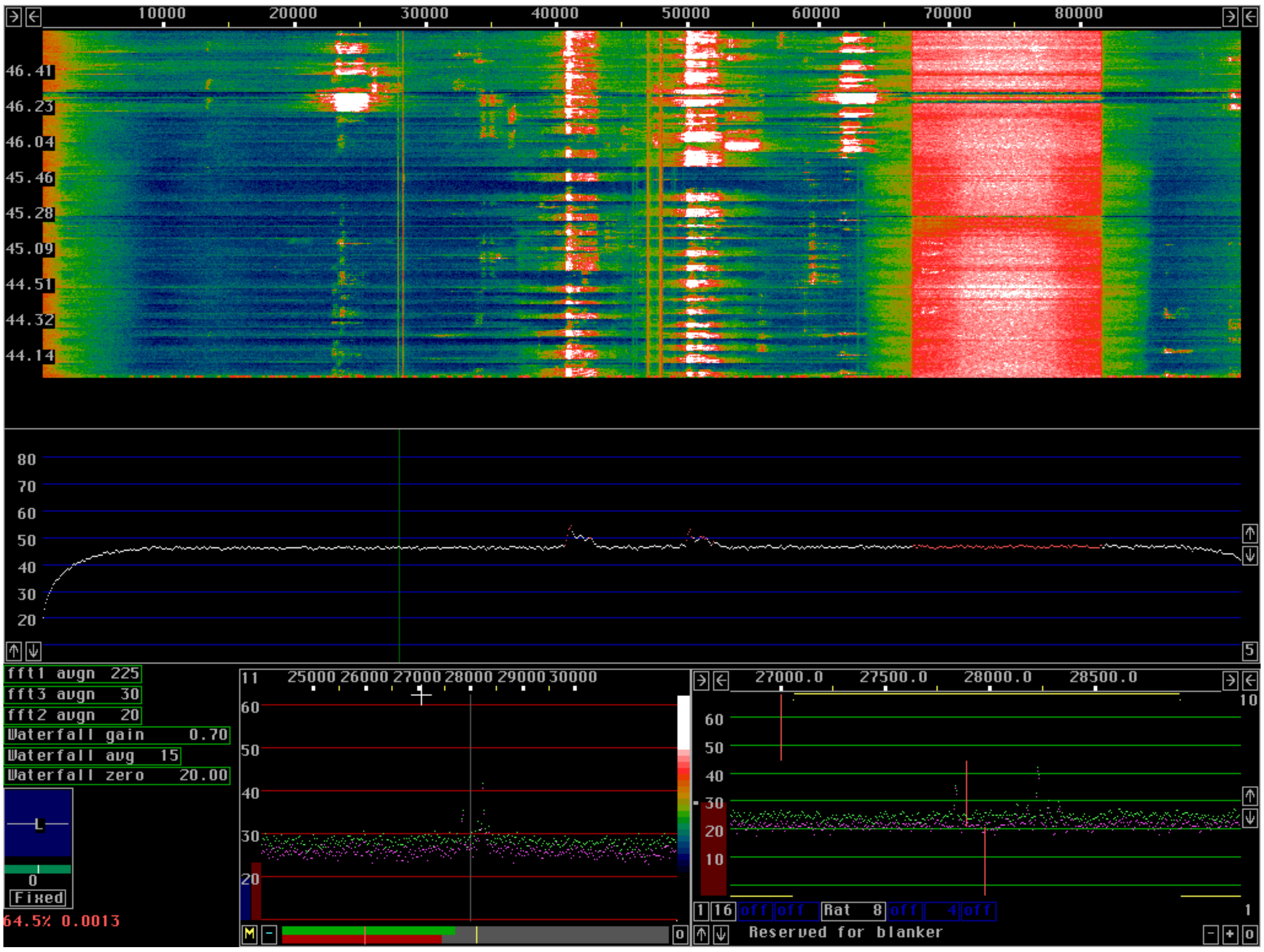

```
fft1 avgn  225
fft3 avgn   30
fft2 avgn   20
Waterfall gain      0.70
Waterfall avg   15
Waterfall zero     20.00
64.5% 0.0013
```

***Fig 4.** Dumb blanker only. The frequency range 67 to 82kHz is excluded from the blanker as if there was a strong signal there. As a consequence the noise level is about 25dB higher in the excluded frequency band.*

The smart blanker that subtracts a pulse with the shape that is given by the pulse response of the hardware behaves differently for excluded points. Fig.5 shows what happens if only the smart blanker is operated on excluded signals.

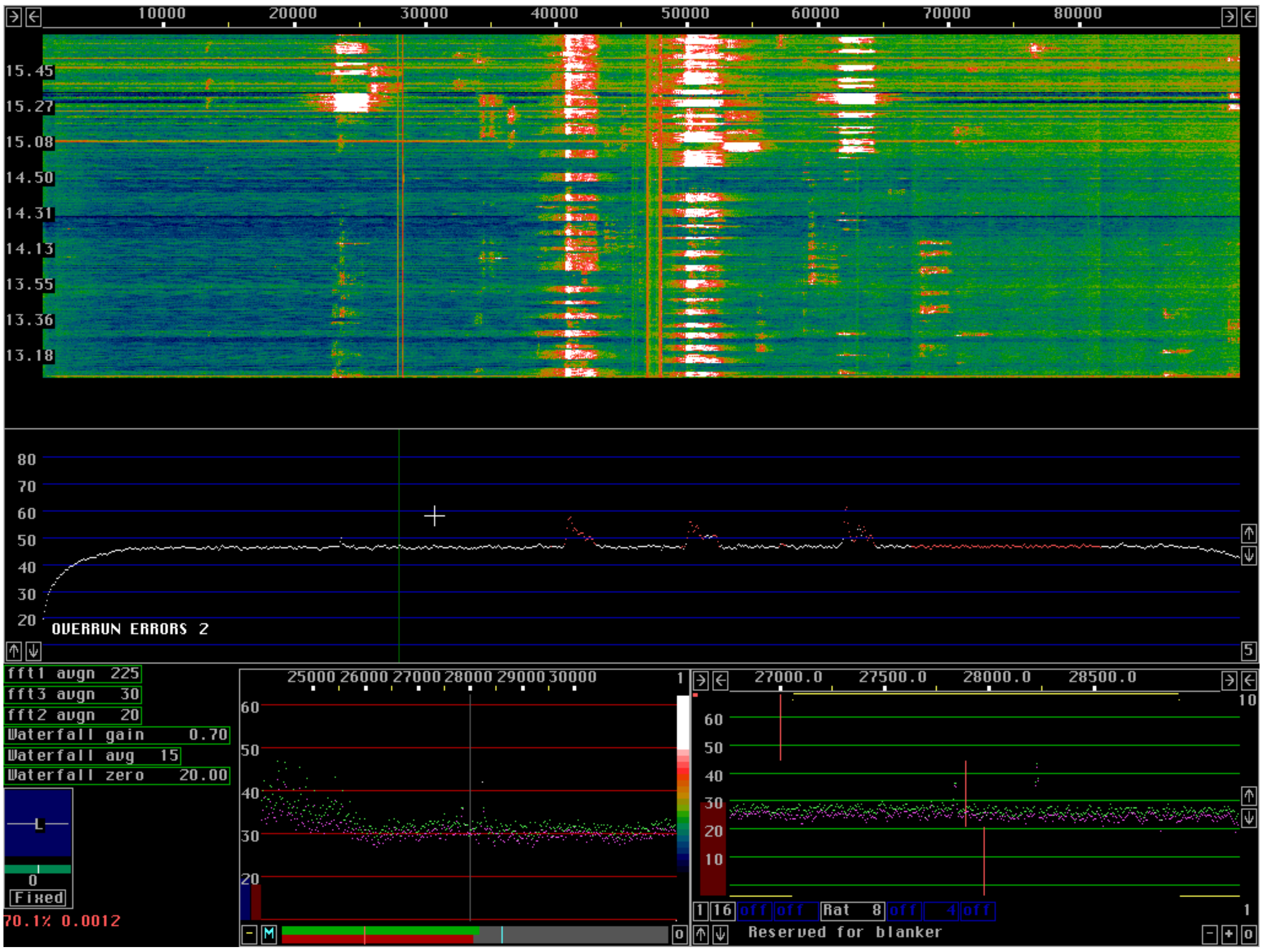


Fig 5. Smart blanker only. The frequency range 67 to 82 kHz is excluded from the blanker as if there was a strong signal there. Note that pulse noise is removed from the excluded frequency band!!!

Since a reference pulse is removed **with the pulse shape it would have if no points were excluded**, the signal does not become zero during the time of the pulse. What remains during the pulse time after subtraction of the reference pulse is a smaller pulse that will cancel the pulse present in the excluded frequency range.

Under normal operation the dumb blanker is operated after the smart blanker. If the threshold of the dumb blanker is set very low and if many points are excluded, the dumb blanker will clear points which actually constitute correct data produced by the smart blanker to cancel pulse energy in the excluded frequencies. Fig. 6 shows both blankers running with a too low level for the dumb blanker.

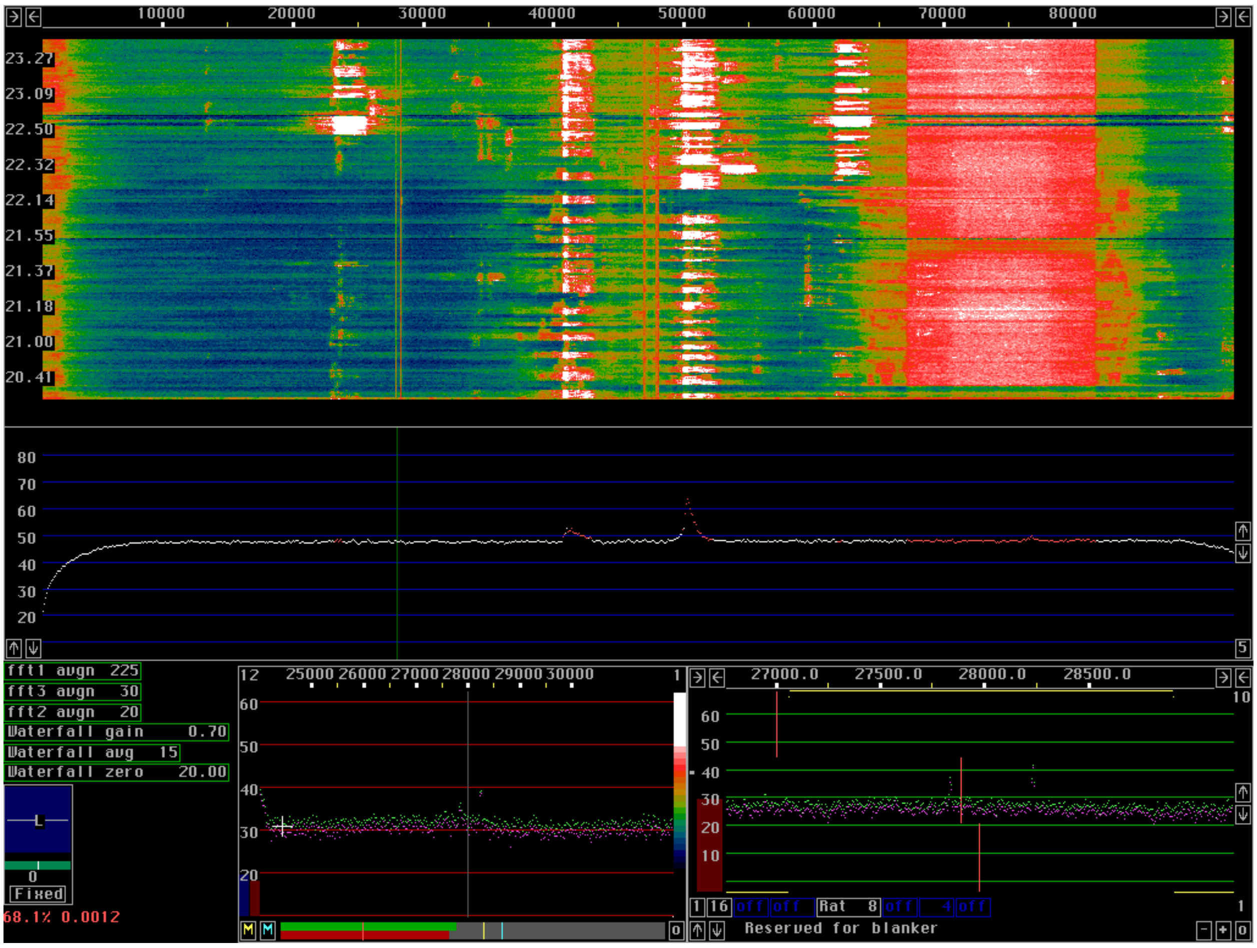


Fig 6. *Smart and dumb blanker together. The frequency range 67 to 82kHz is excluded from the blanker as if there was a strong signal there. Due to a low setting of the level for the dumb blanker, the pulse that would cancel the pulse in the excluded frequency range is removed.*

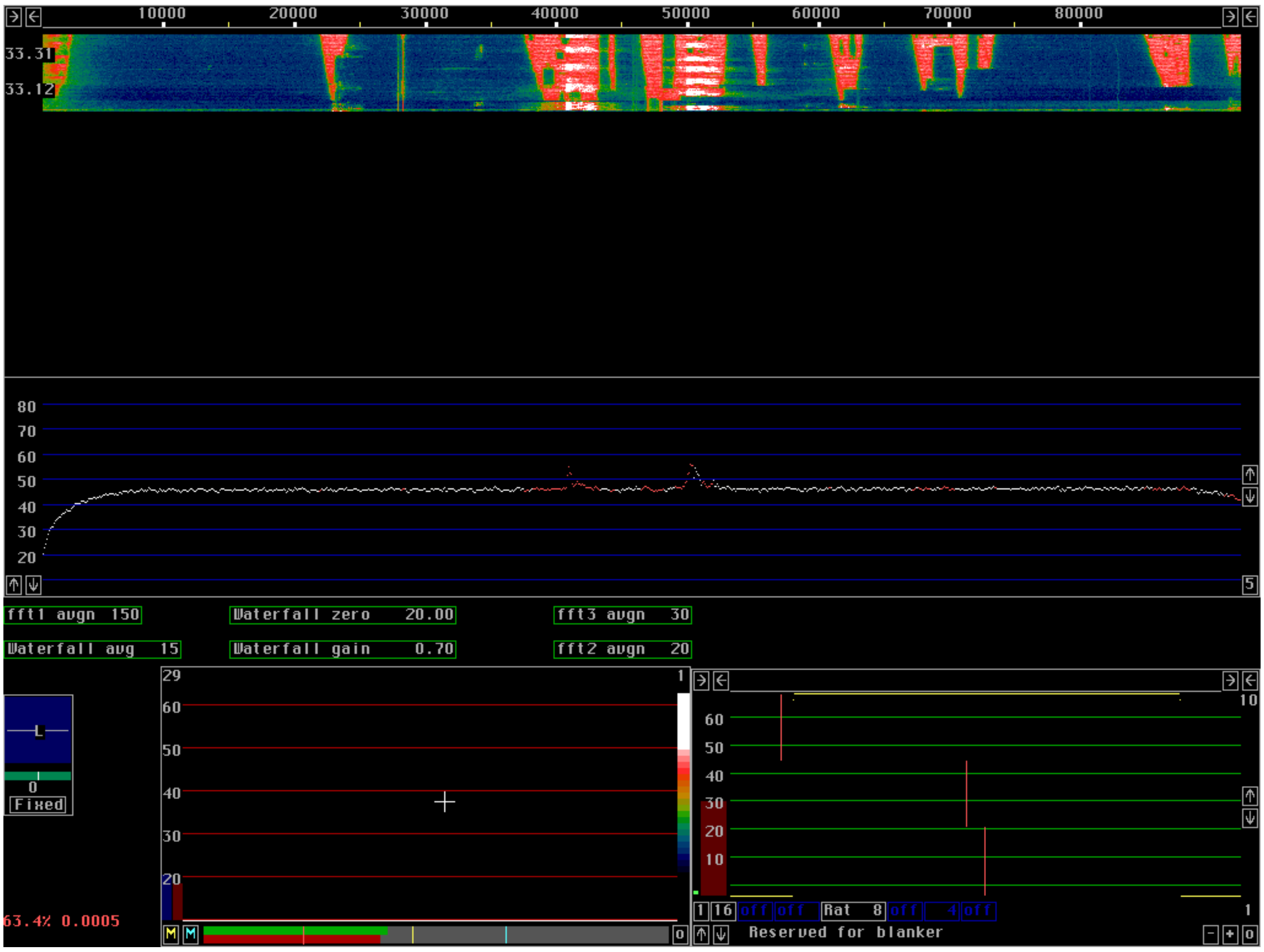
Practical considerations

When experimenting with the blanker it is a good idea to select minimum size for the baseband fft (fft3) in order to have a fast response on parameter changes. Listen to the desired frequency and adjust all four blanker controls for best S/N.

Setting controls too high is safe but will not give optimum performance.

If the blue vertical bar is too low for the fft1 averaging chosen, points will be excluded due to random variations in the noise floor. This is easily detected in the main spectrum, red points that have no corresponding signal in the waterfall graph. To look for this phenomenon, disable the dumb blanker because it generates signals in the excluded frequency range.

If the red vertical bar is too low and the dumb blanker is operating with a low threshold, random frequency bands with artificial strong signals are produced as shown in fig. 7.



fft1 avgn 150
Waterfall zero 20.00
fft3 avgn 30
Waterfall avg 15
Waterfall gain 0.70
fft2 avgn 20
63.4% 0.0005
Rat 8

Fig 7. *Smart and dumb blanker together. Dumb blanker threshold is very low and the red S/N control bar is low. Once a frequency is excluded it will stay excluded since the noise produced in excluded frequency bands by the dumb blanker is strong enough to keep them excluded. If patterns of this kind show up on the waterfall graph the red bar has to be moved upwards or the yellow line has to be moved more to the right. Increasing the waterfall averaging may also be helpful.*

Blanker levels may be set too low causing severe loss of data. This error is shown in fig. 8.

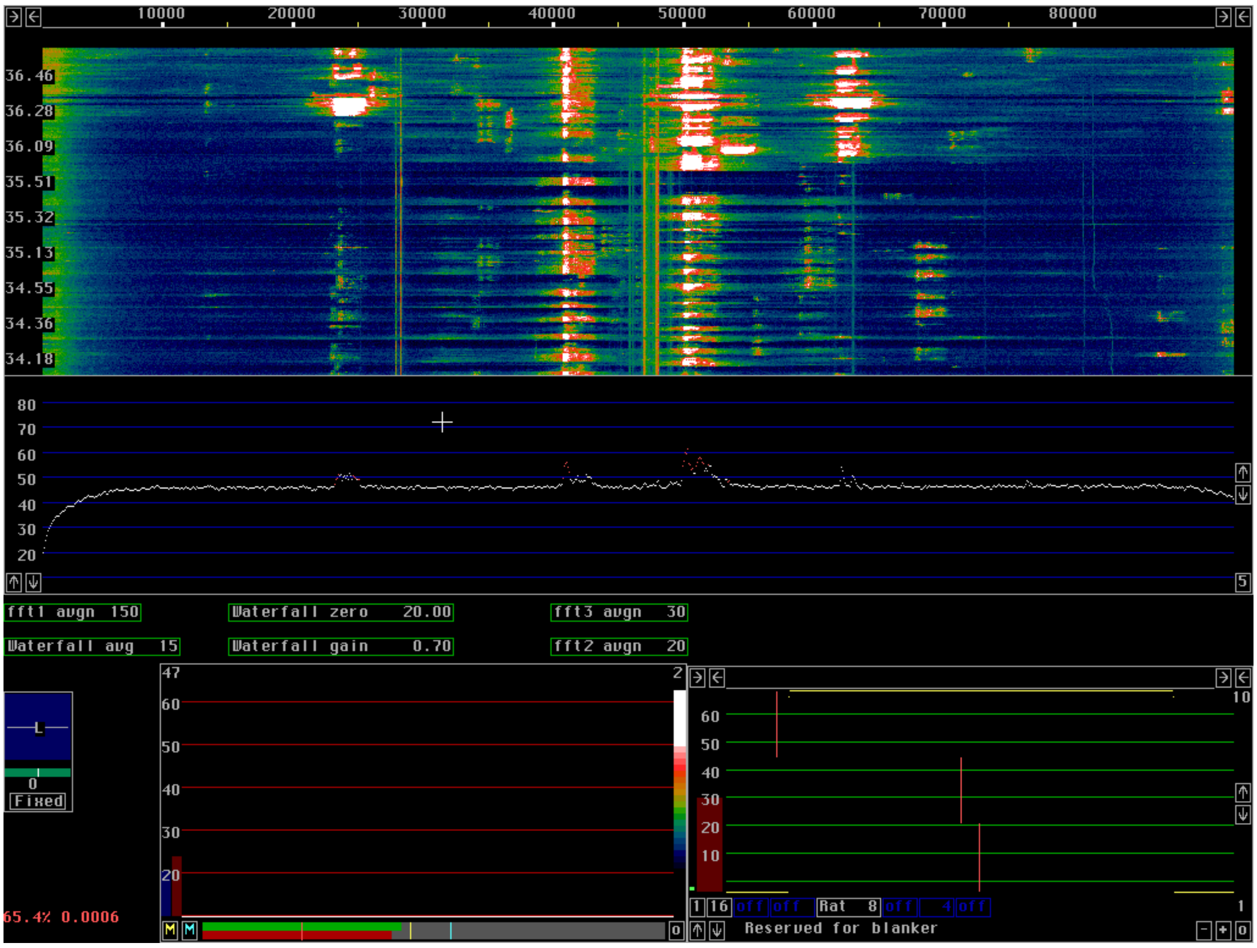

fft1 avgn 150
Waterfall zero 20.00
fft3 avgn 30
Waterfall avg 15
Waterfall gain 0.70
fft2 avgn 20
65.4% 0.0006
Rat 8
Reserved for blanker

Fig 8. *Operation with too low blanker thresholds looks fine in the waterfall graph but will not produce good S/N in the loudspeaker. Here 47% (upper left corner) of all data points are cleared which means that 47% of the desired signal is lost! Moving the yellow control line to the right will improve S/N by increasing the signal more than increasing interference.*

The time functions

To assist in understanding the blanker operation linrad has oscilloscope functions. The two groups of signals, the weak signals on which the blanker operates and the excluded frequencies, the strong signals, are displayed by a mouse click on the little box with an "o" in the lower right corner of the high resolution graph. Fig.9 shows a typical example from the data used for the images on this page.

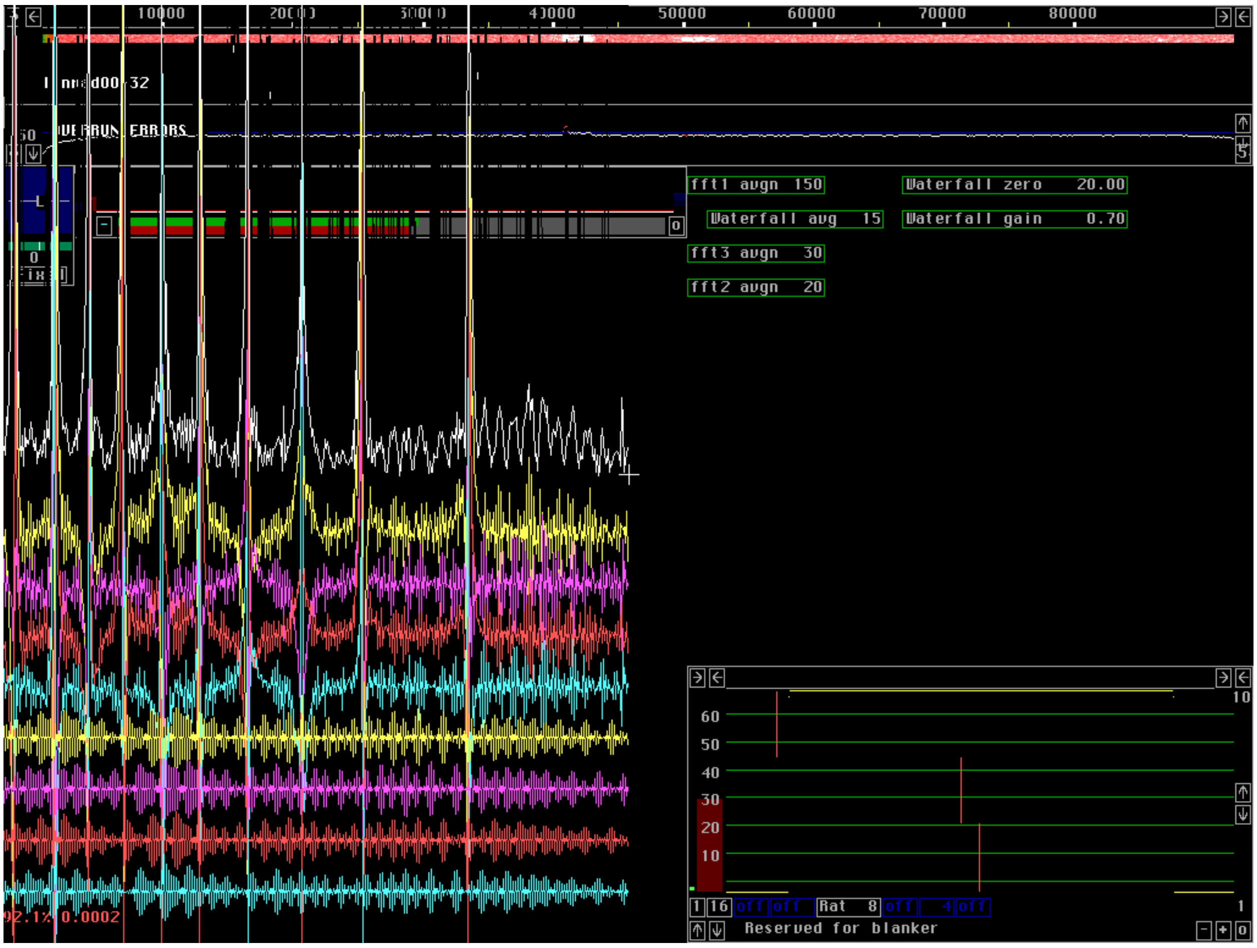


Fig 9. *Typical example of powerline noise. The white track is total power of weak signals, the sum of the squares of the upper four coloured tracks.*
Yellow is I for rx channel 1
Purple is Q for rx channel 1
Red is I for rx channel 2
Blue is Q for rx channel 2 The lower four coloured tracks show excluded frequencies. Note that the blanker is disabled and therefore the noise level is extremely high in the waterfall which means that a lot of strong signals are not excluded as they would be in normal operation. Under normal operation much less signal is present between thew pulses in the weak signal group.

The benefit of using large bandwidth for the blanker is clearly demonstrated by fig. 10.

The noise blanker is improved in linrad00-32. Further improvements are possible but to investigate what can be done I need test recordings from interference situations where the current blanker does not work well.

The demo on this page is showing SSB signals. CW signals are much easier to find. The optimum way of finding what frequencies to exclude is hard to figure out because circumstanses may differ dramatically depending on the frequency band.

Linrad still has only the mode for weak CW implemented. It is really going outside the intended usage to open the bandwidth to allow ssb reception but it works quite well and is a good test for the algorithms. In other modes there is no need for very large fft's on the entire spectrum so it will be possible to use more efficient methods for the noise blanker. Splitting the signal into more than two and making more independent transforms on these signals will make it possible to suppress pulse noise even better.

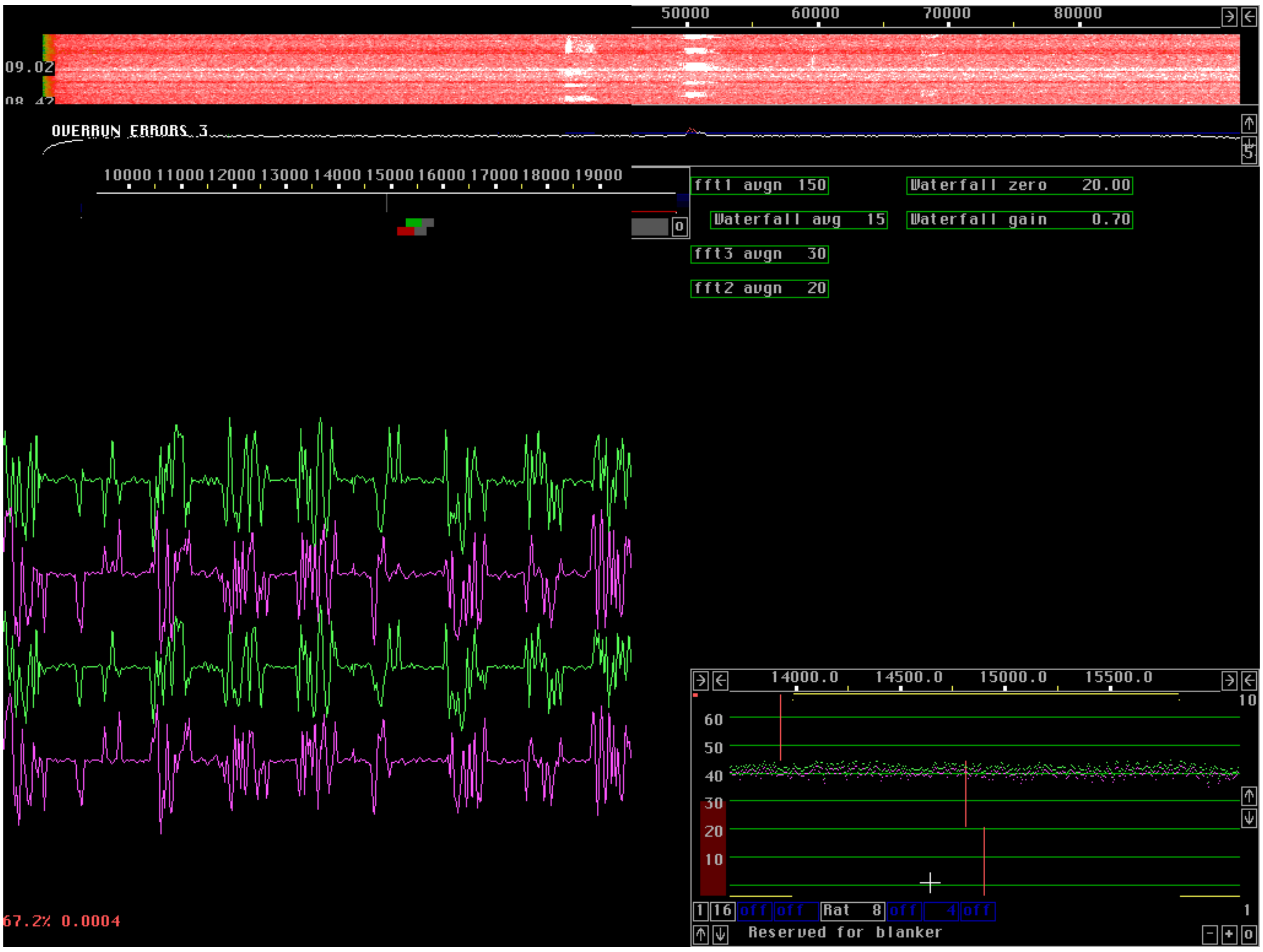


Fig 10. *Typical example of powerline noise in the baseband at 5kHz bandwidth. It is quite clear that a blanker operating in 5kHz bandwidth can not remove these pulses without loosing about 50% of the signal. Even 10kHz bandwidth is not enough to separate the individual pulses from each other. 20kHz bandwidth or more is required for a noise blanker to remove the powerline noise. With a conventional blanker each strong signal would make operation within +/- 10kHz from it very difficult.*

The Linrad noise blanker was originally developed for Pentium MMX 200 MHz. It was necessary to use 16 bit MMX instructions because of the limited CPU speed of the time. Today (2004) the fast computers allow the use of 32 bit floating point arithmetics. Starting with Linrad-01.26 the user can avoid the limitations with the 16 bit arithmetics by selecting fft version=0 for both the backwards fft1 and for the forward fft2. Using 32 bit floating point does not improve the reception of weak signals, but it makes the setup less critical and it makes the reception of strong signals better. Look here for details A comparison between 16 bit integer and 32 bit floating point arithmetics for the Linrad noise blanker

Using the spur removal function.

Linrad can apply notch filters to remove spurs. These notch filters are applied as phase locked oscillators, sine waves that are fitted in phase and amplitude to sinewaves present in the input. Only signals that have a very narrow bandwidth can be removed this way but the process runs in the frequency domain so a very large number of notches can be present simultaneously.

The spur cancellation routines have been present in Linrad since 2002: Spur cancellation. With Linrad-02-39 (Sept 30 2007) spur cancellation is automated. Click here: Automatic spur cancellation. for examples on how to use this feature.

Automatic spur cancellation in Linrad

Theory of operation

Linrad has a number of digital PLL's (phase locked loops) that can be locked to phase stable signals that are present in the input spectrum.

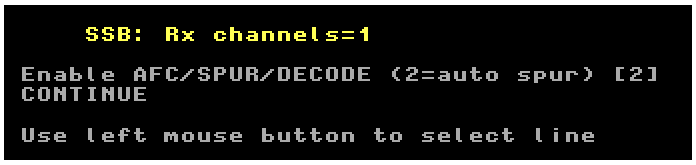
The general theory about PLL's is described by many others and there is no reason to go deeply into it here. Basically a digital LO (local oscillator) is set up with the same frequency as the offending spur. The LO is used in two mixers to produce I and Q (quadrature mixing) and Q is used to control the phase/frequency of the LO through an appropriate loop filter.

The LO of the PLL has ideally the same phase as the spur. The spur amplitude is the I signal. In the presence of noise there will be errors in the LO phase and the I signal, but by selecting a narrow loop filter and filtering I through a similar filter the noise can be suppressed.

Once the amplitude and phase of the spur is known a signal with the same amplitude and opposite phase can be generated and added to the wideband signal before it is further processed. This way the spur is cancelled without any effect on desired signals at the same frequency, provided that the desired signals have much larger bandwidth than the bandwidth of the loop filter.

Selecting parameters for spur removal.

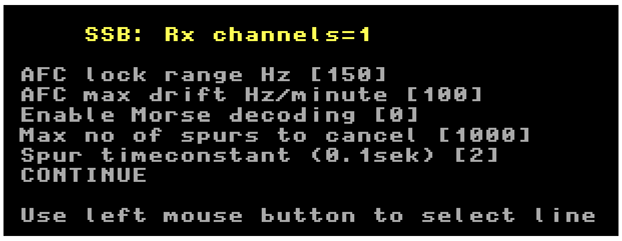
As a first step, select 2 for automatic spur removal as shown in figure 1.



```
     SSB:  Rx channels=1

Enable AFC/SPUR/DECODE (2=auto spur) [2]
CONTINUE

Use left mouse button to select line
```

Fig 1. *Enable automatic spur cancellation like this.*

The spur removal will operate on the second FFT if it is enabled, if not it will operate on the first FFT. The bin bandwidth of the FFT must be larger than the bandwidth of the spur one wants to remove. Too much bandwidth will degrade S/N in the phase locking process causing the spur removal to fail on weak spurs.
There are two parameters that affect the spur removal. The maximum number of spurs and the spur time constant. See figure 2.



```
     SSB:  Rx channels=1

AFC lock range Hz [150]
AFC max drift Hz/minute [100]
Enable Morse decoding [0]
Max no of spurs to cancel [1000]
Spur timeconstant (0.1sek) [2]
CONTINUE

Use left mouse button to select line
```

Fig 2. *Here maximum 1000 spurs can be removed and the bandwidth of each notch is about 5 Hz (the time=0.2 seconds).*

It should be obvious that aggressive settings for the spur removal, large bin bandwidth and short time constant will produce wide notches that will find and remove the carriers of stable CW transmissions. On the other hand, many real life spurs do not have the phase stability required for notch bandwidths below a few Hz.

A practical example of the automatic spur removal.

This file http://www.sm5bsz.com/linuxdsp/spur/file1/20mnoise1.raw (14 916 809 bytes) was recorded by Roger, W3SZ, on 20 meters. To use this file as input to Linrad in order to experiment with interference reduction, place the file in a directory **/demo** if you run under Linux or in a directory **c:\demo** if you run under Windows. **Note that this file contains incorrect calibration data.**

Figure 3 shows the Linrad screen for the 20mnoise1.raw file without any blanker or spur removal routine. Like all .raw files saved by Linrad, this file has a header with various pieces of information. This particular file was recorded on a system that contained calibration data from another WSE system so calibration was incorrect. The 20mnoise1.raw file therefore contains calibration data that does not match the recording and as you can see in figure 3 the noise floor is far from flat. The blankers operate well despite this, but the noise floor is not very flat.

In Linrad-02.41 and later there is a .raw to .wav file converter. With the 20mnoise1.raw file as input it produces the file http://www.sm5bsz.com/linuxdsp/spur/file1/20mnoise1.wav (13258796 bytes) which is a standard 16 bit wav file that can be used with Winrad and other SDR softwares. Figure 4 shows the Linrad screen processing the 20mnoise1.wav file without any interference fighting. and without any calibration files present in the Linrad directory.

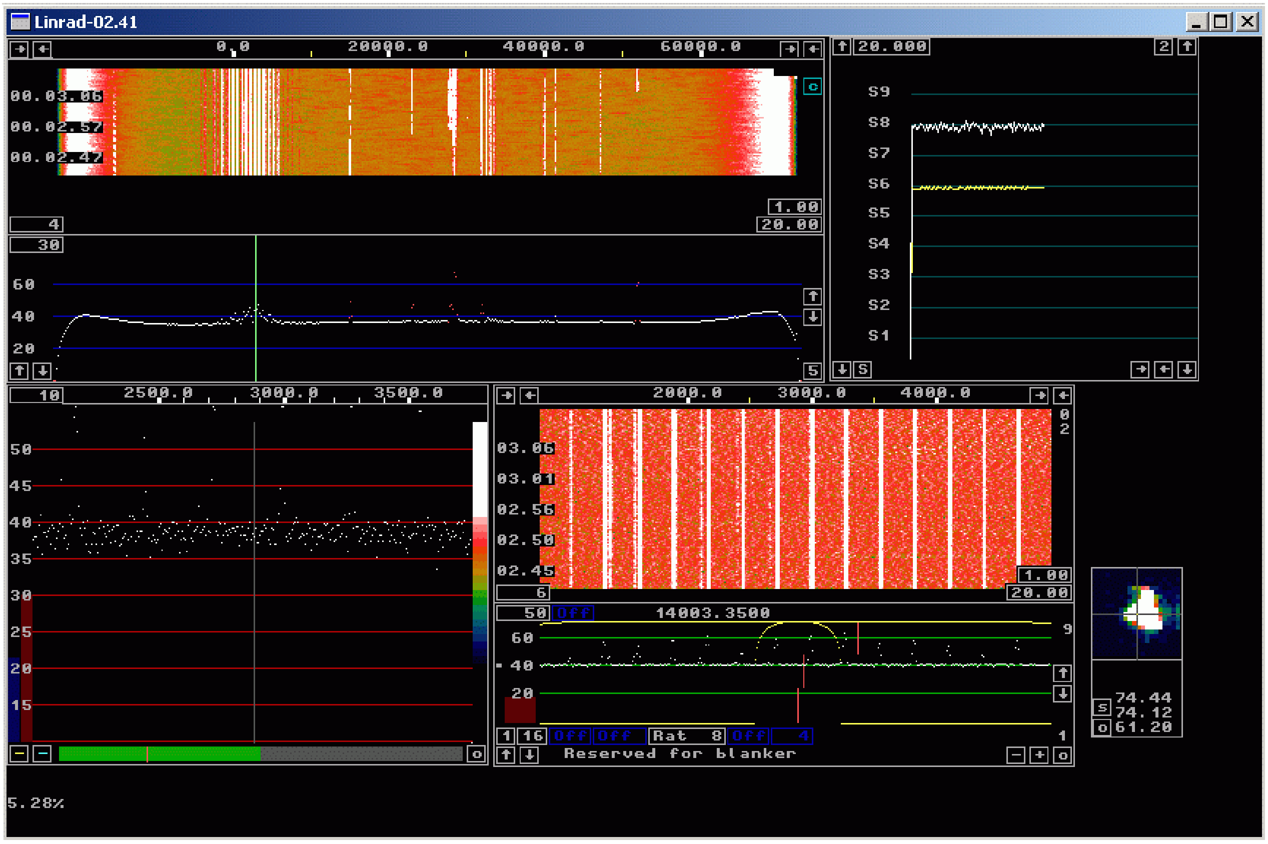


Fig 3.No blanker, no spur removal. Input is 20mnoise1.raw with incorrect calibration.

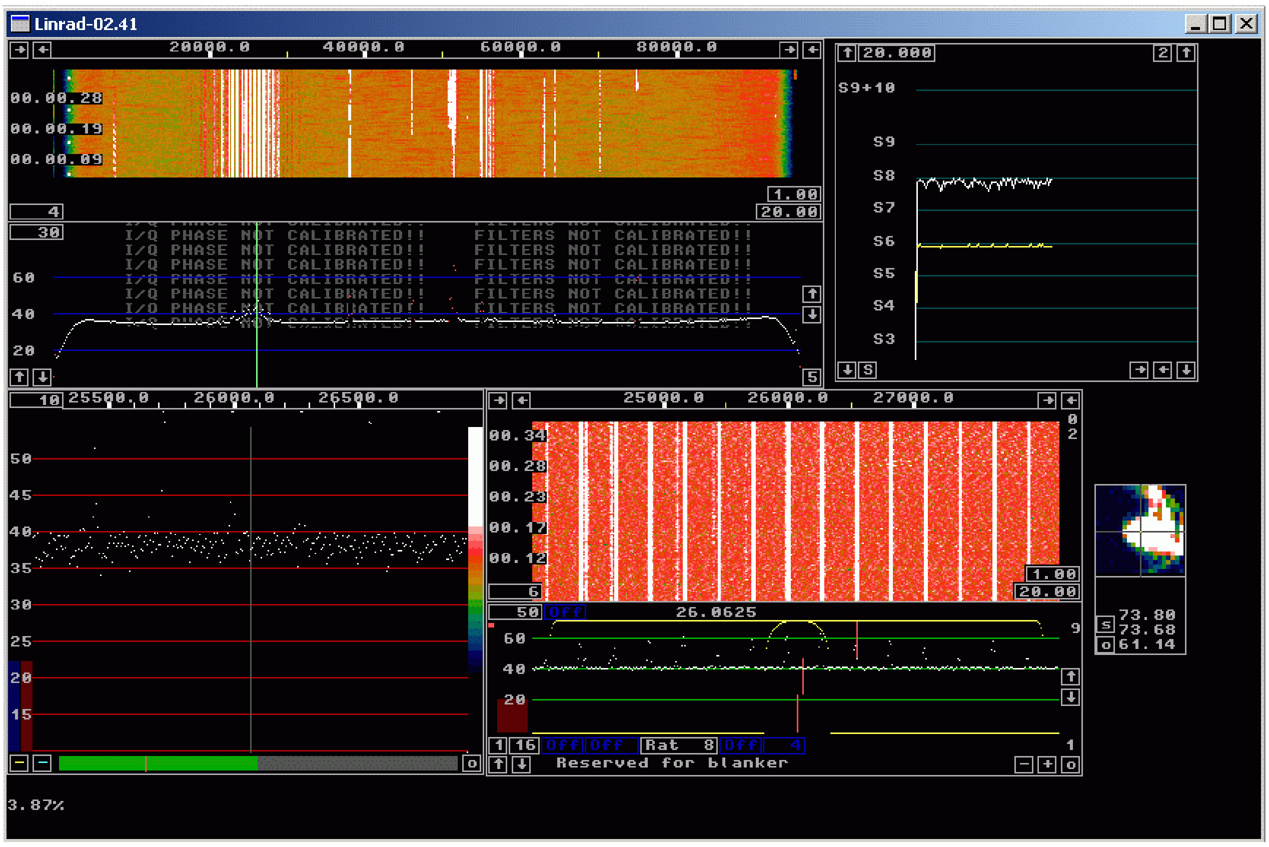


Fig 4.*No blanker, no spur removal. Input is 20mnoise1.wav without calibration.*

This file [http://www.sm5bsz.com/linuxdsp/spur/file1/20mcalpul.wav (7175168 bytes)](http://www.sm5bsz.com/linuxdsp/spur/file1/20mcalpul.wav) contains pulses recorded with the same WSE converters that were used about two months earlier to record the 20mnoise1.raw file. When running in normal receive mode with these pulses as input, the screen looks as figure 5.

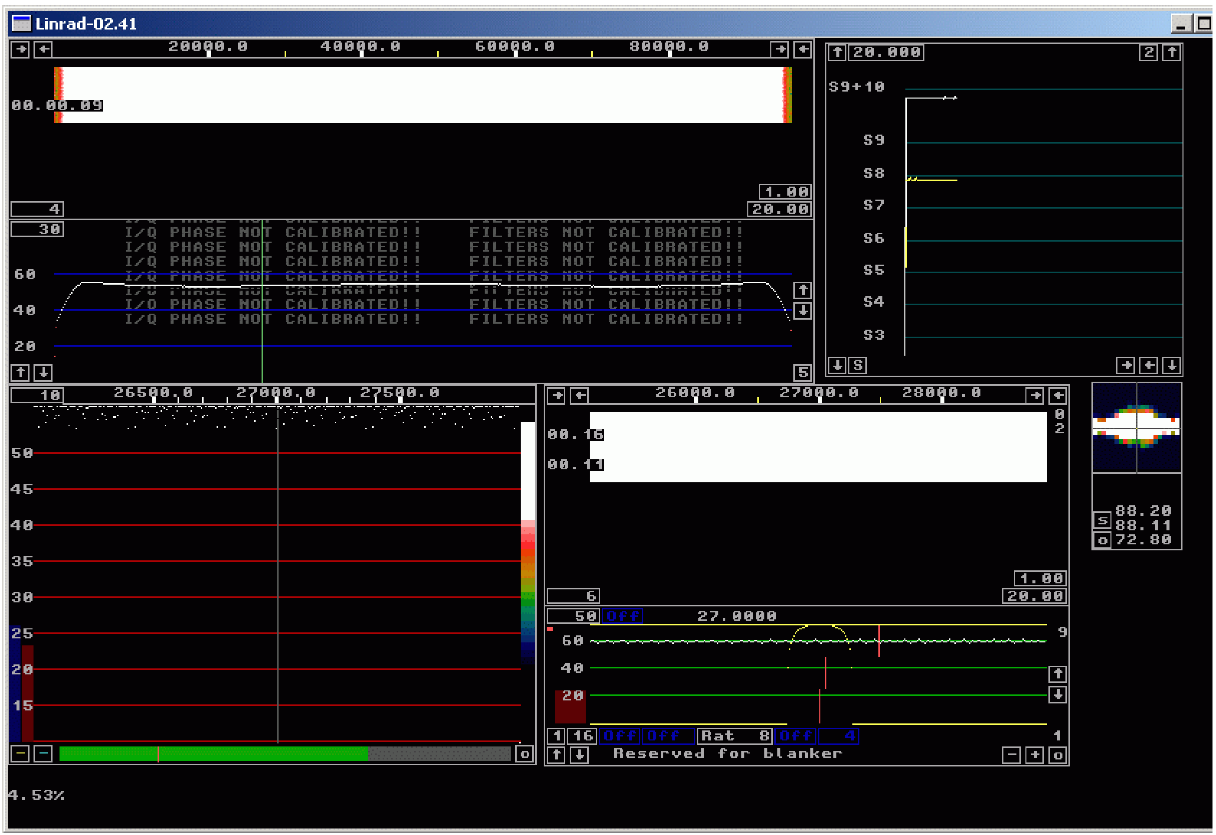


Fig 5.*No blanker, no spur removal. Input is 20mcalpul.wav without calibration.*

By pressing **X**, then **C** when the 20mcalpul.wav file is selected for input (repeat recording endlessly) one comes to the calibration menu. Press **B** to start calibrating the frequency response. The procedure is described here: Calibration of filter responses in Linrad

The calibration procedure will generate a file dsp_wcw_corr (in weak CW mode) and place it in the Linrad directory. Subsequent runs with the the .wav file as input will then use this file for calibration. Runs with the .raw file will be unaffected, they will continue to use the incorrect calibration information from the .raw file header.

Figure 6 shows the screen when the 20mnoise1.wav file is processed with the correct calibration file dsp_wcw_corr (3152 bytes) present in the Linrad directory. The calibration file can be produced from the 20mcalpul.wav file or downloaded here. As can be seen in figure 6, the noise floor is very flat. In case you make your own calibration file you can try different desired shapes for the filter edges and find out how blanker operation is affected.

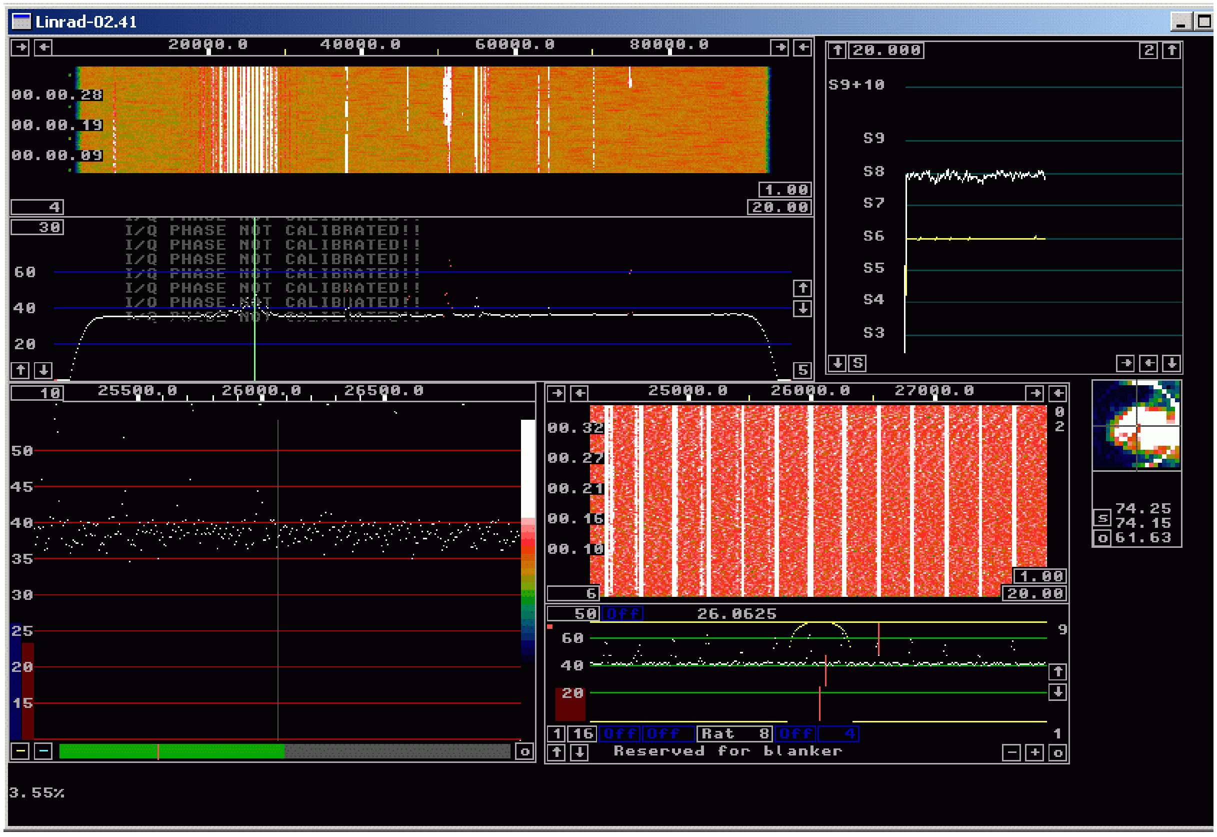


Fig 6.No blanker, no spur removal. Input is 20mnoisel.wav and the calibration file is produced from the 20mcalpul.wav file.

In figure 7, the noise blanker is enabled and the noise floor is about 15 dB lower. In figure 8, both the noise blanker and the automatic spur removal is running. The settings for the spur removal is aggressive, the time constant is only 0.2 seconds. Figure 9 shows the spur at 14.004644 MHz at high resolution. It should be pretty obvious that this spur cannot be removed with a very narrow notch and that is the reason for using aggressive notch parameters that could destroy CW signals. SSB should not be affected though.

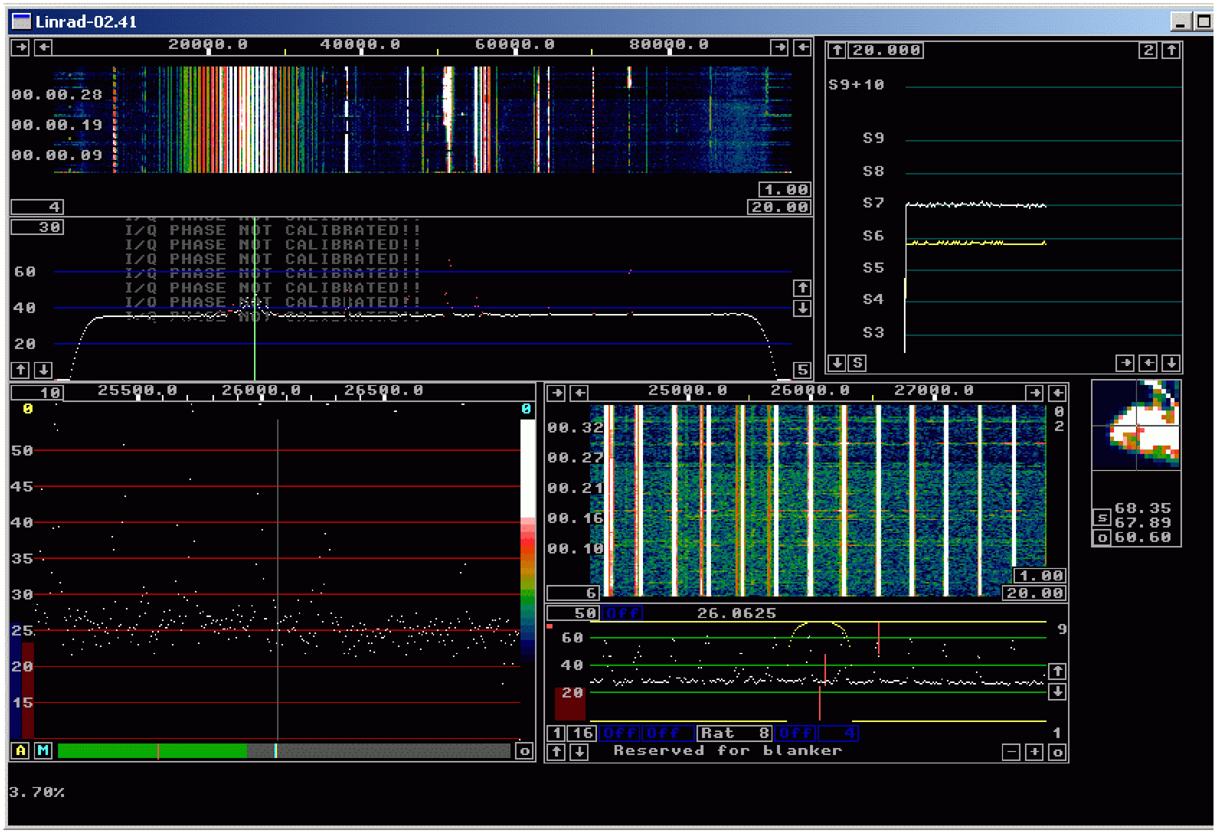


Fig 7. *Blanker enabled, but no spur removal.*

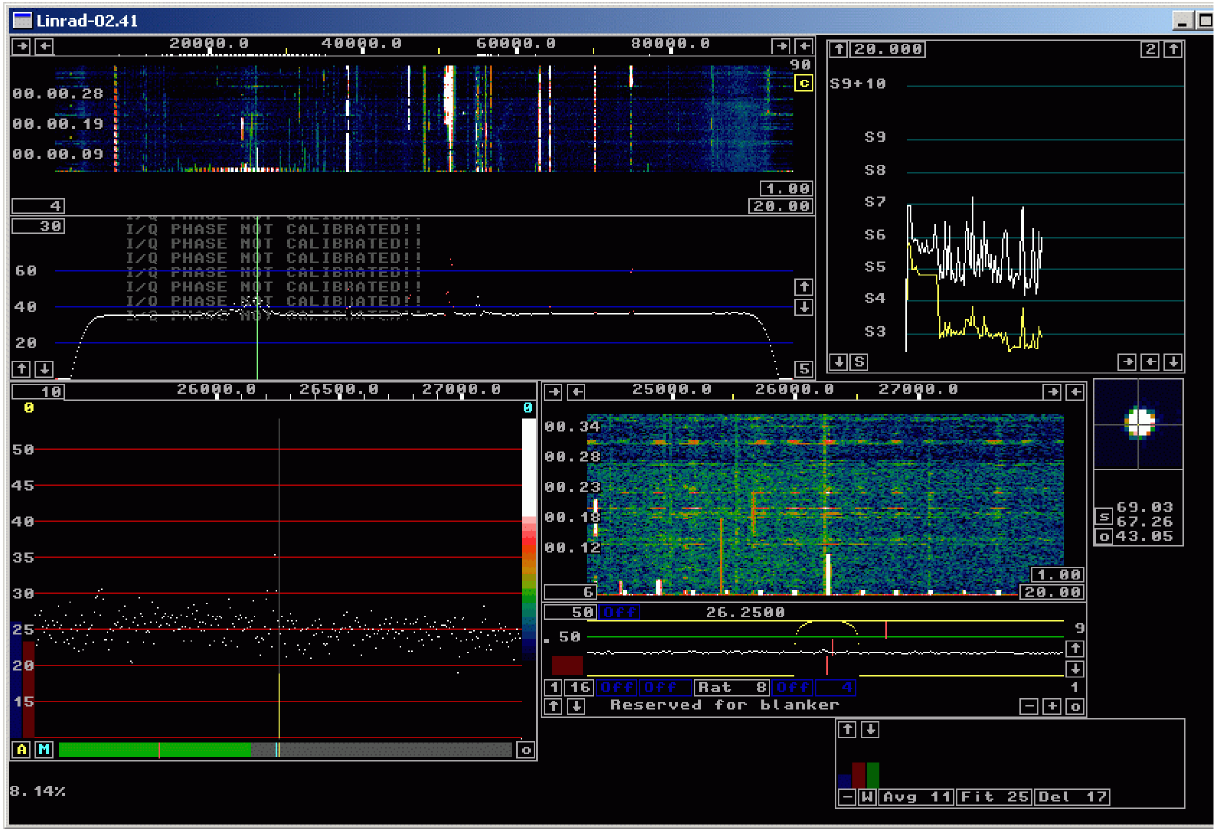


Fig 8. *Both blanker and spur removal running.*

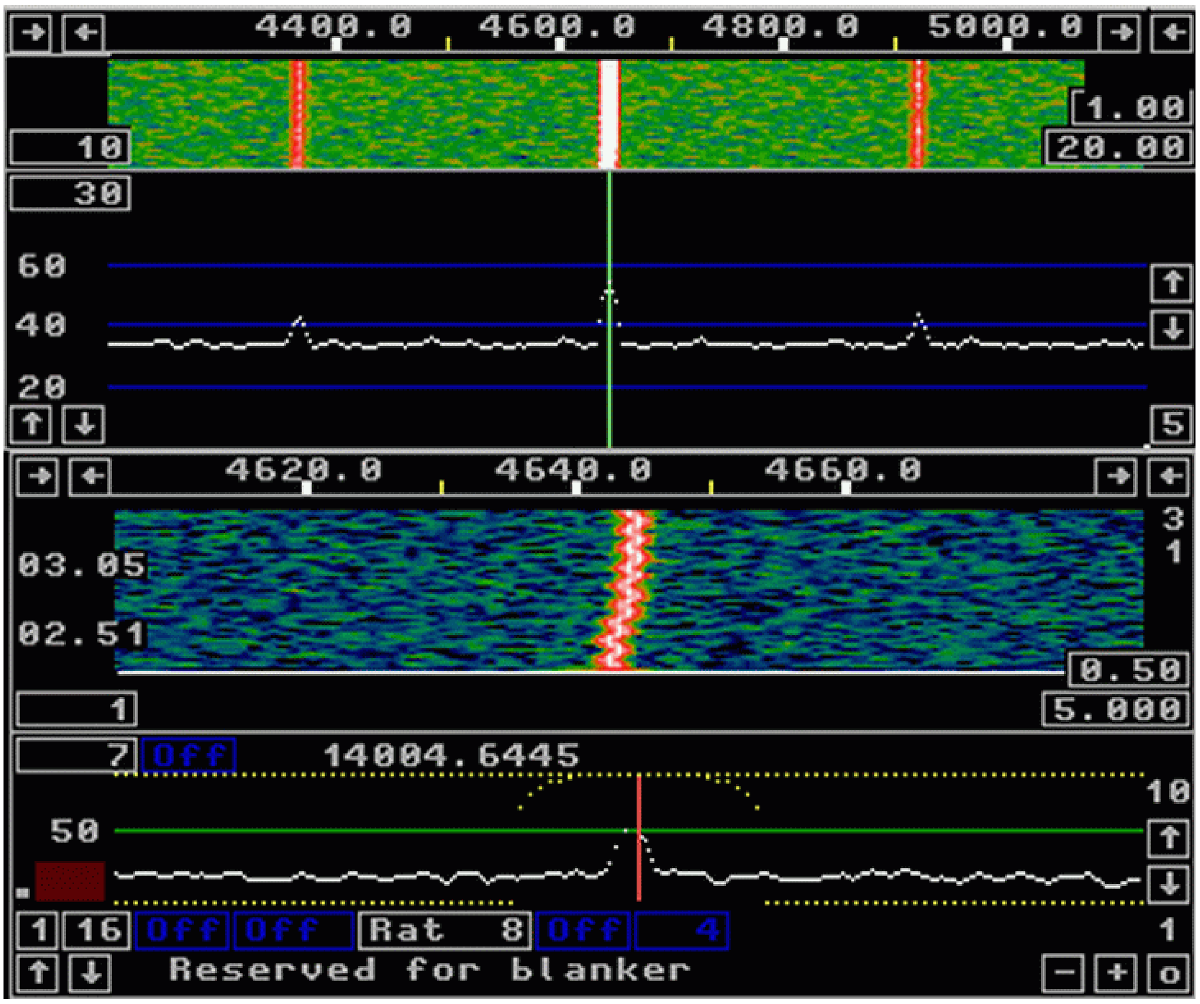
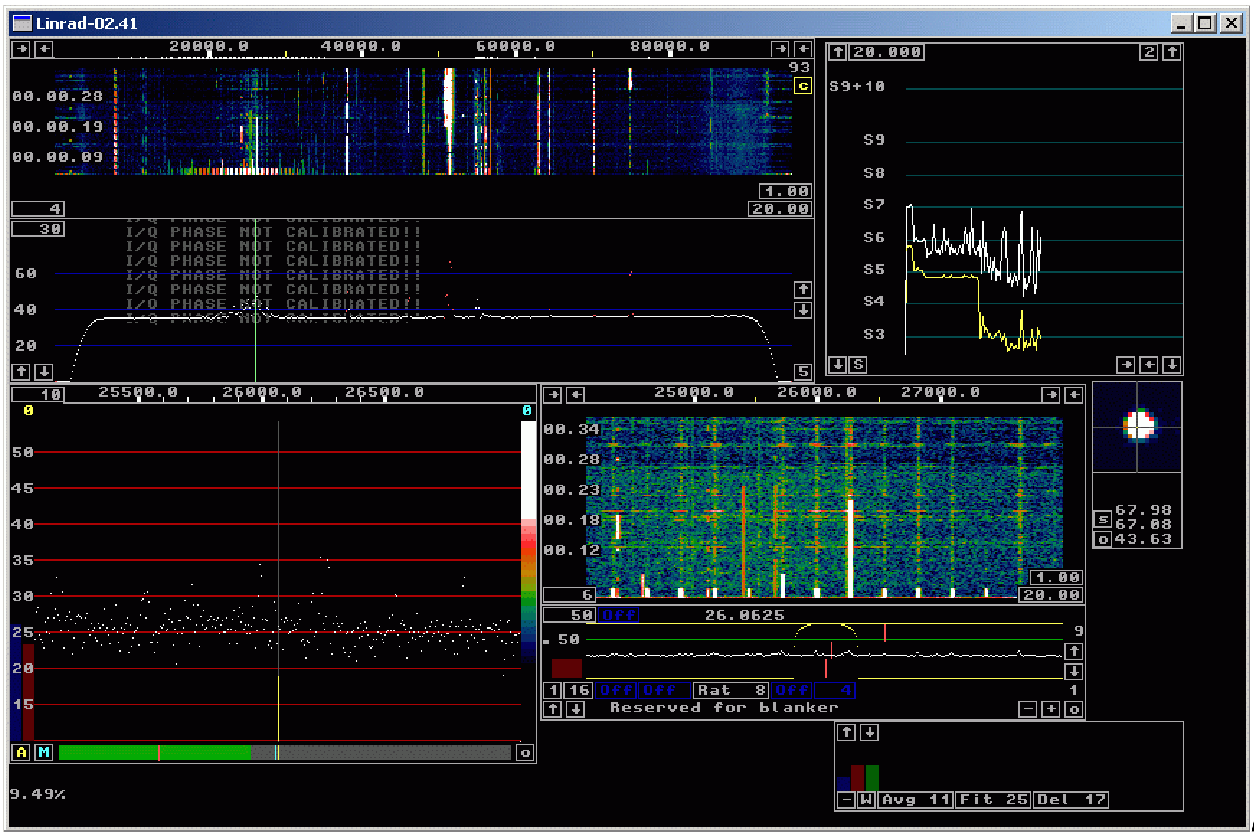


Fig 9. *This spur needs a rather wide notch!*



Fi

g 10. *Here the time constant for the spur removal is 0.8 seconds.*

A signal processing challenge.
The file 20mnoise1.wav that can be downloaded from this page contains interference pulses as well as weak and strong CW signals. These signals are of particular interest:

Call	Freq	Real Freq	Peak level
unkn	51.44kHz	14028.44kHz	80 dB
W5ZR	50.91kHz	14027.91kHz	84 dB
OK1HB	48.74kHz	14025.74kHz	44 dB

The signal of OK1HB is weak and does not last long. It is subjected to QSB down to the noise floor which is at 26 dB RMS in a bandwidth of 20 Hz, the optimum bandwidth for this signal. The maximum S/N is 18dB in 20 Hz, corresponding to -3dB in SSB bandwidth. The signal of OK1HB is well visible in all main waterfall graphs above where the blanker is enabled despite the fact that the waterfall is 96 kHz wide in only 512 pixels.

Download the file 20mnoise1.wav (13258796 bytes) Run it with your favourite SDR software, compare the visibility in the waterfall with a similar total width as abowe (192 kHz = 1024 pixels) Listen try to find out which station OK1HB is calling.

The normal loudspeaker output from Linrad ok1hb.wav (751362 bytes) allows copying of both transmit periods despite the strong level of the two nearby stations during the second transmission. In the processing mode coh2, where Linrad uses the carrier of the keyed signal to detect in-phase and out-of-phase components within the 20 Hz passband, copy is a little easier. The noise is split equally between I and Q but the signal is present almost exclusively in I. Listen to this file ok1hb_coh2.wav (1513108 bytes) with stereo head-phones. In situations where the carrier is not stable enough, too weak or if a long time delay is not acceptable, the coh2 mode will fail and put the signal in Q occasionally. With stereo head-phones, one will hear it anyway. The signal will have a fixed phase relation between both ears and seemingly come from one direction while the noise will have random phases and seem to come from all directions. In the coh2 file, S/N is nearly 3 dB better in the I channel as compared to the normal loudspeaker output.

When coherent processing is sucessful, selecting the coh3 mode will send the I signal to both ears. The signal is sent only when the signal is in phase, but not when it is 180 degrees out of phase. With noise only, the RMS power is reduced by another 3 dB this way, but the noise itself becomes a keyed signal. Here is the sequence in coh3 mode: ok1hb_coh3.wav (745828 bytes)

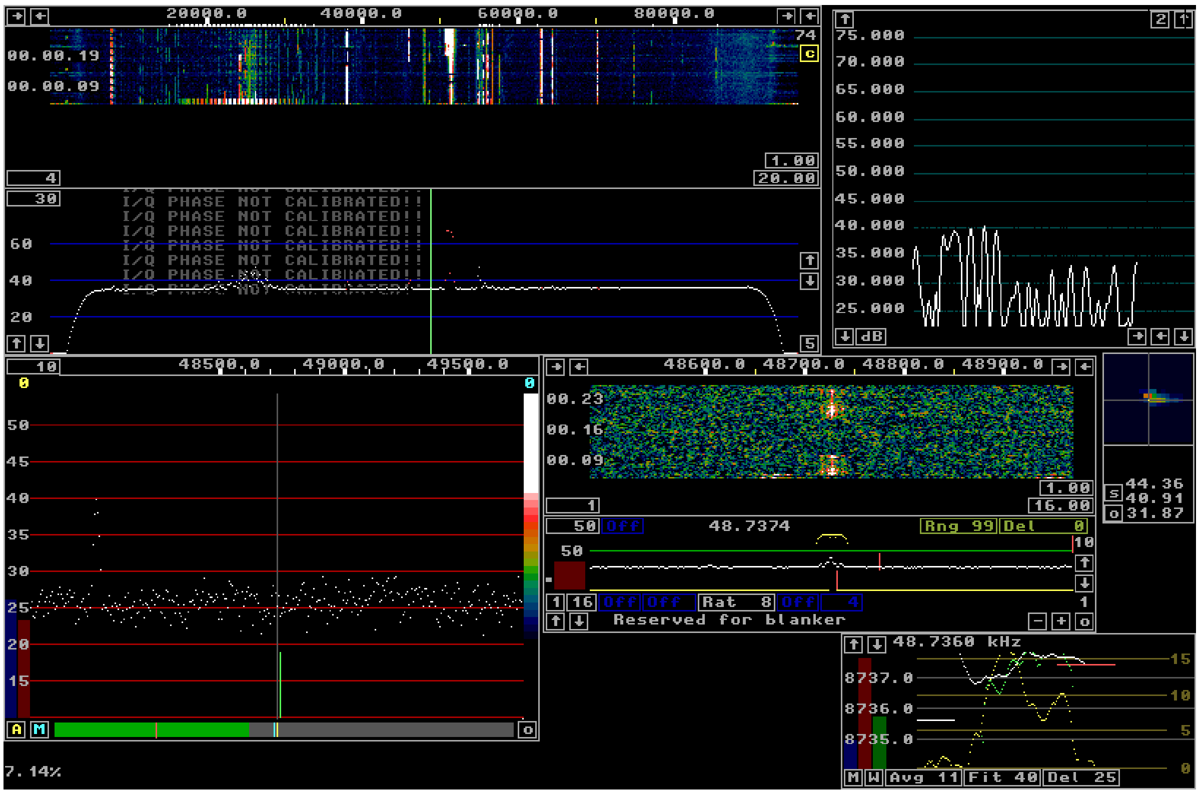


Fig 11. *OK1HB on 48.737 kHz. The final 'B' of the second transmission is visible in the S-meter graph.*

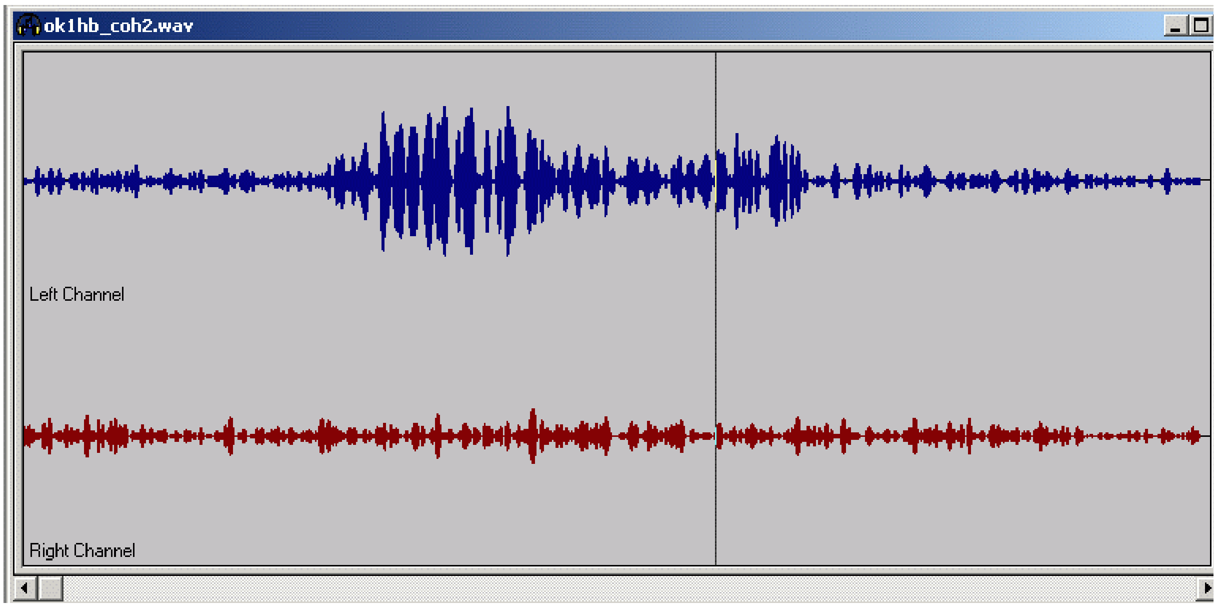


Fig 12. *The middle part of ok1hb_coh2.wav as seen with a wave file editor. The Linrad AFC was sucessful in finding the correct frequency and phase. Very little of the signal is in Q.*

Second operator over network

If two operators want to receive the signals from a common antenna they may use the same radio hardware and add a second computer in which a Linrad slave will process the digital data sent to it over a network from the Linrad master.

The LINRAD NETWORK can be used for raw data or partly processed data. See **Appendix 4**

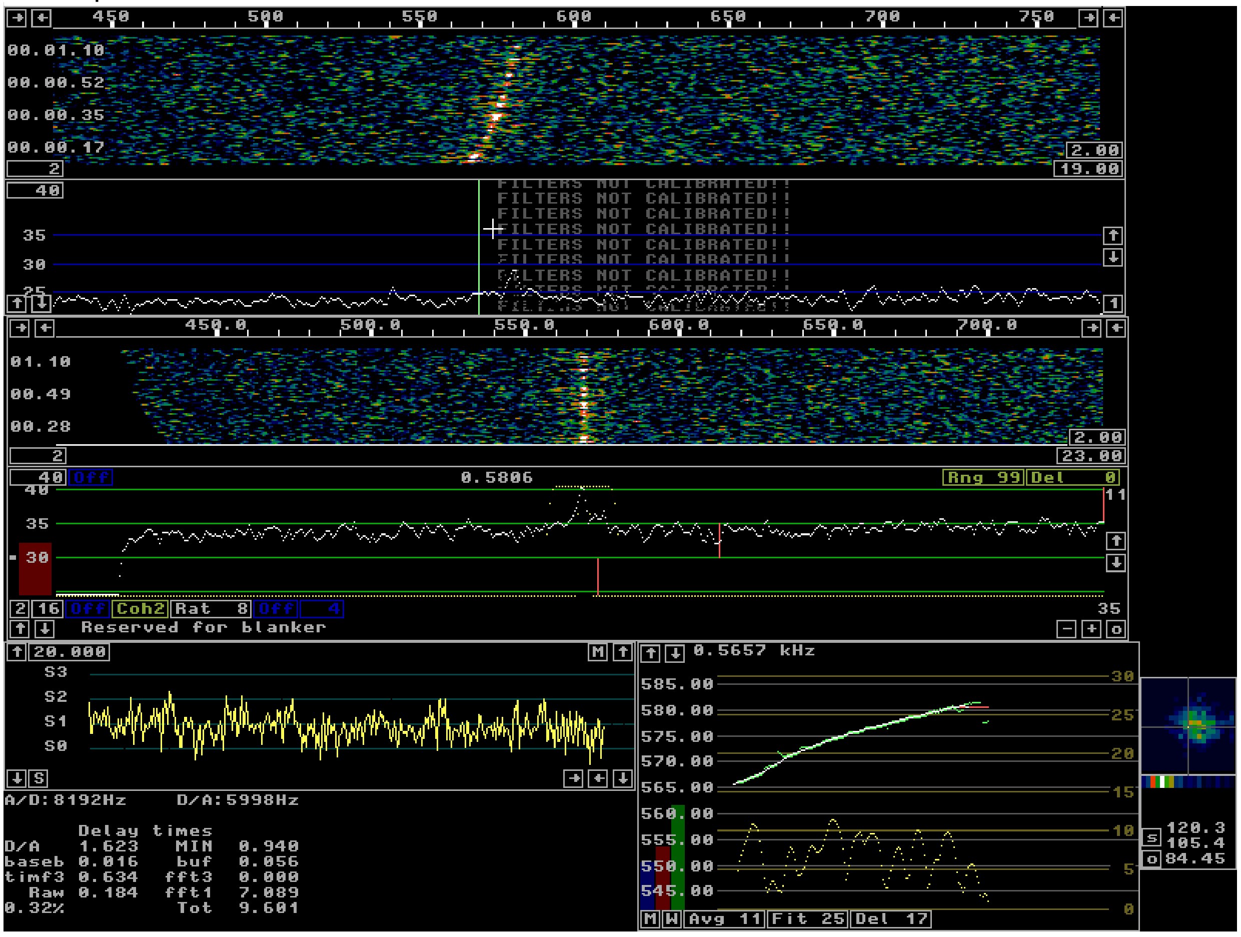
Linrad for single polarisation system using a conventional SSB receiver.
(Oct 25 2006)

Spectrum display and audio filter with AFC

When using a normal SSB receiver at VHF/UHF frequencies the builtin noise blanker of the SSB radio usually works well. Problems with strong signals within the passband used by the noise blanker are not as much of a problem as on more crowded bands. The noise blanker within linrad can be used to improve in case the analog blanker fails. To use the linrad blanker, the system has to be calibrated by use of a pulse generator.

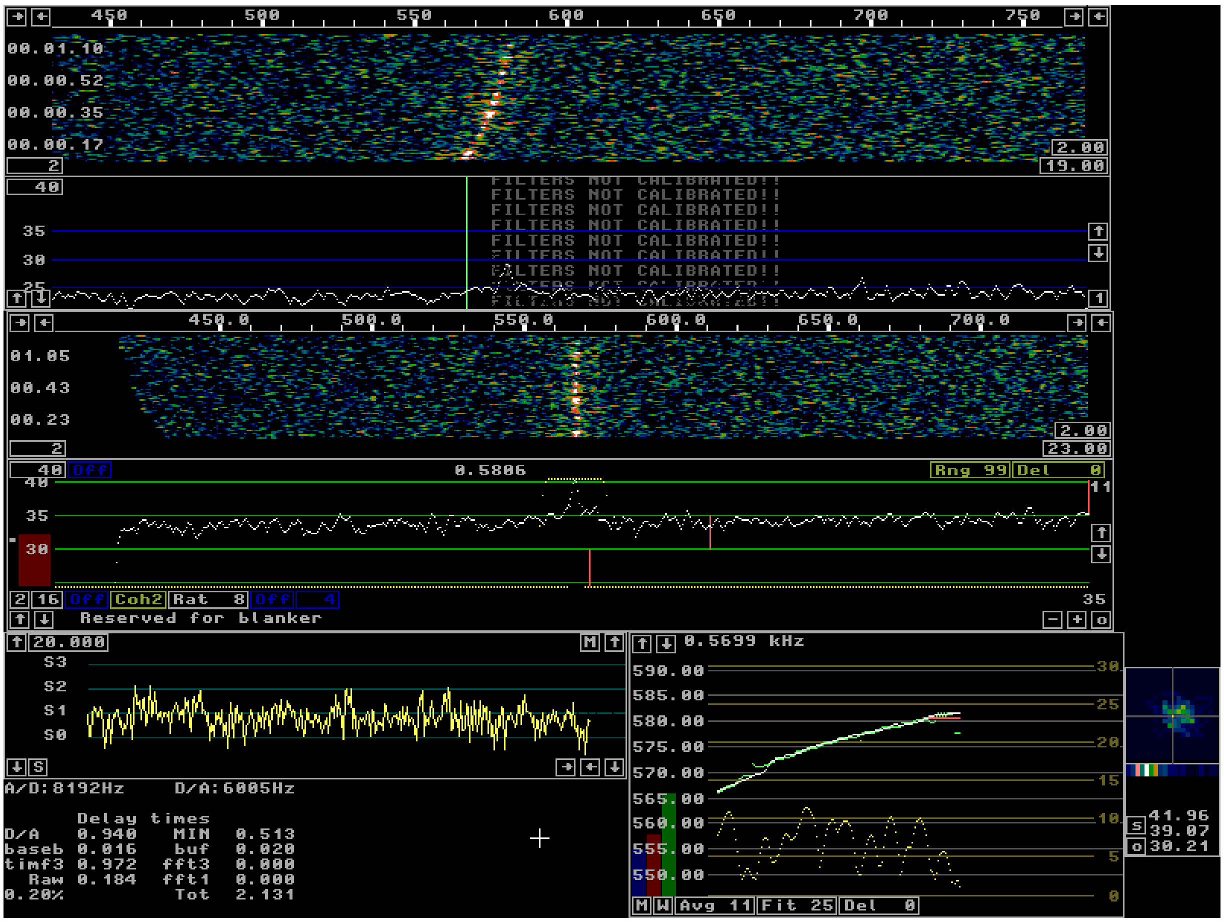
In normal operation with a good receiver having a reasonably flat frequency response calibration is not required As an illustration to the practical use of linrad for a single polarised antenna with a conventional SSB receiver fig. 1 shows the reception of the UNKN422.WAV file.

The parameters are set for high performance in copying really difficult signals. The time delay is about 8.5 seconds from input to audio in the headphones, but nothing prevents the transmitter to start while the last part of the received signal is still being processed. The delay is primarily caused by the AFC which is set to interpolate and compute the optimum frequency based on the spectrum 7 seconds back in time as well as 7 seconds into the future. Figure 2 shows the screen when the AFC is set to extrapolate by fitting a straight line to the spectrum 14 seconds back in time



```
A/D:8192Hz      D/A:5998Hz

         Delay times
D/A    1.623    MIN   0.940
baseb  0.016    buf   0.056
timf3  0.634   fft3   0.000
  Raw  0.184   fft1   7.089
0.32%           Tot   9.601
```

Fig. 1. *A CW signal in SSB bandwidth received by Linrad on an uncalibrated system. The time delay is about 8.5 seconds, when listening to a live signal from the soundcard the delay caused by the D/A buffer can be set very small.*



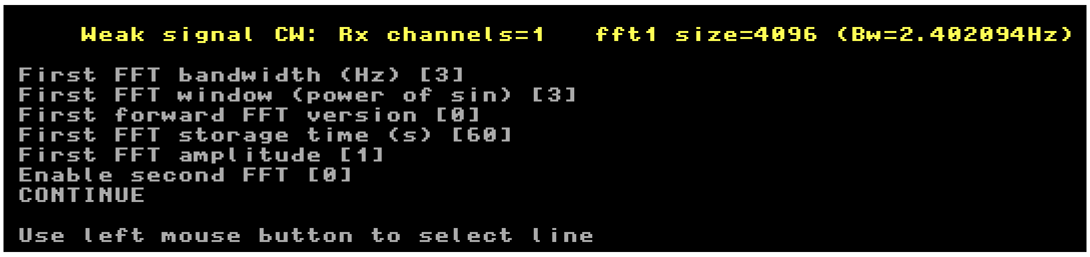
```
A/D:8192Hz     D/A:6005Hz

       Delay times
D/A    0.940    MIN   0.513
baseb  0.016    buf   0.020
timf3  0.972   fft3   0.000
  Raw  0.184   fft1   0.000
0.20%           Tot   2.131
```

Fig. 2. Here the delay is reduced to less than two seconds by allowing the AFC to use an extrapolated value

In figures 1 and 2 the main waterfall and the baseband waterfall are set to have the same parameters so one can directly see how the AFC affects the spectrum. It is obvious that the quality of this signal is good enough to allow operation with a small time delay. As can be seen from the AFC graph the frequency error is about 1 Hz for a while when extrapolation is used. This is the error caused by extrapolating across the first region of no signal in the first fadeout. When the AFC uses interpolation, the error is much smaller. Having to open the baseband filter by 2 Hz to accomodate for some frequency error might be a good price to pay for the much shorter processing delay.

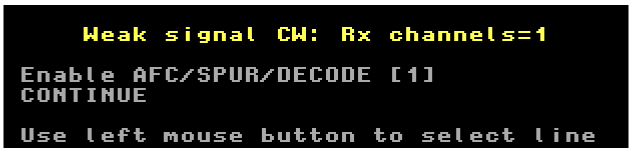
Setup parameters

The figures 3 to 5 below show the parameters that were used to produce figures 1 and 2.



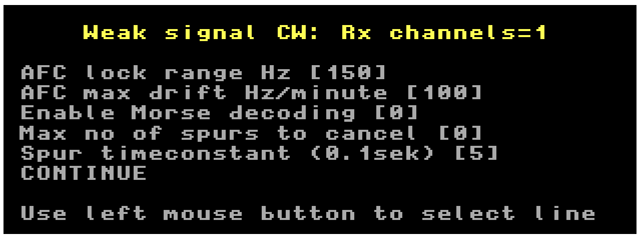
```
    Weak signal CW: Rx channels=1    fft1 size=4096 (Bw=2.402094Hz)

 First FFT bandwidth (Hz) [3]
 First FFT window (power of sin) [3]
 First forward FFT version [0]
 First FFT storage time (s) [60]
 First FFT amplitude [1]
 Enable second FFT [0]
 CONTINUE

 Use left mouse button to select line
```

Fig. 3. Parameters for the first fft. Note that the low value of First FFT amplitude is because the input is a 8 bit .WAV file with a high signal level. Since the noiseblanker is not needed the second fft is deselected.



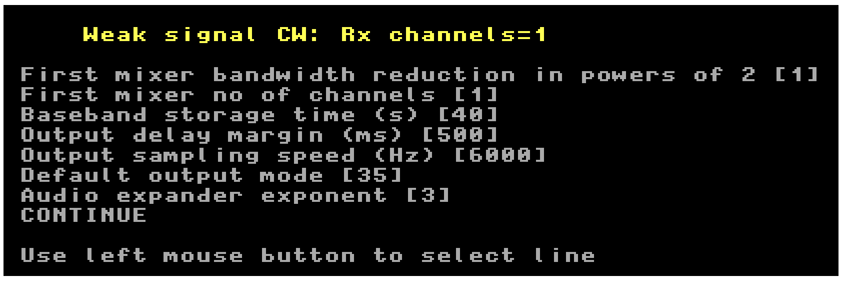
```
    Weak signal CW: Rx channels=1

 Enable AFC/SPUR/DECODE [1]
 CONTINUE

 Use left mouse button to select line
```

Fig. 4. *AFC is enabled.*



```
     Weak signal CW: Rx channels=1

AFC lock range Hz [150]
AFC max drift Hz/minute [100]
Enable Morse decoding [0]
Max no of spurs to cancel [0]
Spur timeconstant (0.1sek) [5]
CONTINUE

Use left mouse button to select line
```

Fig. 5. *These are the default AFC parameters.*



```
     Weak signal CW: Rx channels=1

First mixer bandwidth reduction in powers of 2 [1]
First mixer no of channels [1]
Baseband storage time (s) [40]
Output delay margin (ms) [500]
Output sampling speed (Hz) [6000]
Default output mode [35]
Audio expander exponent [3]
CONTINUE

Use left mouse button to select line
```

Fig. 6. *The unkn422.wav file is recorded at 8192 Hz in real format (one data value per sample). The conversion to complex format (Hilbert space) represents the same data at a sampling speed of 4096 Hz (two values, I and Q, for each sample) The baseband sampling speed is 2048 Hz. The default output mode is coherent processing. It is necessary to use stereo head-phones when listening because the decoding is done in the brains of the operator and the phase information between the two ears will help a little.*

Phase stability of the EME signal

With the parameters of figure 1 the waveforms produced by the coherent processing are shown in figure 7. This is an output of the experimental routines that some day may become the CW decoding algorithms of Linrad.

The first level "Raw" shows amplitude and phase of the signal that passes the baseband filter with a bandwidth of about 20Hz. The second level, also "Raw" shows the same signal in complex format I and Q. The bottom level, "Carr" shows the signal that passes through a filter that is about 2.5 Hz wide. (The Coh parameter 8 times narrower than 20Hz) It is obvious from the carrier track that the phase is stable over several seconds during the initial QSB peak. The signal sent to the earphones, "Coh1" is obtained by shifting the "Raw" signal by the phase angle determined from the "Carr" signal. It is displayed both as phase and amplitude. In Coh2 mode, the complex signal is sent to the ear-phones after being shifted from zero frequency to the frequency selected with the BFO control. I to one ear and Q to the other ear.

When the Coh1 mode is selected, the "Raw" signal is sent to one ear and the "Carr" signal to the other while the in-phase signal, the green curve of the fourth level is sent to both ears when the Coh3 mode is selected. Note that the relative phase between the two ears is reversed if you select one channel only for the output and that this will degrade performance if you use loudspeakers rather than head-phones.

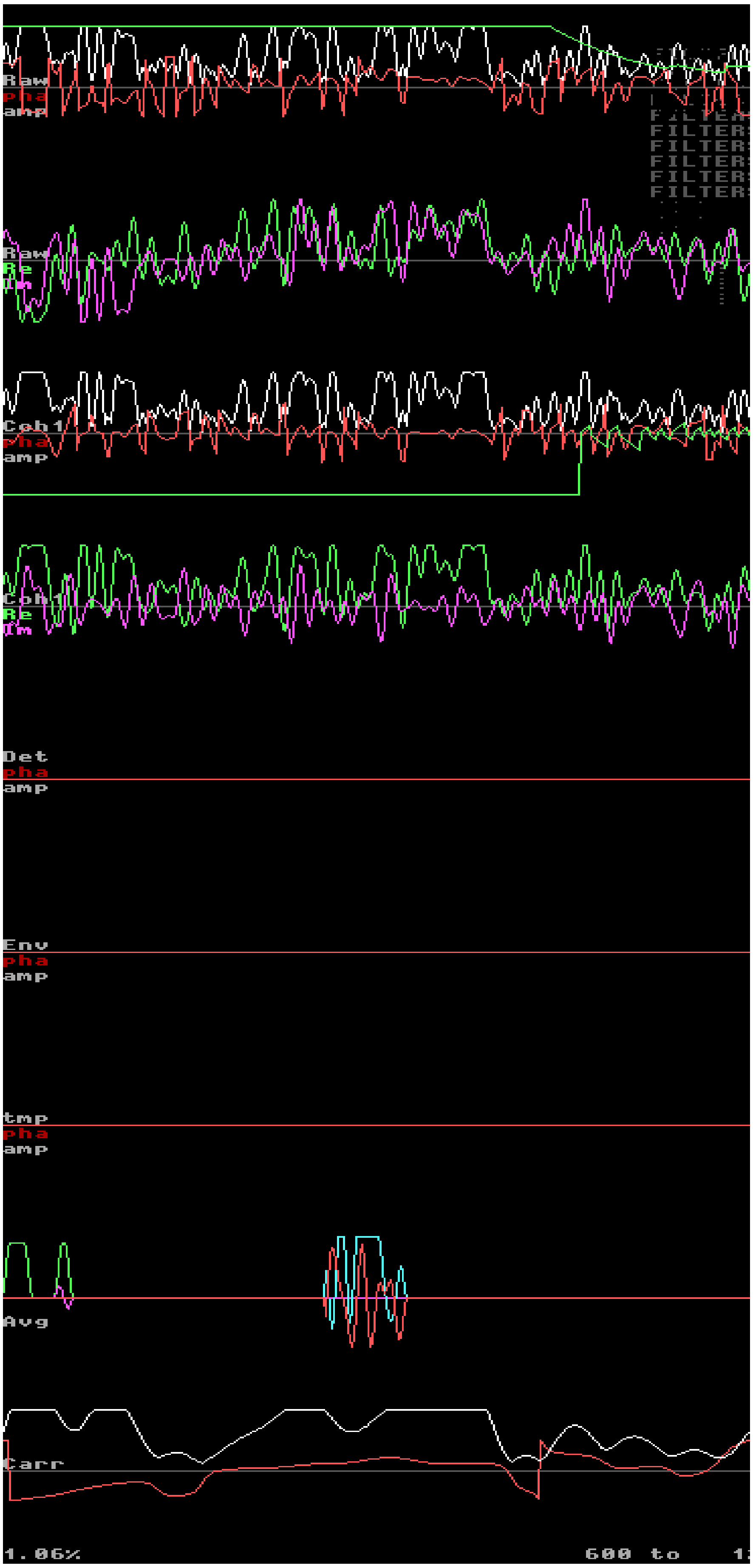
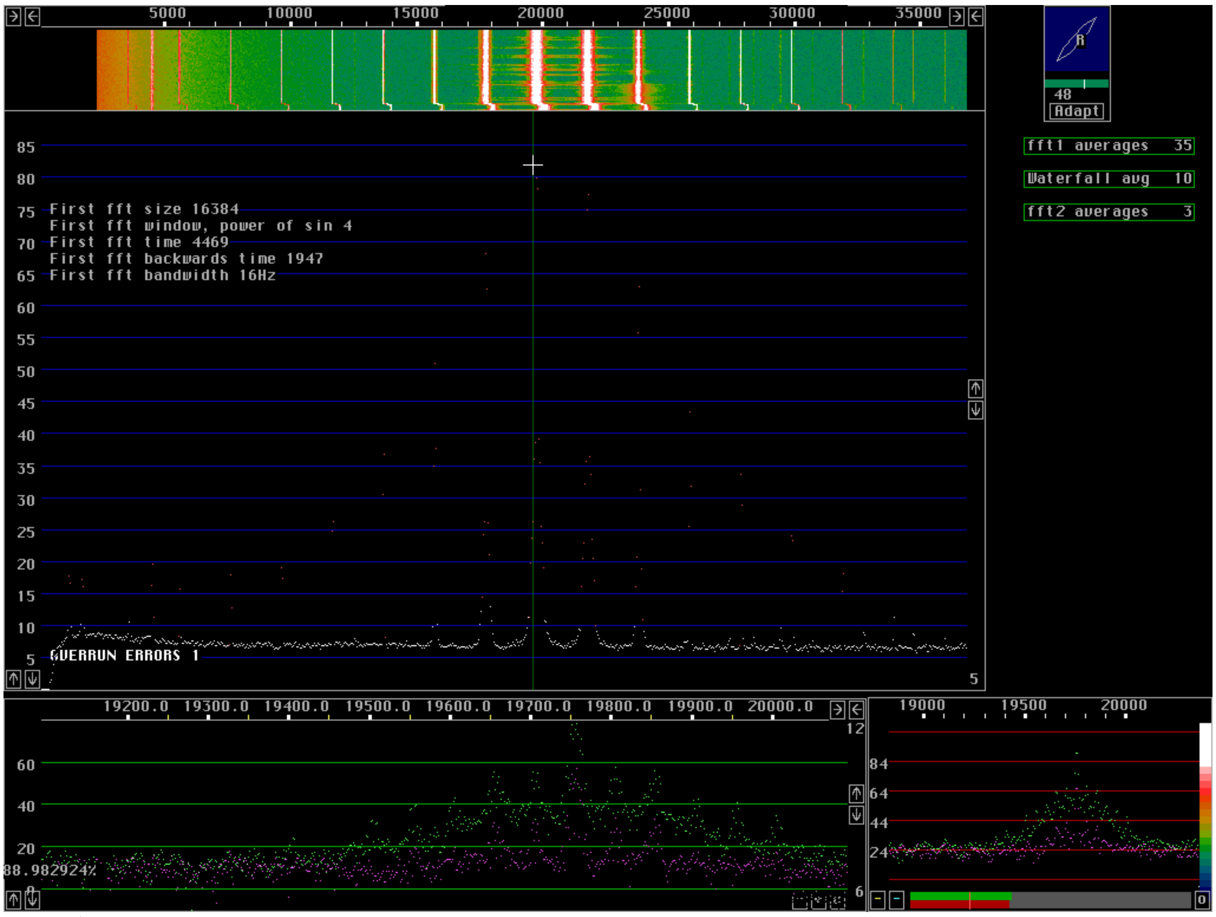


Fig. 7. *The signals passing the baseband filter when UNKN422.WAV is processed like in figure 1. This is the very beginning of the recording.*

Linux dsp radio, setting up the first fft.

The information on this page is valid only if the second fft is enabled

The first fft setup affects receiver dynamic range
This page shows some screen dumps with the signal from a AM modulated signal generator (old style, vacuum tubes). Fig 1 shows what the signal really looks like. The processing is with 16384 points for both the first and the second fft, with a sin to power 4 window for the first fft and a sin squared window for the second.
The high resolution graph shows the second fft with one pixel per fft bin separated into the polarisations specified by the polarisation graph. Nearly linear polarisation at an angle of 48 degrees, slightly elliptic right hand. The signal from the signal generator is fed to both channels of the receiver with slightly different cable lengths, that is why the polarisation is slightly elliptic.
The waterfall graph of fig. 1 does not show more spurs than those visible in the first fft graph so they originate in the analog hardware. A synthetiser of modest purity was used as a local oscillator.

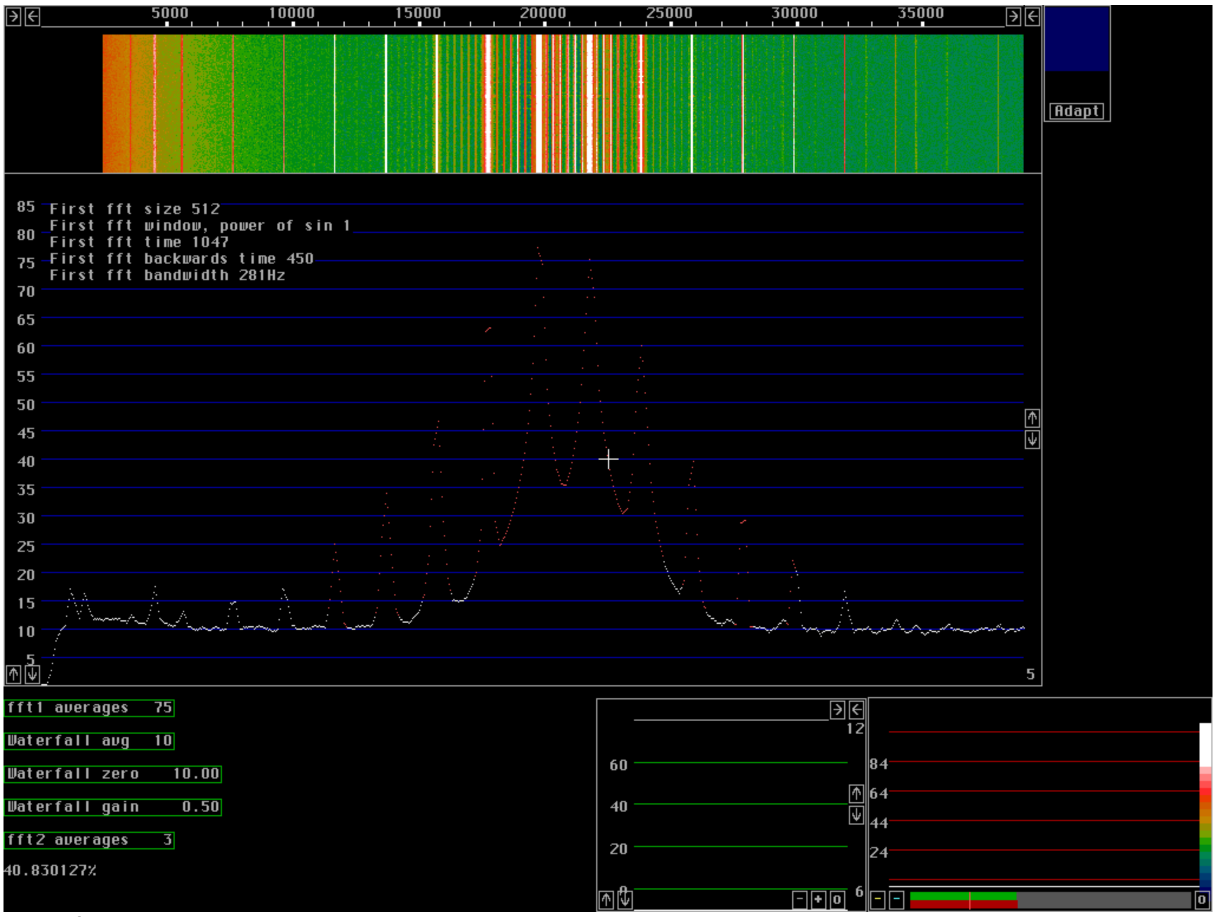


```
First fft size 16384
First fft window, power of sin 4
First fft time 4469
First fft backwards time 1947
First fft bandwidth 16Hz
OVERRUN ERRORS 1
fft1 averages    35
Waterfall avg    10
fft2 averages     3
48
Adapt
88.982924%
```

Fig 1.: *Overkill processing parameters used to demonstrate the signal used for screen dumps on this page. The high resolution graph (red) shows that the resolution is far better than indicated by the waterfall graph. The signal generator is modulated by 50Hz hum. Not as bad as the graph indicates, the selective limiter has attenuated the carrier but not the sidebands. The baseband graph, green (dsp00-09) is the third fft operating at a sampling speed of 6000Hz with negliable processing time. Signals of better quality are needed to show what it is useful for!*

Figure 2 shows what happens when inadequate processing is performed by the first fft. The reason for the spurs is that sucessive back transforms do not fit to each other. The second fft runs with a much larger transform size, 8192 points in fig 2 as in all other screen dumps

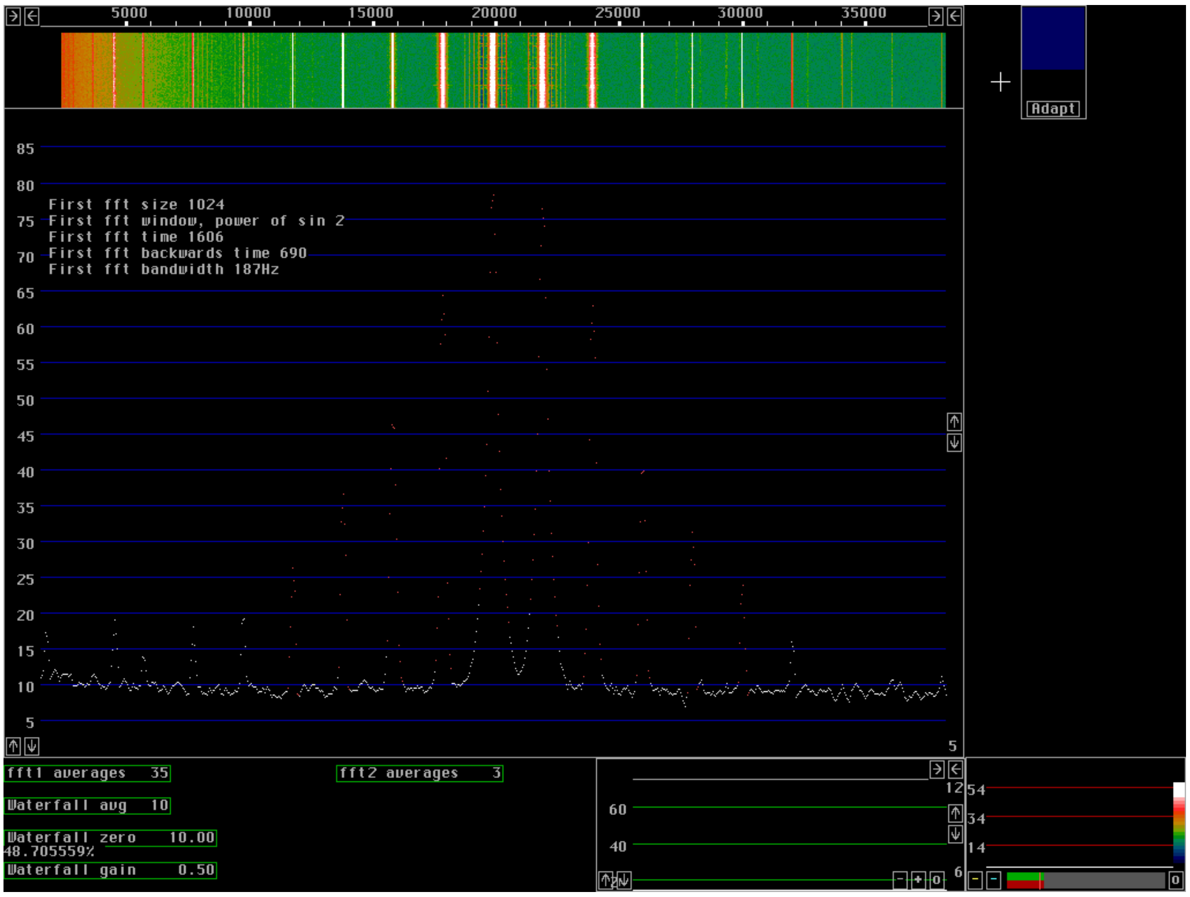
except fig.1. The mismatch between transforms create transients that form the sidebands clearly visible in fig 2.
In real life it is not as bad as it looks. A signal 70dB above the noise floor will typically be a few kHz wide anyway so some extra interference close to it might not make much of a difference.



```
First fft size 512
First fft window, power of sin 1
First fft time 1047
First fft backwards time 450
First fft bandwidth 281Hz

fft1 averages     75
Waterfall avg     10
Waterfall zero    10.00
Waterfall gain    0.50
fft2 averages     3
40.830127%
```

Fig 2. *Inadequate processing of the first fft. Few points and low order window make the strong signals surrounded by spurs that might hide a desired weak signal.*

There are two ways to get rid of the sidebands around strong signals.
Just by increasing the size of the first fft one can make transients occur less frequently, with less total energy in the sidebands as a consequence.
By use of a better window (see appendix 2) one can remove the transients inherent in the transform itself that arises from the matching of the ends of each transform. This way energy belonging to the strong sine wave will not be distributed over many fft bins.
Fig. 3 shows the performance with 1024 fft and sin squared windows. This level of processing will be adequate in most situations although the discontinuity spurs are still visible.
The 1024 sin squared window forward plus backward transform use 16.06 + 6.90 = 22.96% of the total available time.

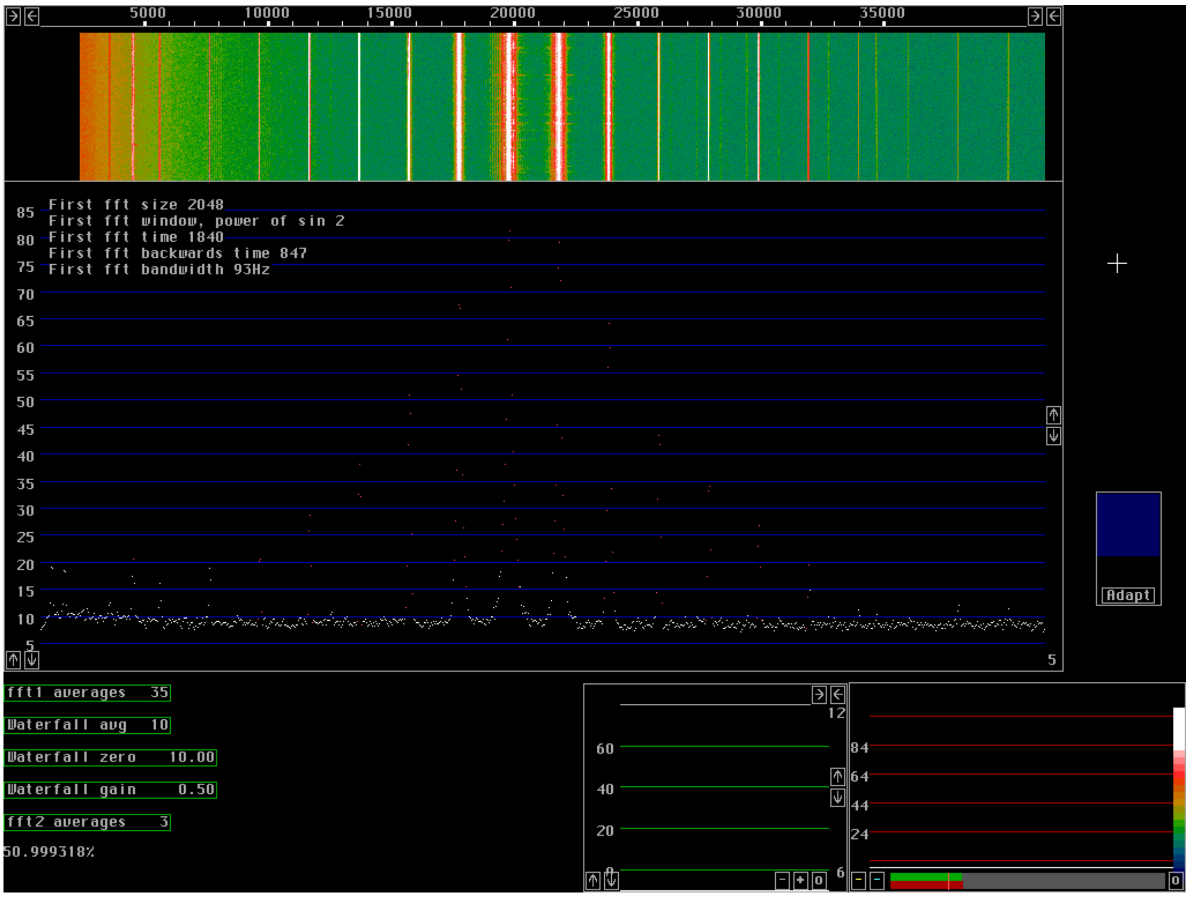


```
First fft size 1024
First fft window, power of sin 2
First fft time 1606
First fft backwards time 690
First fft bandwidth 187Hz
```

```
fft1 averages    35          fft2 averages    3
Waterfall avg    10
Waterfall zero      10.00
48.705559%
Waterfall gain      0.50
```

Fig 3. *First fft size 1024 with sin squared window*

Doubling the transform size to 2048, keeping the sine squared window increases the processing time to 26.61% and practically eliminates the transient sidebands. This is shown in fig. 4.



```
First fft size 2048
First fft window, power of sin 2
First fft time 1840
First fft backwards time 847
First fft bandwidth 93Hz
```

```
fft1 averages    35
Waterfall avg    10
Waterfall zero      10.00
Waterfall gain      0.50
fft2 averages    3
50.999318%
```

Fig 4. *Compared to fig. 3 the transform size is doubled. The processing time does not increase very much, from 23 to 26.6% only.*
As an alternative, the window can be improved. Figure 5 shows a 1024 fft with a sin to power 3 window. The transforms are spaced closer in time so they join each other at a point where the window is about -3dB. for this reason and because the window has to be compensated for in the back transform the time increases more this way as compared to the doubling of the transform size but the difference is small.
Note that the AGC action can be seen in fig 5. The blue regions close to the strongest spectral lines indicate a lower signal level despite the hum and noise sidebands of the signal itself.

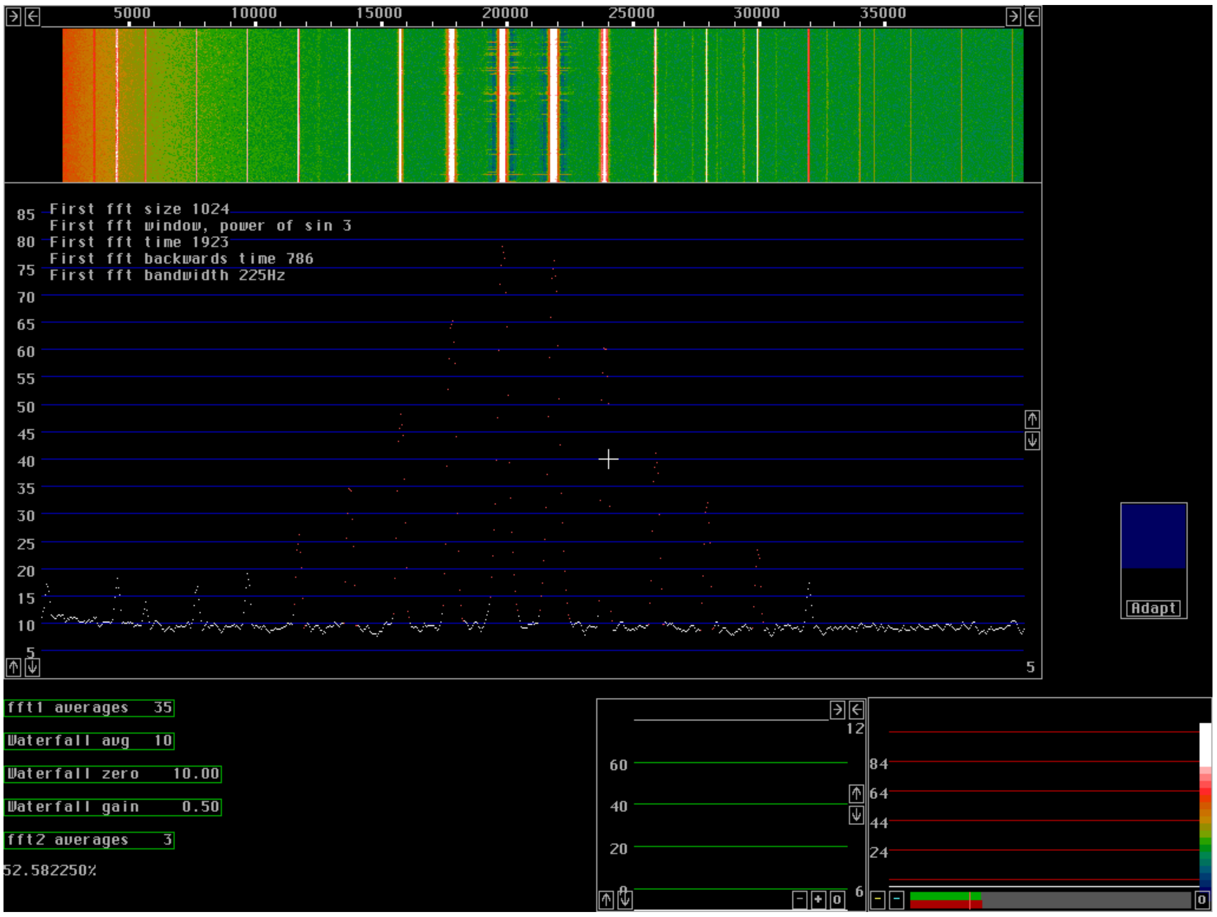
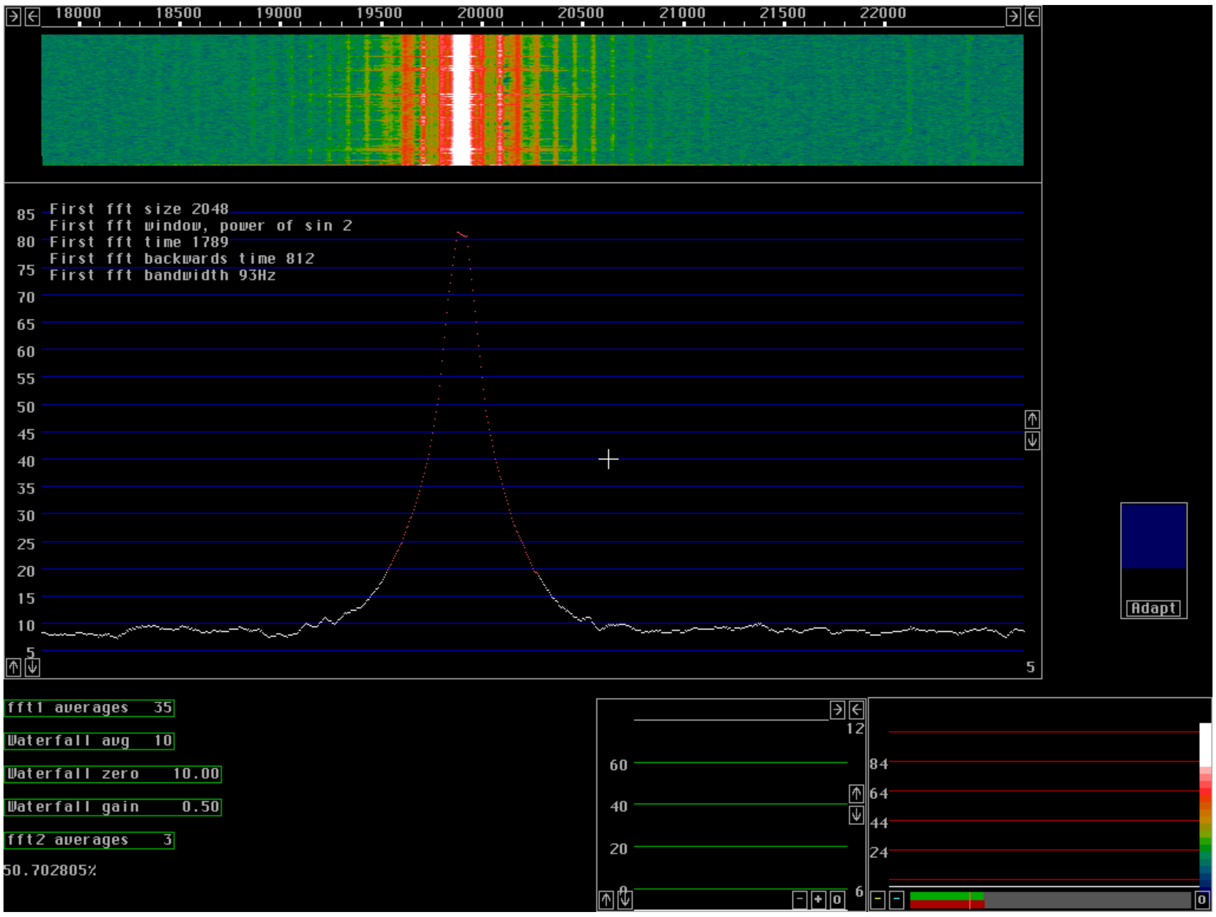

```
First fft size 1024
First fft window, power of sin 3
First fft time 1923
First fft backwards time 786
First fft bandwidth 225Hz
```
```
fft1 averages     35
Waterfall avg     10
Waterfall zero    10.00
Waterfall gain    0.50
fft2 averages     3
52.582250%
```

Fig 5. *Compared to fig. 3 the window order is increased by one to sin to power 3.*

Suggested parameters for the first fft

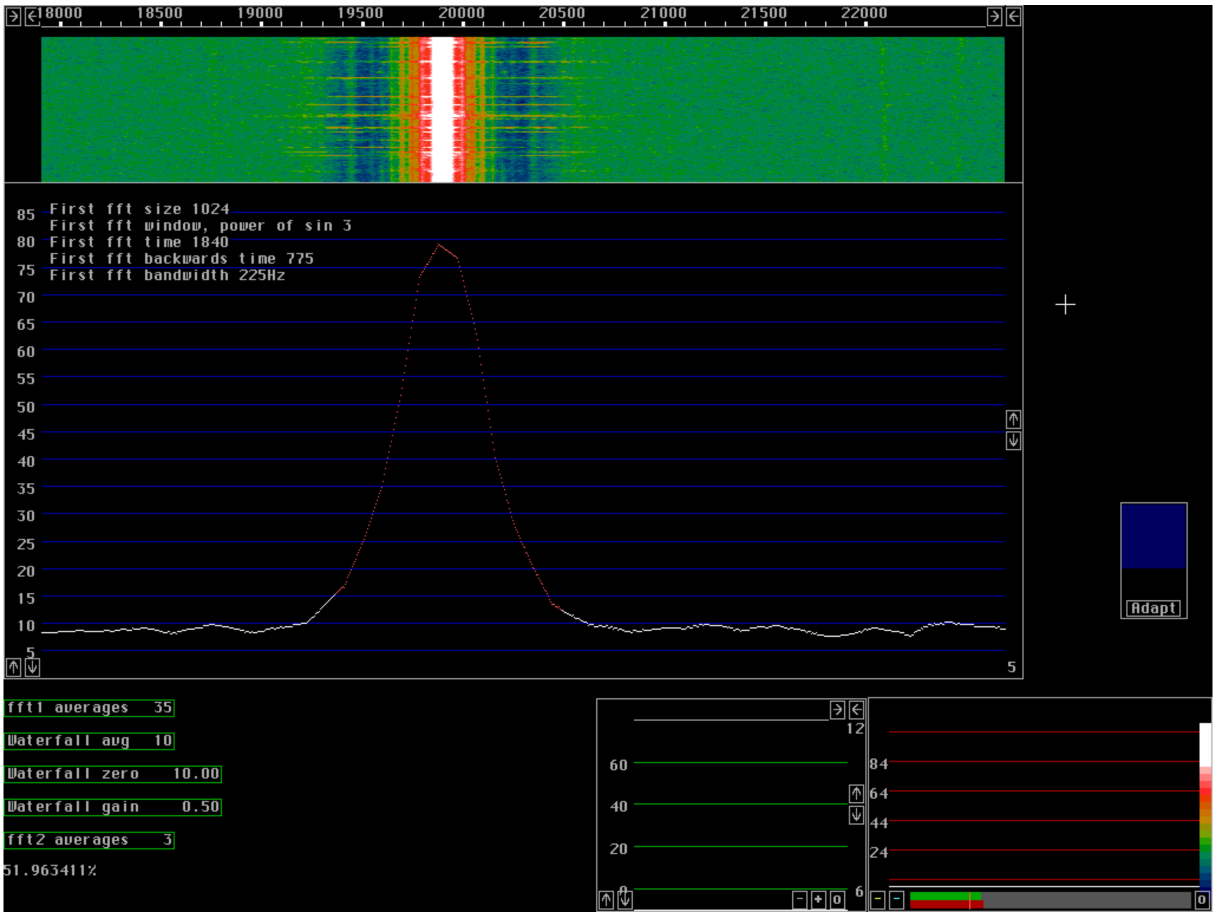
From an inspection of fig. 4 and 5 one would conclude that they are both equally good. This is not really the case as one can see from a comparison between fig. 6 and 7. Both these figures show a zoom wiev on the signal with modulation switched off. It is quite clear that improving the window is better than increasing the transform size.



```
First fft size 2048
First fft window, power of sin 2
First fft time 1789
First fft backwards time 812
First fft bandwidth 93Hz
```

```
fft1 averages     35
Waterfall avg     10
Waterfall zero    10.00
Waterfall gain    0.50
fft2 averages     3
50.702805%
```

Fig 6. *Compare to figure 7.*

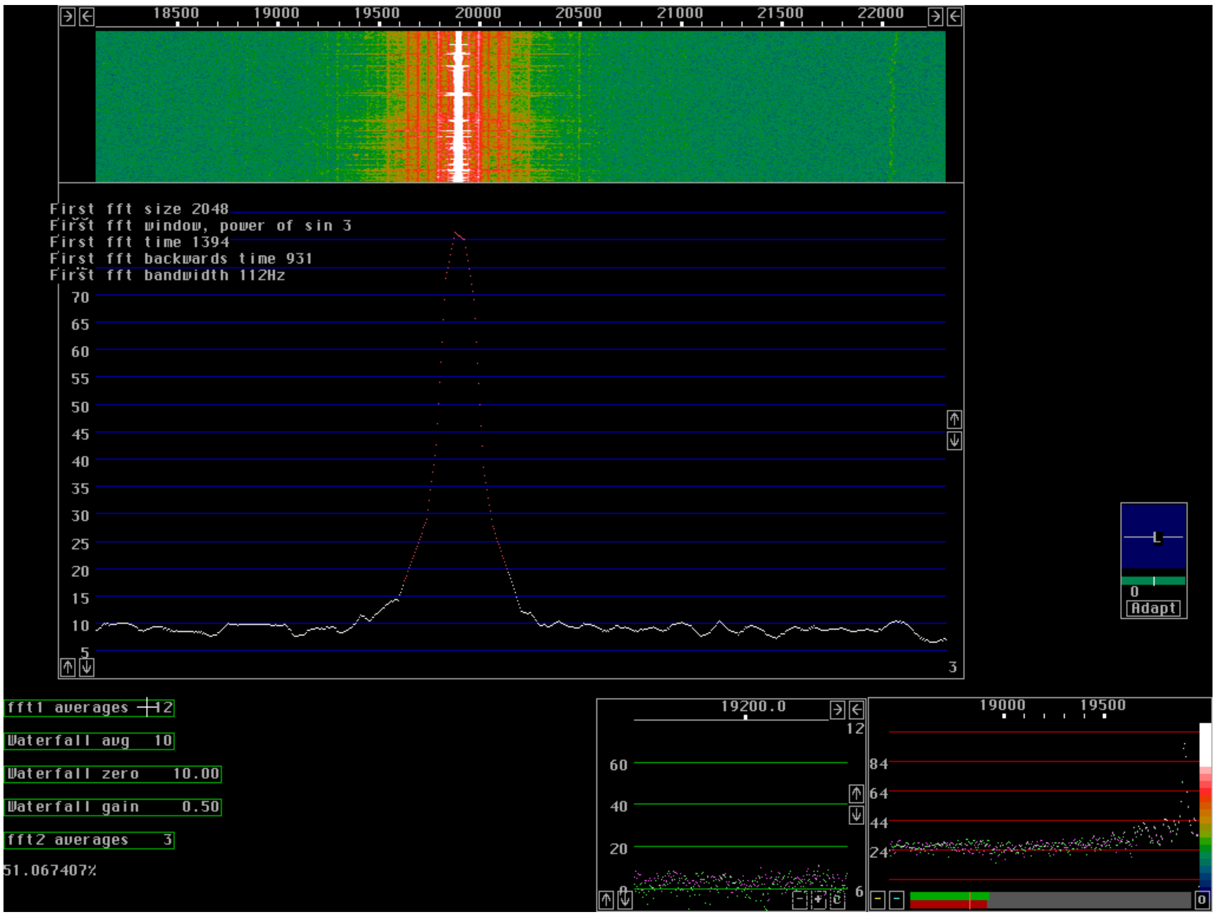


```
First fft size 1024
First fft window, power of sin 3
First fft time 1840
First fft backwards time 775
First fft bandwidth 225Hz
```

```
fft1 averages     35
Waterfall avg     10
Waterfall zero    10.00
Waterfall gain    0.50
fft2 averages     3
51.963411%
```

Fig 7. *An improved window function is better than a twice as large transform. Compare to figure 6.*

All the figures 1 to 7 were produced with the first fft running in its simplest implementation, vers 0.
The Linux pc radio using Delta44 at 96kHz to provide 90kHz bandwidth, when run on a Pentium III, can well use the parameters demonstrated in fig.8. Using the SIMD instructions of Pentium III (fft1 version 5) the total time for forwards and backwards transformation is 23.25% on a 650MHz Coppermine. Note that the sidebands visible in fig. 8 are the 50Hz hum sidebands of the signal generator, not a processing error.



```
First fft size 2048
First fft window, power of sin 3
First fft time 1394
First fft backwards time 931
First fft bandwidth 112Hz
```

```
fft1 averages     12
Waterfall avg     10
Waterfall zero    10.00
Waterfall gain     0.50
fft2 averages      3
51.067407%
```

Fig 8. *Good enough, with margin, is to use 100Hz bandwidth with a sin to power 3 window. For 90kHz bandwidth the transform size becomes 2048.*

Appendix 2

Sliding FFT and DSP Filtering

Sampling

The sampling of a wave form must be done at a high enough speed in order to avoid false data. The sampling works like a frequency mixing with half the sampling frequency and overtones of that. Corresponding to mirror frequencies in frequency mixing are the alias frequencies that occur in sampling. The aliasing is a phenomenon produced by the sampling, and it is there regardless if the data is used for a digital filter or for FFT or just a digital recording. To avoid aliasing, no signal at frequencies above a certain frequency may reach the sampling input. Depending on the intended use of the digital data, the permissible maximum frequency may be higher than the Nyquist frequency, which equals half the sampling frequency. Look here for more details on sampling and anti aliasing filtering. (see Appendix 3)

The digital Data
Let us start at some time t = start time, and sample data at 8kHz. We label the data points 0,1,2,3,4..... This means that we will have a row of numbers stored in the computer memory, and these numbers are x(0), x(1), x(2),x(3), ... These numbers are just numbers, but they contain all information **necessary** to reproduce exactly the original continuos wave form x(t).
x(k) is a measure of the input voltage at a time, t = start time + k/8000 seconds.
Starting at some particular time, with the corresponding sample point number k, we pick N consecutive points. These points, x(k), x(k + 1)..... x(k + N - 1) form an array A[k] of length N, where k is the number of the first sample point in A[k]. For the discussion here, comparing FFT to digital filters, the elements of A[k] are denoted A[k](m), where m is the point number within A[k]. Obviously m is in the interval 0 to N-1.

The Fourier Transform

Any array of N points can be expressed as a sum of N arrays, each one describing a sine wave. There is a one to one correspondence between the function of time, the array of N points, and the N amplitudes and phases that describe the N sine wave functions that, when added together, become the time function. The following basic computer program will calculate the fourier transform of an array X of length N:

```
FOR I=0 TO N/2-1
SUMI=0
SUMQ=0
FOR J=0 TO N-1
SUMI=SUMI+X(J)*SIN(I*J*2*3.141593/N)
SUMQ=SUMQ+X(J)*COS(I*J*2*3.141593/N)
NEXT J
ARE(I)=SUMI
AIM(I)=SUMQ
NEXT I
```

This is the fourier transform for real valued functions. There are better algorithms, known as FFT algorithms, but they do exactly the same thing. The FFT looks up the sine function values in a table, and it avoids doing the same multiplication over and over again as is done in the simple program above. The result however comes out identical, so for the purpose of discussion the above computer program may illustrate the fourier transform.
The fourier transform for complex valued functions is more efficient to use, but the final result is identical. Look here for details: The Complex Fourier Transform

Frequency Response of the Fourier Transform

Each amplitude component in the fourier transform corresponds to a sine wave. Each sine wave has exactly 0,1,2... up to N/2 periods within the time spanned by the N samples. A sine wave at some intermediate frequency will be represented in the fourier transform as several different frequencies present at the same time. If we feed a sine wave signal into a 512 point fourier transform, and sweep the frequency, the spectra obtained look like shown in figure 1. The broadening is usually explained by saying that the fourier transform can only represent periodic functions. The discontinuity represented by the mismatch between the beginning and the end of the input data creates "keying clicks" that have a wide spectrum.
If the input data is white noise with only a very weak signal, the broadening due to mismatch at the ends has little influence, but if there is some signal well above the noise, the broadening of it will cause extra noise that may hide weak signals at moderate frequency separations.

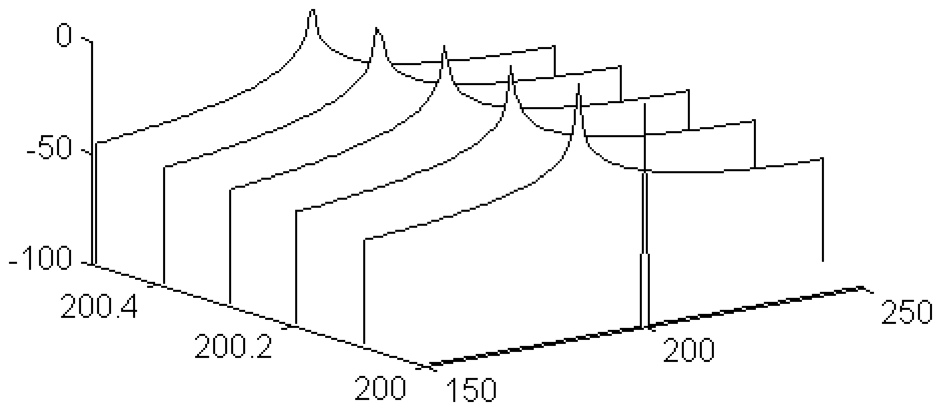


Figure 1. Spectral response of 512 point fourier transform. The width at -20dB varies between 1 and 7 points depending on if the signal falls on a point or between two points. For further details look at this table: Fourier spectra of pure sine waves without windowing

Windowing
The problem of broadening of signals due to mismatch at the ends is solved by windowing. The trick is simple: Force the points to zero at both ends by multiplication by some function that gradually goes to zero at both ends. Obviously a string of zeroes at one end will give a perfect match with the string of zeroes at the other. Windowing removes the tails that stretch far away from the centre frequency, and thus it gives a very significant improvement of the dynamic range. When windowing is used, the resolution becomes lower. The frequency response becomes broader also for sine waves that fit exactly to the fourier frequencies. On the other hand the broadening is controlled and independent on whether the signal is right at a fourier frequency or somewhere between.

An infinite number of window functions is possible. As a demonstration, look at figures 2 to 4.

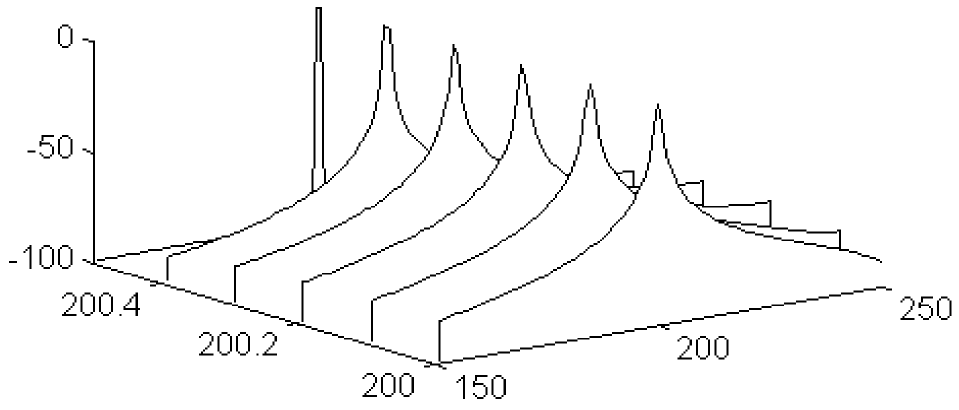


Figure 2. Fourier spectrum using a sine function for window. For details, look at this table: *[Fourier spectra of pure sine waves with sine window]*

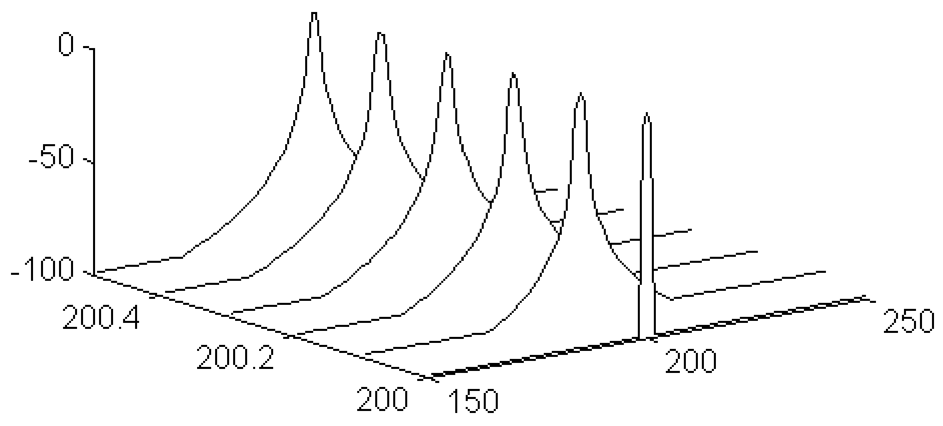


Figure 3. Fourier spectrum using a sine squared for window. For details, look at this table: *[Fourier spectra of pure sine waves with sine squared window]*

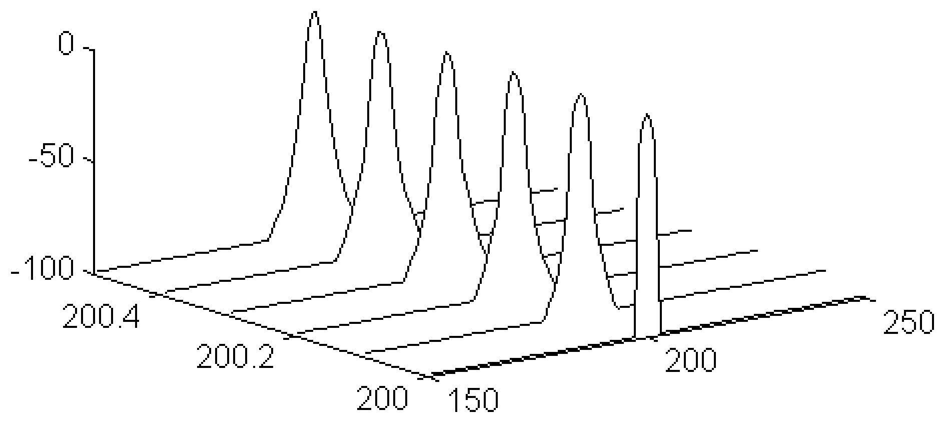


Figure 4. Fourier spectra of pure sine waves with sine^4 window. For details, look at this table: [Fourier spectra of pure sine waves with sine^4 window]

The stop band rejection is greatly improved by windowing. Each transform uses fewer points, so the bandwidth increases. The window function greatly improves the shape of the filters in the equivalent filter bank - and it also makes the equivalent filters overlap.
The windowing is illustrated in the basic code below, which of course is very inefficient, but gives an identical result as a properly written FFT with a sine squared window.

```
FOR I=0 TO N/2-1
SUMI=0
SUMQ=0
FOR J=0 TO N-1
SUMI=SUMI+X(J)*SIN(I*J*2*3.141593/N)*SIN(J*3.141593/N)^2
SUMQ=SUMQ+X(J)*COS(I*J*2*3.141593/N)*SIN(J*3.141593/N)^2
NEXT J
ARE(I)=SUMI
AIM(I)=SUMQ
NEXT I
```

Sliding FFT

When the fourier transform is used without a window function, it is natural to use each point only once, with the notations presented above, this means that the consecutive input arrays for the windowless FFT will be:
A[i],A[i+N],A[i+2*N]....
When a window function is used, some points have to be reused because otherwise the points that happen to be the last or the first point in one of the A arrays will be multiplied by zero, and thus have no effect on the final result. From an information theoretical point of view, all points must be equally significant for the final result, and throwing away something like half the input data has to be a serious waist of S/N. The consecutive input arrays to the windowed FFT procedure will be:
A[i],A[i+N/K],A[i+2*N/K]....
K is a number between 2 and N, and how large it has to be depends on the window function.
If K=1, each point is used only one time, and the input data fits the windowless FFT.
If K=N, each point is used N times. This would always be serious overkill, but it is interesting to look AT what the output would be. As an illustration, the basic program for the windowed fourier transform given above is modified to produce an output for one frequency only, but with one data point out for each data point in corresponding to K=N. This means that one frequency in the FFT is selected, and consequently the process will be some kind of digital filter. Let the frequency of interest correspond to the point IFRQ in the FFT output.

```
1 FOR J=0 TO N-2
X(J)=X(J+1)
NEXT J
WAIT FOR X(N-1) TO ARRIVE FROM A/D CONVERTER
SUMI=0
SUMQ=0
FOR J=0 TO N-1
SUMI=SUMI+X(J)*SIN(IFRQ*J*2*3.141593/N)*SIN(J*3.141593/N)^2
SUMQ=SUMQ+X(J)*COS(IFRQ*J*2*3.141593/N)*SIN(J*3.141593/N)^2
NEXT J
REM DATA IS AVAILABLE IN SUMI AND SUMQ, BUT ONE IS ENOUGH
REM SEND SUMI TO THE D/A CONVERTER
GOTO 1
```

What we see here is a computer program that makes a FIR filter. (Finite-duration Impulse Response) In fact it is two FIR filters with a 90 degree phase shift between them, but we may neglect SUMQ and just route SUMI through a D/A converter to our headphones. A sine wave with amplitude 1 at the input will have an amplitude of about N/2 (depends on the window), so if the noise is allowed to have a maximum amplitude of about 2 bits below the maximum level of the A/D converter, and if we want the output to saturate for a sine wave 6dB below, SUMI has to be divided by N/16 before it is sent to the loudspeaker.
SIN(IFRQ*J*2*3.141593/N)*SIN(J*3.141593/N)^2 is the impulse response of this digital filter, (J goes from 0 to 2N-1) and we immediately see that the window chosen for the FFT becomes the magnitude response of this FIR filter.
When K=N, the complete fourier transform is a filter bank with N/2 evenly distributed equal digital filters. An average power spectrum is obtained by computing the average of the output signal squared for each frequency. Such average power spectra is the best method for detecting the presence of weak signals in noise. For maximum sensitivity, the length of the window function (in time) should match the coherence length of the expected signal. Too long windows make no harm, but require more computing resources without improving sensitivity.

Using the complex amplitude

Consider the filter above producing SUMI and SUMQ. This complex pair gives a complex amplitude, and this complex number describes the output from the FIR filter. Now, since the filter is narrow, the output has to be a narrow band signal, and consequently SUMI and SUMQ do not change fast with time. The actual numbers change because the reference for

the phase angle they describe changes, but if this is taken into account, they change very slowly. This can be used to update SUMI and SUMQ relatively infrequently.
One way to use the complex amplitude is to select a frequency within the FFT, (or a few frequencies if more bandwidth is desired) The filtered output signal is then obtained from the centre part of the backwards fourier transform. If the phase is properly managed, successive inverse transforms will match each other and form a nice filtered output signal. If K is made a little too small, each sequence becomes a little too long, and the matching becomes less good. This would create wide band noise, while saving a lot of computing time. A simple band pass filter at the output (digital, before the D/A or analog after it) will restore a nice signal.
Another way is to use SUMI and SUMQ to set the amplitude and phase of a local tone oscillator at a fixed frequency. The amplitude and phase will be updated every (N/K)'th sample. A quite conventional analog filter at the selected fixed frequency will conveniently remove high frequency clicks which makes very low values for K give satisfactory results.
Demo program for sliding FFT
The "basic program" below is intended to show how the system I currently use for weak signal communication works. The system is implemented in a TMS320C25, which provides two complete sets of FFT's, one for each polarisation of my cross yagi system, and a 80186 that does the rest (and is nearly idle). How to combine the two FFT's coming from two orthogonal polarisations is described here: Adaptive polarisation

The algorithm described below, partly as plain text and partly as basic code, will produce a bandwidth that depends on the window length. For a sin squared window, a suitable value for K is N/4. The window I use with N=512 and K=128 is slightly more rectangular and produces a bandwidth of about 17 Hz. If you wish to listen to it, look here: Station XX, a demonstration of a weak EME signal

This algorithm has two outputs, one for the screen, which is the average power spectrum, and one for the loudspeaker/headphones. Very weak signals are easily seen on the average power spectrum, and the program assumes that the operator has pointed the variable IFREQ to a point close to one of the possible peaks in the power spectrum.

The "program" below is of course only intended to illustrate the method. The real code does not multiply with numbers known to be zero, and it has a buffer allowing the average power spectrum to be calculated from points both before and after the time actually being processed for output. Further, only those transforms containing above average energy close to IFREQ, are used to calculate the average power spectrum.

Subroutine getfft: Wait for N/4 new points to arrive (complex or real). Produce windowed fft of 3N/4 old plus N/4 new points. Store the NMAX complex amplitudes in ARE and AIM. NMAX=N/2-1 for real fft and N-1 for complex fft.

01 CALL GETFFT

Loop I to form average power spectrum.
```
02 FOR I=0 TO NMAX
03 PWR(I)=127*(PWR(I)+ARE(I)*ARE(I)+AIM(I)*AIM(I))/128
04 NEXT I
```

Update power spectrum on screen. Possibly small part each time.

Locate the maximum in the power spectrum and get peak shape. If the level is below MINPWR, keep the old peak position and shape.
```
05 MAXPWR=-1
06 FOR I=IFREQ-4 TO IFREQ+4
07 IF(PWR(I) < MINPWR)GOTO 11
08 IF(PWR(I) < MAXPWR)GOTO 11
09 MAXPWR=PWR(I)
```

```
10 NEWFQ=I
11 NEXT I
12 IF(MAXPWR < 0) GOTO 14
```

Subroutine shape: Fit a parabola to the max point and the two surrounding points. In this way, the frequency will come out with decimals from the decimal position of the maximum point on this parabola. Store the frequency as a point number with decimals in IFREQ. Find the average of the points surrounding the peak (noise floor) Subtract the noise floor from the peak, and store in PWRCLEAN. Make PWRCLEAN=0 outside IFREQ +/- 3. Finally make the sum of the square of all points in PWRCLEAN = 1
13 CALL SHAPE

rem Multiply the complex amplitudes by the normalised cleaned average
rem power spectrum in PWRCLEAN. This is the real filtering.
rem It is a matched filter because it means it means
rem filtering through a filter with a frequency response that
rem is equal to the power spectrum of the signal.
rem If the signal is in one channel only, the process just means
rem pick that channel as it is.

```
14 FOR I=0 TO NMAX
15 BRE(I)=ARE(I)*PWRCLEAN(I)
16 BIM(I)=AIM(I)*PWRCLEAN(I)
17 NEXT I
```

Produce a single complex amplitude from those few that remain. It is not meaningful to do any sophisticated back transformation because the time between each new fft is too long in this example (K=N/4). As long as the signal is narrow banded enough to be compatible with update time interval, just summing points with alternating sign works fine. The whole procedure becomes equivalent to picking the complex amplitude at the peak, and it is independent on if the peak is centred on a point in the fft or between points.

```
18 S=1
19 REH=0
20 IMH=0
21 FOR I=1 TO NMAX
22 S=-1*S
23 REH=REH+BRE(I)
24 IMH=IMH+BIM(I)
25 NEXT I
```

If two orthogonal antennas are used in a stereo system, here is the point to combine the two complex amplitudes REH,IMH and REV,IMV into a new set of complex amplitudes in such a way that one contains all the signal (and some noise) and the other contains only noise.

Convert complex amplitude to amplitude and phase.
26 AMPLITUDE=SQR(REH*REH+IMH*IMH)
27 PHASE=ATAN2(REH,IMH)-2*3.14159*IFREQ*(K/N)

Expand the dynamic range of amplitude by table lookup. Approximately exponential function to compensate for logarithmic characteristics of human ears.

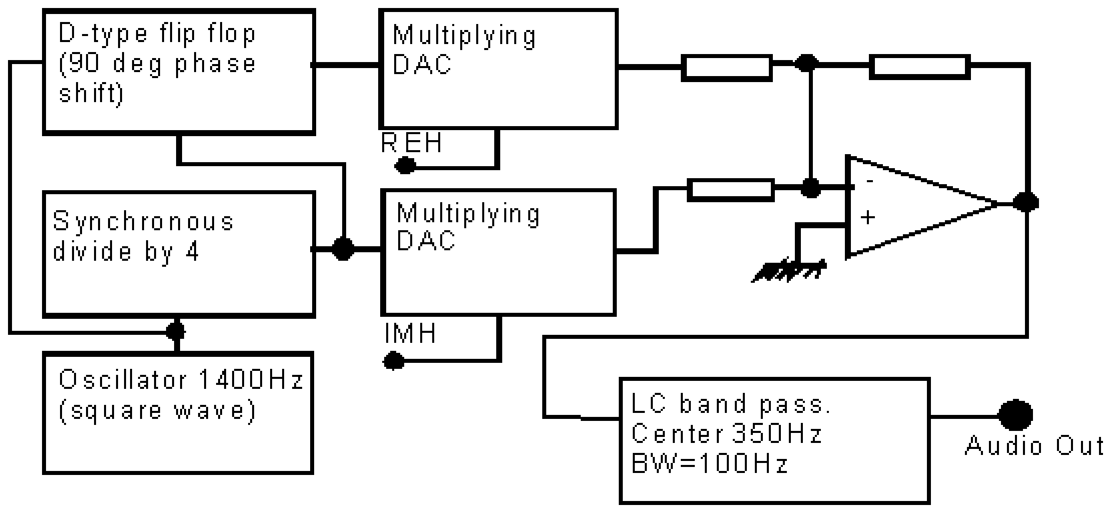
```
28 CALL EXPAND(AMPLITUDE)
```

Restore complex amplitude (with proper phase)

```
29 REH=AMPLITUDE*SIN(PHASE)
30 IMH=AMPLITUDE*COS(PHASE)
```

Send REH and IMH to the hardware to generate a sine wave with this complex amplitude (see diagram below). Then go back to start of loop.

```
31 CALL AMPLOUT
32 GOTO 01
```


D-type flip flop (90 deg phase shift)
Multiplying DAC
REH
Synchronous divide by 4
Multiplying DAC
IMH
Oscillator 1400Hz (square wave)
-
+
LC band pass. Center 350Hz BW=100Hz
Audio Out

One way to produce a sine wave from digital complex amplitudes. The narrow output filter removes clicks from phase and amplitude discontinuities. It also converts the square waves into sine waves.

I guess a soundblaster can produce a sine wave from complex amplitudes directly as two "musical instruments" being a sine and a cosine function. Then, clicks may be removed just by interpolation in the complex phase. It is not necessary to do fft's more often.

Multichannel filtering

The sliding FFT can easily provide the complex amplitudes for several different stations simultaneously. The 80186 I use, can keep long buffers with complex amplitudes for about 15 stations simultaneously, and it has a lot of idle time. Of course it is very tempting to try to convert the contents of these buffers to ASCII code and put them all out on the screen. My attempts in this direction have not been very successful despite quite a lot of effort. I cannot make the computer nearly as clever as my ears in receiving weak EME signals. The main problem is the QSB. Maybe in the future....

Appendix 3

Sampling and Anti Alias Filtering

Sampling

The voltage across the loudspeaker output of a receiver is a function of time. Let us call it x(t). Of course time is continuous, but since x(t) cannot vary very fast because of the limited bandwidth of the receiver, the function x(t) is completely known if it is known at a limited number of points. We may use a computer to sample at 8 kHz. This is standard .WAV format with a Sound Blaster and a PC, and the real wave form will be identical to a smooth wave form that is fitted to these points if the receiver does not produce any signal energy at all above 4 kHz.

Aliasing is a phenomenon that limits the useful spectrum width when using fourier transforms. It has nothing to do with this particular use of the sampled data, it comes from the sampling process itself. I think it is best described by an example.

Assume data is sampled at 8 kHz. The highest frequency that can be represented with such data points is 4 kHz. With all odd points = +1 and all even points = -1, the data corresponds to a sine wave of frequency 4 kHz and amplitude 1, and this is one possible result when sampling a 4kHz signal at 8kHz. Another possible result is that all points are = 0. That would happen if the sampling points accidentally happened to coincide with the zero crossings of the sine wave.

If the frequency is slightly below 4 kHz, say (4-x) kHz then the phase will vary slowly, so the data points will be +/- 1 for a while, and then gradually go through 0 towards +/- 1 again. If we look at the odd points only, they will be a perfect sine wave of frequency x. The even points will just have reversed sign.

When this signal is fourier transformed, the transform will give a large amplitude for the frequency (4-x) kHz and nothing else.

If we feed (4+x) kHz into the sampling input, still sampling at 8 kHz, the data points will be identical to the ones we got at (4-x) kHz. When we take the fourier transform, the result will be a large amplitude at (4-x) kHz and this is incorrect. Aliasing means seeing frequencies within the passband that really belong outside.

Filtering

Aliasing has to be avoided. It corresponds to mirror frequencies while the sampling process corresponds to frequency mixing with half the sampling frequency. When using fourier transforms for weak signal monitoring a reasonable maximum level for alias frequencies could be -60dB. This means that local stations have to be 60 dB stronger than the signals we look for before they start to cause confusion through aliasing.

Usually it is assumed that one needs a filter that removes all signals above the Nyquist frequency, half the sampling frequency. This is however not necessary. If we have a low pass filter that falls from 0 to -60 dB in half an octave, we may place -60dB at 4.0 kHz, but then the flat response goes only up to 2.82kHz, square root of two lower. By placing the -60dB point at 4*1.189 kHz = 4.76 kHz, the flat response will go up to 3.36 kHz, still square root of two below. By doing that, the digital data may be incorrect for frequencies between 3.36 and 4.0 kHz, but by removing that frequency range in the digital signal we have a larger bandwidth with flat frequency response, without any aliasing. This is particularly convenient if the data is used for calculating FFT spectra. Removing frequencies above some limit just means discard the points above some upper limit in the spectrum.

There are more aliasing frequencies corresponding to mixing with overtones of the Nyquist frequency. It is trivial to make sure they do not impair the digital data because they are far off in frequency.

Appendix 4

The Linrad network

General

Computers have hardware that allows them to be connected to other computers via a network. Such hardware is called a network interface and there may be several of them in the same computer. Each network interface must have an IP address and there may be several IP addresses in one computer.

Linrad uses the network to make raw or processed data available to other programs that may run on the same computer or on any other computer in a local network.

If network transmit is enabled, Linrad will blindly transmit UDP messages to the IP addres that was specified during Linrad setup. The default address is in the range 239.255.0.0 to 239.255.0.15 and no computer should have an address in this range. Datagrams sent to one of these addresses is intended for a group, the group of all computers that want to join the group. When more than one computer is listening, Linrad is multicasting data on the network. Routers typically do not send such packages to the outside world.

It is possible to set the address to 127.0.0.1 to transmit UDP to other programs in the same computer. Such packages will not reach the outside world. It is also possible to set something like 192.0.0.10 to send data to a particular computer.

Firewalls will not allow the UDP messages of Linrad by default. When setting up the Linrad network it is a good idea to physically disconnect from the Internet and other computers not directly involved in using the Linrad transmissions. By temporarily disabling all firewalls one can make sure that everything is set up properly. It is a good idea to use fixed IP addresses to make sure that DHCP does not change the addresses from time to time. When everything works properly one can enable the firewalls again - but personally I think it is better to not use firewalls in any computer in my own local network. A single but very restrictive firewall in the ADSL modem is adequate protection from the Internet. Others may not want a restrictive firewall for all computers and then it might be a good idea to have one in each computer.

To receive the packages into another instance of Linrad, one has to set the same address that the transmitting instance of Linrad is using.

When an address outside the range 239.255.0.0 to 239.255.0.15 is desired, a file par_netsend_ip or par_netrec_ip has to be present in the Linrad directory. The file has to be a plain text file (ascii) containing one line with the IP address in the standard format like 192.1.2.38

The Linrad base port has to be set between 50000 and 65000 in steps of 10 and Linrad will send data to a port with an offset with respect to the base port that depends on the data format.

Table 1 shows the Linrad data formats and offsets used in the Linrad network.

ormat	Offset	Description
RAW16	0	Raw data in 16 bit format.
RAW18	1	Raw data in 18 bit format.
RAW24	2	Raw data in 24 bit format.
FFT1	3	Fourier transforms in float format, the output of fft1.
TIMF2	4	The output from the noise-blanker 16/32 bit int/float.
FFT2	5	Fourier transforms, the output of fft2 16/32 bit int/float.
BASEB	6	Baseband in 16 bit format.

Table 1. *The different formats used in the Linrad network and their port offsets from the base port.*

A single Linrad program can send all these formats simultaneously to the network.

Raw data or fft1 transforms can be used as the input to Linrad and thereby allow several operators to listen simultaneously to the same receiver/antenna. The network can also be used to distribute the processing between several computers. A computer with network input can send the output of later processing stages to the network for other computers to process. It will then use the same base port as the one used for the input, but it must be given another group address.

Raw data formats.

It is possible to run many instances of Linrad simultaneously on the same computer when using X11 or Windows. Figure 1 shows four instances of Linrad on a Pentium IV running a single audio channel at a sampling speed of 96 kHz. One is as a master using a modified Delta 44 soundcard for input while the other three use the different raw data formats of the network.

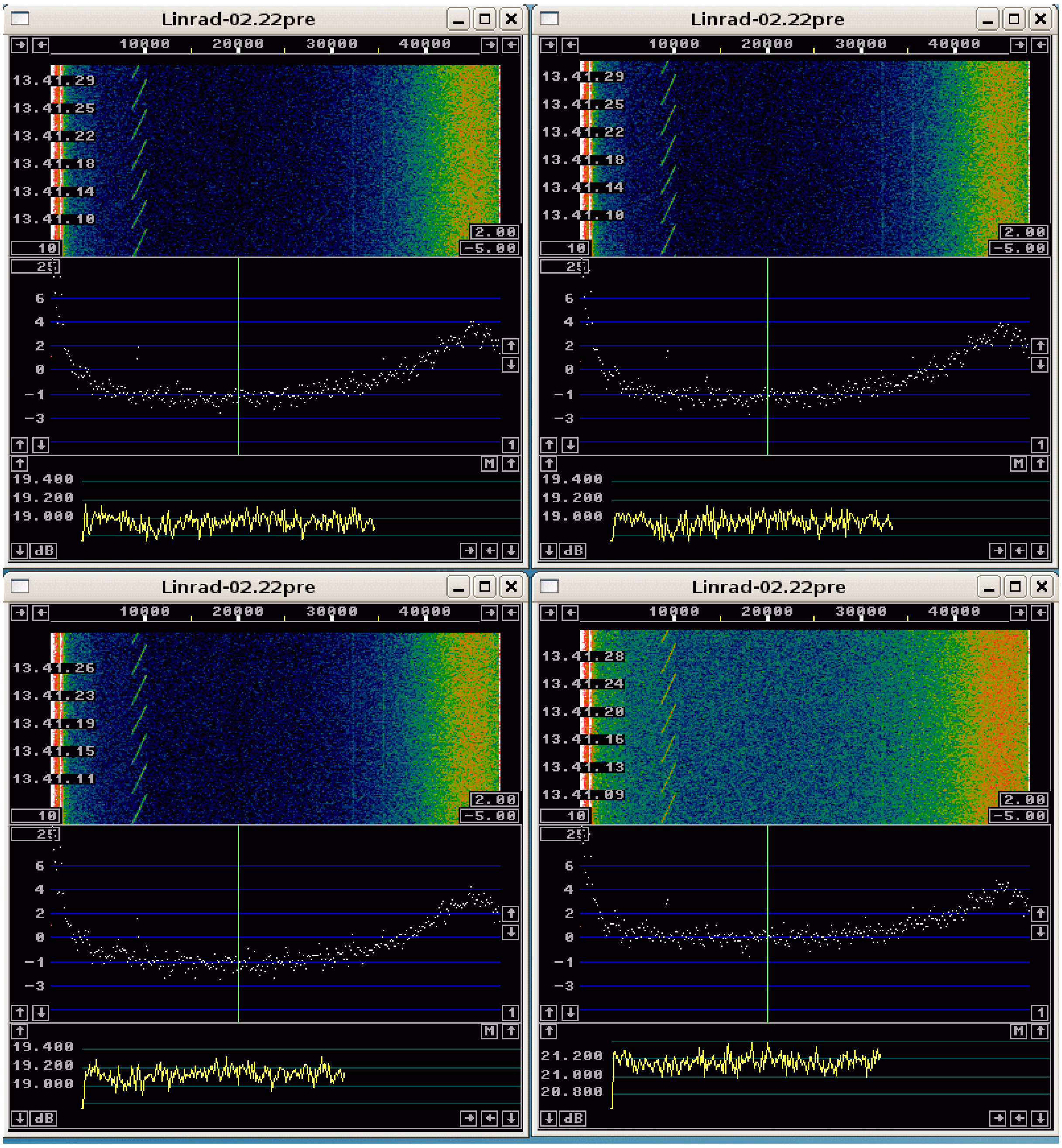


Fig 1. *Four instances of Linrad running simultaneously under X11 on a Pentium IV computer. The upper left waterfall is generated by the master which reads 24 bit data from a modified Delta 44 soundcard with nothing connected to its input. The slanted lines are from a function generator that produces a sweeping sinewave. It is coupled magnetically to the soundcard and the level is set for a weak signal. The upper right waterfall is generated by a slave using the 24 bit signal from the network. It is identical to the master waterfall. The lower left is generated from the 18 bit network data. A small degradation is visible. The lower right is generated from the 16 bit data and it is obvious that S/N is degraded significantly.*

When a 10 V p-p signal (3.53V RMS) is applied to the input, the dB scale reads 119.2 dB. With all 24 bits in use, the level in a 10.0 kHz bandwidth is 19.0 dB which means that the noise voltage at the input is 340 nV in a bandwidth of 1 Hz. Since the thermal noise in 10 k ohm is 12.6 nV the noise figure (referenced to 10 k) for the soundcard in this experiment is 24.3 dB.

The 18 bit data format which was introduced to make the saving of raw data files possible on the Pentium III computer that war used in the early phases of Linrad development causes a degradation of the noise figure by 0.2 dB. This is a perfectly acceptable loss of dynamic range for a 25% reduction of file sizes. It is obviously perfectly adequate for use in a networked system.

The 16 bit format is 11% smaller than the 18 bit format but the associated dynamic range loss is 3 dB. It is perfectly ok to use it for network or disk storage, but it requires 3 dB more system gain for the same system noise figure and therefore saturation on strong signals will occur 3 dB earlier. In most situations this is perfectly OK.

The fft1 format.

The first FFT has to use floating point arithmetics and may be the dominating load for the CPU. By sending transforms rather than the raw data it may be possible to use old computers as slaves on the network.

The network itself is a heavy load to old computers however so receiving all the bits via the ethernet interface might be too much and then it does not help that most of the processing is done already. The number of bytes that have to be transferred for the first FFT is four for each sample from the soundcard multiplied by a factor that depends on how much the transforms overlap. A Delta 44 that samples two RF channels at 96 kHz in complex format (I and Q) reads 96000*4=384000 samples/second. If no window is used the data rate for transferring fft1 over the network then is 1.536 MB/s or 12.3 megabit/s. Such a rate is close to the limit for a Pentium processor at 133MHz. It is actually possible to use one as a slave but one has to make waterfalls small and averaging on spectra reasonably large to not use too much cpu time on the graphics. The audio output would have to use a low sampling speed. One has to set a large bandwidth reduction for the first mixer in order to have a low load from the baseband and disable the second fft.

Without windows the dynamic range becomes lower but the effect is not severe when large transforms are used. With a sine squared window the fft1 data rate is twice as high as without a window and using a Pentium 133 MHz as a slave is then impossible. A 200 MHz Pentium MMX can be used as the slave in this case, but only without the second fft and with low load from graphics and output processing.

On a 800 MHz Pentium III it does not matter much what format one selects for the network. Table 2 shows some timing information.

Input	Format	CPU load (%)	Data rate (MB/s)
Delta44	16bit	10.2	0.768
Delta44	32bit	10.7	1.536
Network	RAW16	11.2	0.768
Network	RAW18	13.2	0.864
Network	RAW24	12.9	1.536
Network	FFT1	10.0	3.688

Table 1. *Timings for Linrad on a 800 MHz Pentium III with standard parameters for the WSE converters and a Delta44 soundcard. FFT1 bandwidth = 100 Hz, window = sin^3 and fft1 version =5 (SIMD). For computers of this generation and later the CPU power is not a limiting factor when running Linrad in terminal mode. The table shows that the network is not a limiting factor either. When running Linrad as a master on 32 bit input while sending fft1 data to the output the CPU load increases from 10.7% to 19.2%.*

The timf2 format.

The second time domain signal in Linrad is the full bandwidth output of the noise blanked data. This format is intended for input to other softwares that might need a wideband noise blanker. One example is MAP65, a wideband JT65 decoder under work by K1JT.

Linrad can not use this format as input. The reason is that strong signals are treated differently depending on whether they are the desired signal or not. In a JT65 decoder, the strong signals will not be interesting and the way they will become distorted by the selective AGC that affects all strong signals which the master computer regards as not interesting should not destroy readability. For SSB voice signals it is different however and for Linrad there is no reason to have the noise blanker in the master computer.

The fft2 format.

The output of the second FFT will normally fit in much less than 16 bits because of the selective AGC and the noise pulse removal of the blankers. These transforms can therefore be transfered to other computers and stored for use in separate softwares. With two RF channels of 96 kHz bandwidth the data rate is 1.5MB/s with a $\sin^2$ window.

One intended usage is for a separate program zoom by which one can zoom in or out on waterfall spectra that contain all the information from the latest hour or so. One hour of data would require 5.4 gigabytes of storage if all 16 bits were retained. It should be enough to save 8 bits and storage can be on the hard disk because it will be ok if moving focus takes a little time. Optimum bandwidths for the second FFT may range from a couple of millihertz up to maybe 10 Hz depending on the interests of the user. With only 1000 pixels on the screen and say 5 pixels over the resolution bandwidth the usefullness of a zoom tool is obvious.

Appendix 5

Perseus with Linrad

The Perseus. Hardware and softwares.

The unit is designed by Microtelecom s.r.l. in Italy. The unit samples at 80 MHz and uses an FPGA to do digital downconversion , filtering and downsampling to an output suitable for transfer via USB 2 to a PC computer.

Linrad-02-45 and later versions can use the Perseus hardware for input. Besides the software, perseus.exe that Microtelecom supplies with each unit and on their web site, Perseus can also be used with Winrad which can be downloaded from The WEAKSIGNALS pages of Alberto, I2PHD

Linrad is more complicated than Perseus or Winrad and has a much steeper learning curve. On the other hand Linrad allows far more special settings that are optimized for various purposes.

Waterfall sensitivity.

The perseus allows an output sampling rate of 1 MHz with a full performance bandwidth of 800 kHz. This makes an excellent tool for searching for extremely weak signals on microwave bands where the total number of signals is low. This link Comparison between waterfall sensitivities explains why the Linrad waterfall is both much faster and much more sensitive than conventional waterfall graphs.

Two-tone test for IM3.

Analog receivers are usually well characterized by IP3, but A/D converters are not. The Perseus HF receiver has an intermodulation-free dynamic range of 104 dB in 500 Hz bandwidth with an IP3 of +27 dBm for signal levels above -20 dBm. At low signal levels, down to about -70 dBm, the signal at the IM3 frequency is at the same level as the internal noise floor in a bandwidth of 500 Hz, -125 dBm (NF=22 dB.) All these numbers are with dither, preamp and input filters enabled and with the attenuator set to 0 dB. See figure 1 for details.

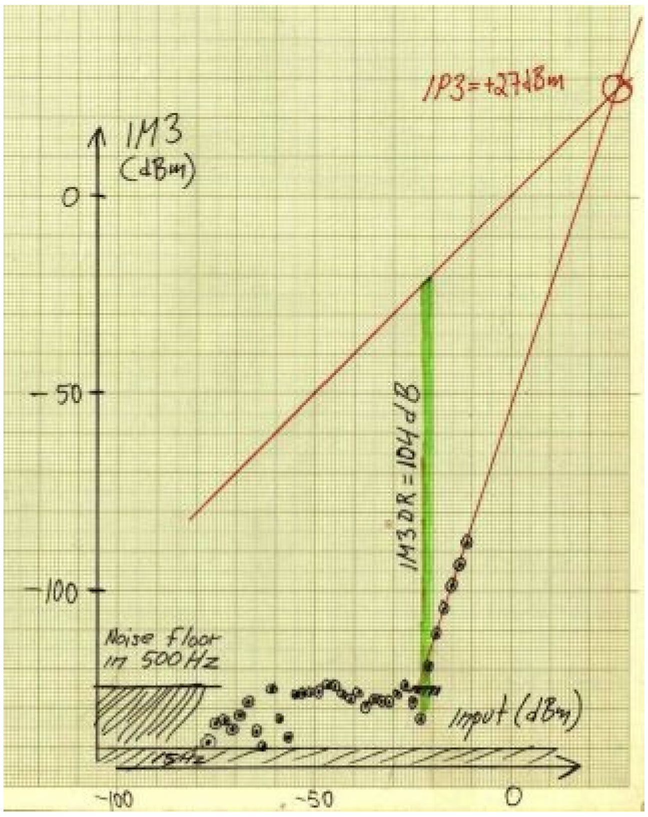


Fig. 1. *Two-tone test with Perseus at 10.7 MHz. Preamp ON, Dither ON, Att 0dB. The measurements were made at a bandwidth of 15 Hz*

Two-tone test for IM2.
Second order intermodulation can be a problem in Europe because of the strong broadcasting stations on the 41 m band.

Figure 2 shows the test setup for IM2 measurements. For the test, generator 1 was set to 14.21 MHz while generators 2 and 3 were set at 7.1 and 7.12 MHz respectively. The output on 7MHz was +16 dBm for each tone and the "pilot tone" on 14.21 was set to -84dBm, 100 dB below the test tones.

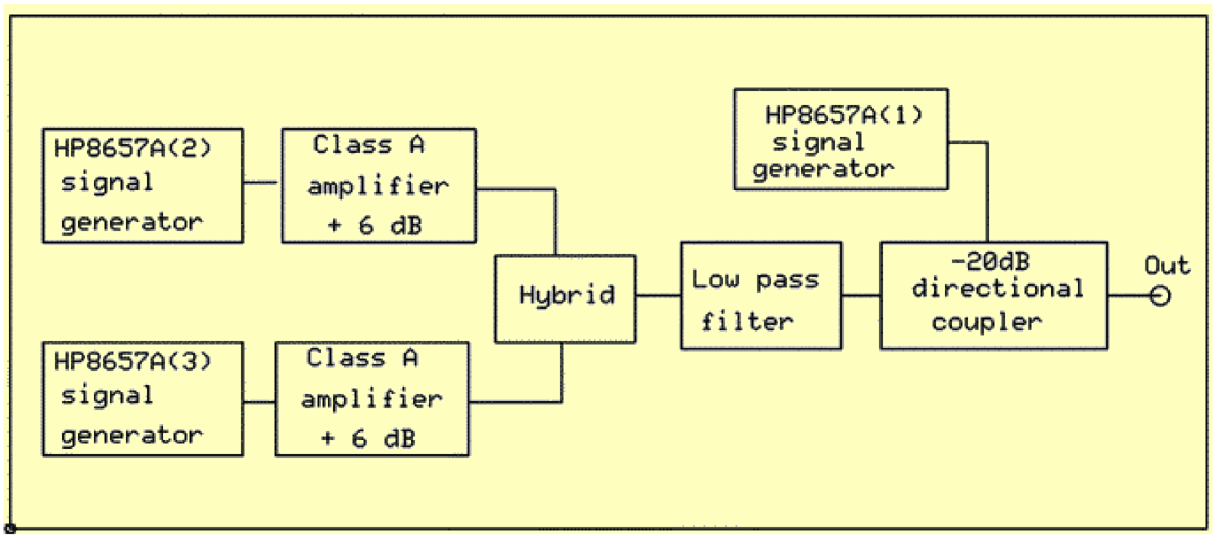

HP8657A(2) signal generator
Class A amplifier + 6 dB
HP8657A(3) signal generator
Class A amplifier + 6 dB
Hybrid
Low pass filter
HP8657A(1) signal generator
-20dB directional coupler
Out

Fig. 2. *Setup for two-tone IM2 test.*

The low pass filter and the 20 dB coupler are designed with iron-free inductors so they should allow a very clean signal on 14 MHz, free from overtones and intermodulation generated in the class A amplifiers. The output is routed through a stepped attenuator into the test object. The "pilot tone" is used to calibrate the S-meter of the test object. (Linrad allows the S-meter to show dBm values.). Figure 3 shows the test result.

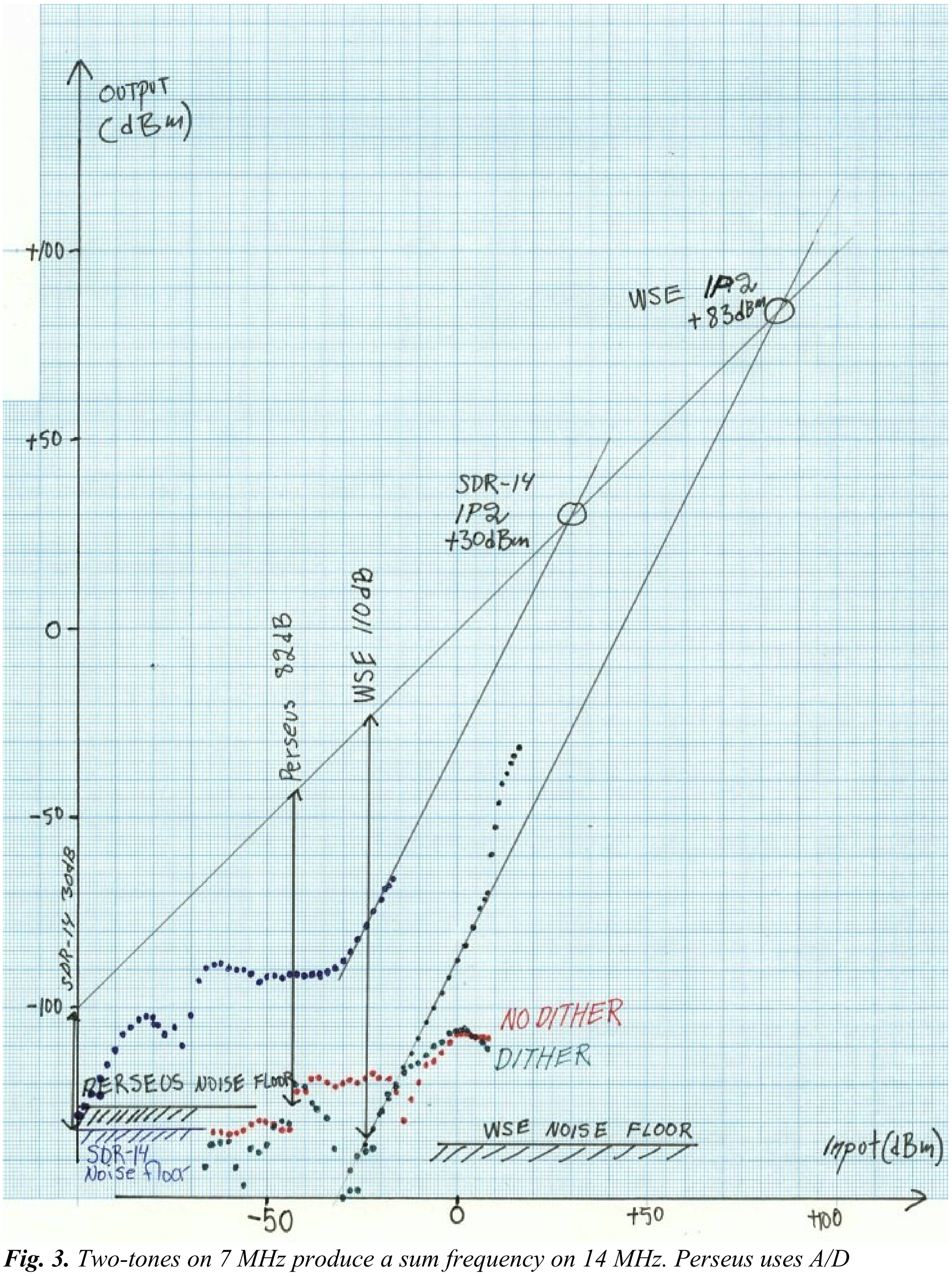


Fig. 3. *Two-tones on 7 MHz produce a sum frequency on 14 MHz. Perseus uses A/D converters while the WSE converters use analog technology. The measurements were made at a bandwidth of 15 Hz. Red and green points are Perseus with and without dither. Blue points are the SDR-14 and black points are the WSE converters. The noise floor levels in this image are for 500 Hz bandwidth.*

It is immediately clear from figure 3 that Perseus behaves quite differently from the WSE converter chain. The sum frequency on 14 MHz is well above the noise floor for 7 MHz signals down to -40 dBm. Inserting an attenuator is not a solution, the desired signal would drop, but the sum frequency would not. Adding a preamplifier would be a better idea as long as the level on 7 MHz would stay below 0 dBm. A simple tuned amplifier that has 10 to 20 dB gain on 14 MHz but that does not amplify on 7 MHz would bring the sum signal below the noise floor on 14 MHz.

The IM2 tests on the Perseus were made with the preselector and preamplifier enabled. It is pretty obvious from figure 3 that the SDR-14 needs a preselector for usage on 14 MHz in Europe.

The second order intermodulation-free dynamic range could be defined as the signal level where IM2 products are equal to the noise floor. With that definition, SDR-14 has IM2DR=30dB while the Perseus is at 82 dB and the WSE converters at 110 dB. With 40 dB of gain, the SDSR-14 would reach an IM2DR of about 60 dB but that would not be very useful since in-band signals would be likely to cause saturation. The SDR-14 needs an external preselector.

Appendix 6

Dynamic range observations for the SDR-14, a VHF sampling radio receiver.

Third order intermodulation vs signal level

Conventional receivers suffer from third order intermodulation (IM3) at high signal levels. Inside any device that is not perfectly linear currents of the second harmonics of strong signals will circulate. These second order products will mix with other signals to produce third order interference at frequencies $2F_1$-F_2 $2F_2$-F_1

When testing receivers one makes the amplitude of F_1 and F_2 equal and then the third order intermodulation product typically follows the amplitude to power three. This means that IM3 is reduced by 30 dB when the test signals are reduced by 10 dB and that the IM3 products disappear completely when the test signals are reduced one third of the way between total breakdown and the noise floor. Analog receivers have an intermodulation-free dynamic range. For more details, look here: Intermodulation in analog receivers

Digital receivers behave differently. There are two distinctly different mechanisms that generate interference at the same frequency where we expect IM3.

One mechanism is the non-linearity of the A/D converter which can not be described well with a polynomial (see the above link) and therefore does not follow the third order law. An A/D converter might have an error in the order of 0.5 or 1 bit when it steps from a binary value like 011111111111 to 100000000000. Such points of non-linearity may be located close to zeroand even small signals become affected by it. The amplitude of the generated interference is essentially independent of the signal level.

Another mechanism is coupling from the data bus to the A/D input. With a test signal at F_1 and a sampling clock at F_s the digital data lines that send the data from the A/D converter to the first data processing chip will contain strong components of F_s+F_1, F_s-F_1, $2F_s$+F_1, $2F_s$-F_1,..... The analog bandwidth of a radio A/D converter is typically large enough to give full sensitivity to all these frequencies. They seem to be signals at F_1 due to aliasing effects but since they vary in amplitude in a strongly non-linear fashion they create a non-linear behaviour.

The phenomenon is illustrated in figure 1 which shows a spectrum analyzer screen of a signal picked up by holding a probe near the AD644 in an SDR-14 when subjected to a two-tone test. The details of such spectra depend on signal levels as well as on which lines on the data bus that couple best to the probe.

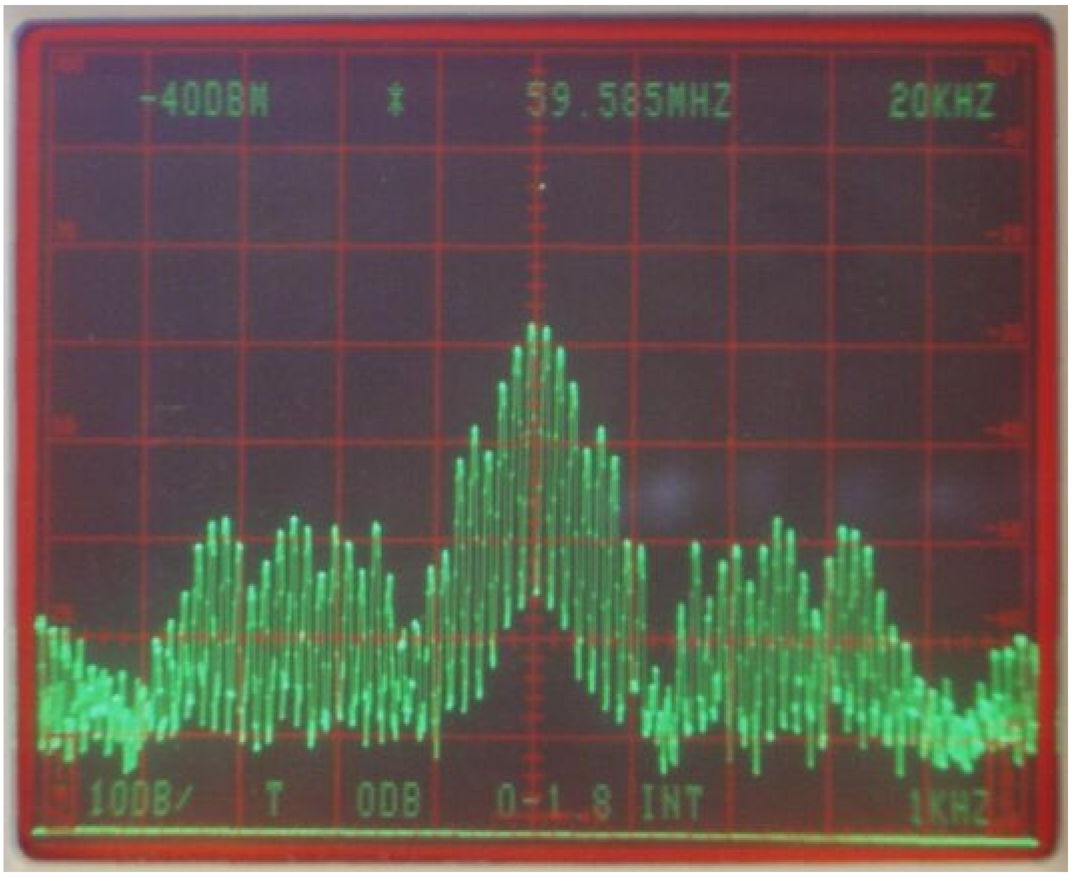


Fig. 1. *The data pins of an A/D converter carry high frequency signals. This spectrum is picked up with a capacitive probe close to the AD6644 in an SDR-14 that samples at 66.667 MHz. Two test signals at 7.0800 and 7.0828 are sent into the antenna input.*

At the same time as the signal shown in figure 1 was picked up, Linrad was used to process the output from the SDR-14. Figure 2 shows the screen dump from Linrad. The similarity of the spectra is obvious. The level of the IM3 signal is fairly independent of the signal level. In figure 2 it corresponds to about -85 dBm for an input signal level of about -35 dBm. Saturation occurs at about -7 dBm. If these numbers were used to compute IP3 according to the third order law one would get -10 dBm and +32 dBm respectively. It is obvious that the SDR-14 does not have an IP3. The signal at the IM3 frequency is probably dominated by noise pick-up from the databus.

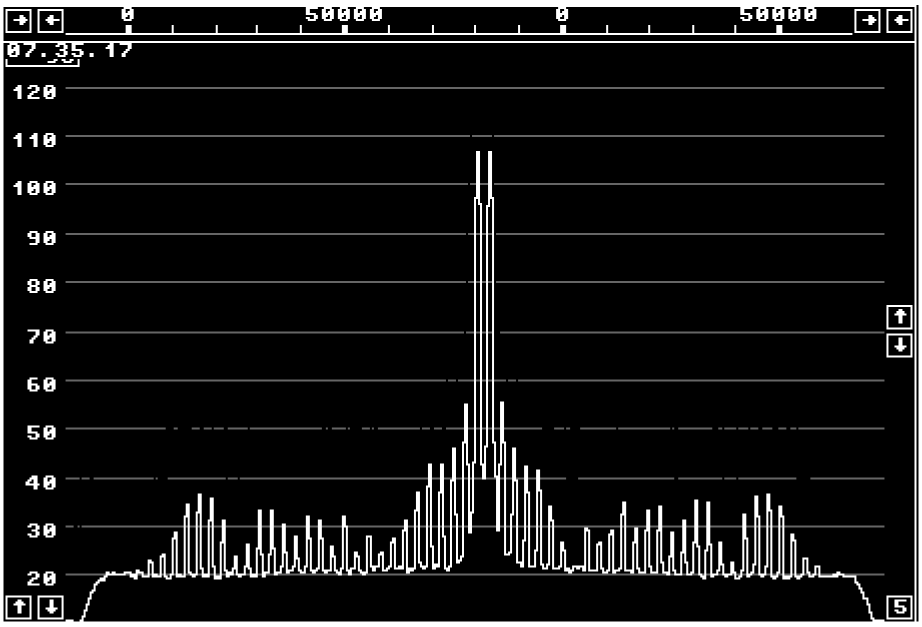


Fig. 2. *The spectrum displayed on the Linrad screen when processing the SDR-14 signal generated while figure 1 was produced.*

Performance of the SDR-14 with real signals from an antenna

To demonstrate the effect of the intermodulation-like interferences in the SDR-14 in real life, two instances Linrad were run simultaneously under X11. One using the SDR-14 hardware, the other using the WSE converters and a Delta 44 soundcard. Both systems were connected to the same antenna through a hybrid, an attenuator and a preselector as illustrated in figure 3.

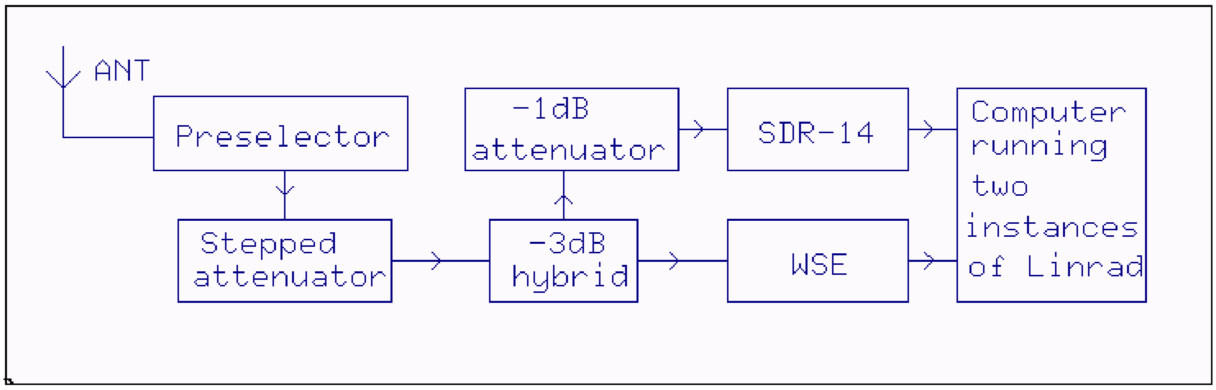
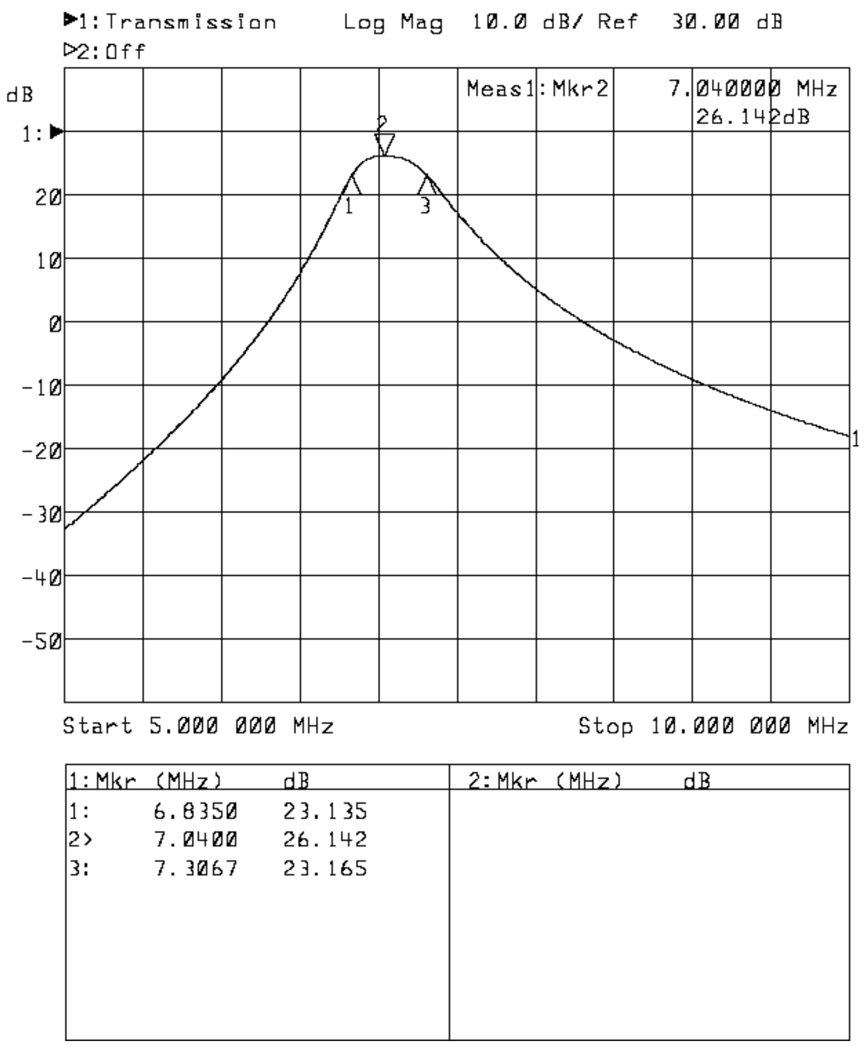


Fig. 3. *Setup to compare an SDR-14 to a WSE converter system.*

The 1 dB attenuator in front of the SDR-14 serves to make the noise figure identical for the two receivers. Both of them have a noise figure of about 17 dB as measured at the input of the hybrid. With zero attenuation in the stepped attenuator the noise figure at the antenna input is 5.5 dB and then the noise floor in both systems is raised by 15 dB. The frequency response of the preselector is shown in figure 4. The 3 dB bandwidth is about 0.5 MHz and the maximum gain is 26 dB. The antenna was a wire about 5 metres above ground with a length of about 10 metres. With zero attenuation the signal level from the antenna, occasionally makes the red light indicating saturation go on for very short times. No adverse effects from these short periods of saturation could be seen on the Linrad screen. A single signal fed into the antenna input turns on the saturation indicator on the SDR-14 at a level of -33 dBm and that is therefore the peak power level that the test antenna occasionally can produce when many BC stations happen to be strong and in phase simultaneously.



```
Meas1:Mkr2    7.040000 MHz
              26.142dB
```

1:Mkr (MHz)		dB	2:Mkr (MHz)	dB
1:	6.8350	23.135		
2>	7.0400	26.142		
3:	7.3067	23.165		

Fig. 4. *Preselector frequency response.*

According to an article by Peter Chadwick, G3RZP, HF Receiver Dynamic Range: How Much Do We Need? QEX May/June 2002, p36-41, the required dynamic range on 40 m is 96 dB. This number is evaluated under the assumption that regular intermodulation is the limiting phenomenon when two signals at -15 dBm give IM3 while the noise floor is at -101 dBm. This happens at 2255 according to table 1 in Peters article.

The summed peak power of two signals at -15 dBm is -9 dBm and in order to accomodate this power the stepped attenuator of figure 3 would have to be set at 24 dB which would degrade the noise figure to about 15 dB. This is 14 dB better than required according to Peters article.

Since saturation is the limiting case rather than intermodulation the margin is smaller at 1715 according to Peters data. Five signals at -15 dBm will give a peak power of -1 dBm occasionally which would call for an attenuator setting of 32 dB with a noise figure of 25 dB. This is only 7 dB below the required noise figure but still a respectable margin.

The screen dumps below show very clearly that the SDR-14 has to be run with strong signals that lift the noise floor above the level of the data-bus interference.

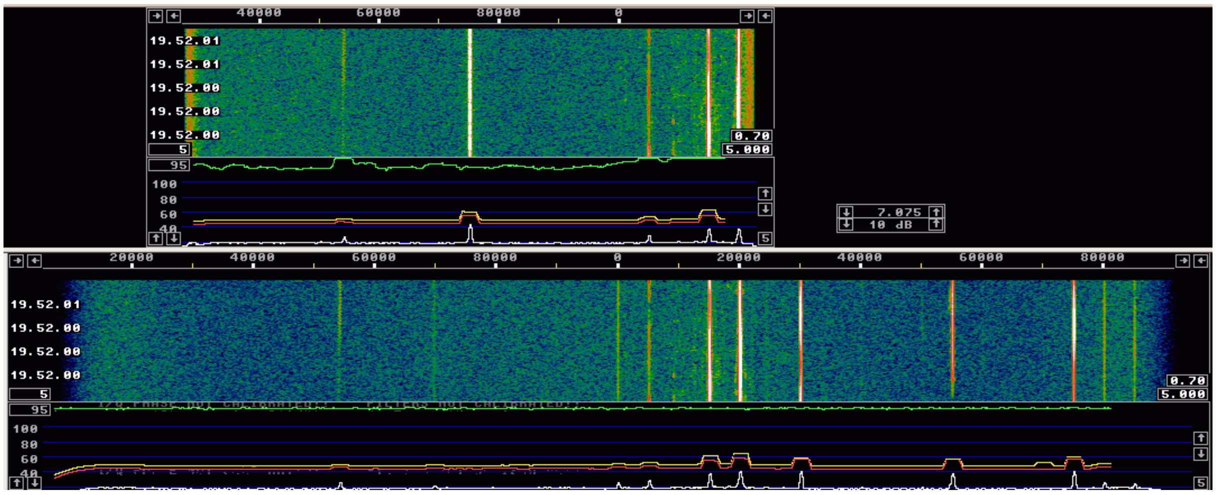


Fig. 5.*WSE and Delta44 on top with a total bandwidth of 96 kHz. SDR-14 below with a total bandwidth of 189.4 kHz. The frequency scales are similar so the SDR-14 waterfall is twice as wide. The attenuation is 80 dB for a NF of 71 dB. The carriers of the AM broadcast stations are visible about 40 dB above the noise floor. The DC component of the Delta44 is at 7.075 MHz while the much weaker DC component of the SDR-14 is at 7.100 MHz. The SDR-14 has a spur of unknown origin at about 7.070MHz. The only signal within the amateur band is at 7.054 MHz. Both systems work without interferences at this very low signal level.*

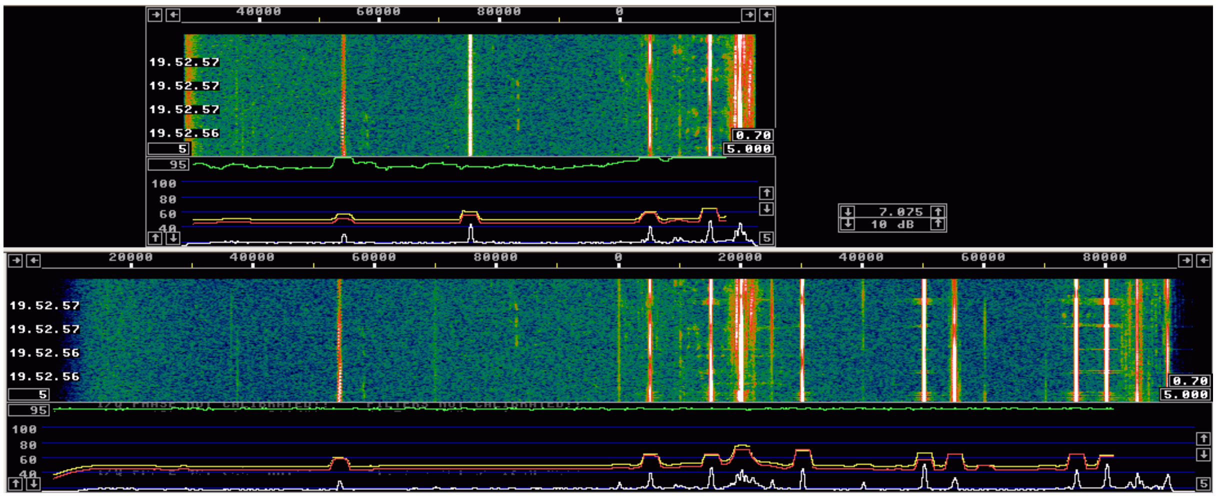


Fig. 6.*The attenuation is 70 dB for a NF of 61 dB. Signals are (of course) 10 dB stronger compared to figure 6.*

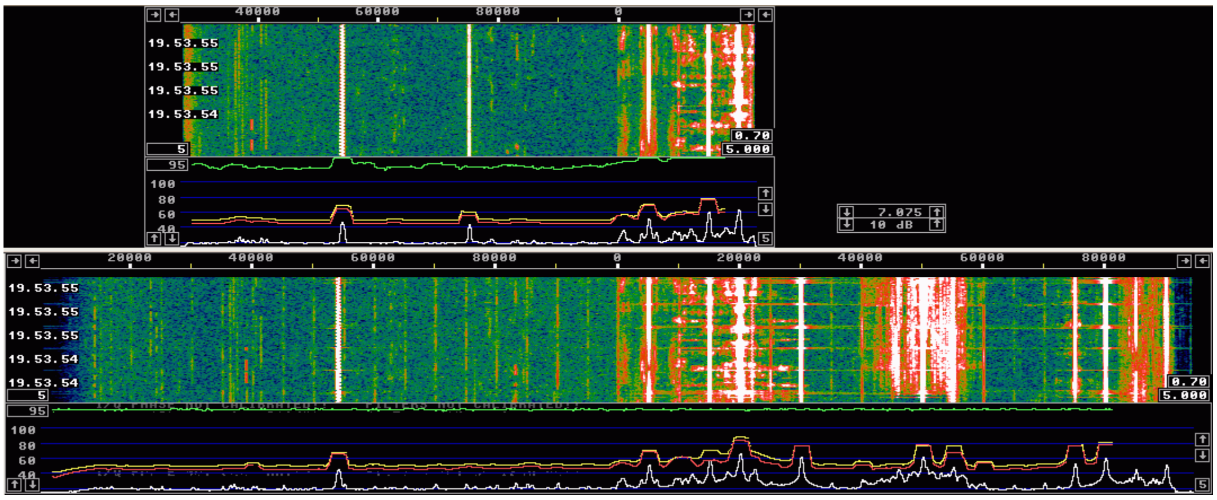


Fig. 7.*The attenuation is 60 dB for a NF of 51 dB. The signals are now strong enough to cause the noise on the data bus of the SDR-14 to be synchronous with 5 kHz, the channel separation between AM broadcast stations. There are false carriers every 5 kHz in the SDR-*

14. The noise floor has also rised slightly because all the AM modulation sidebands also contribute.

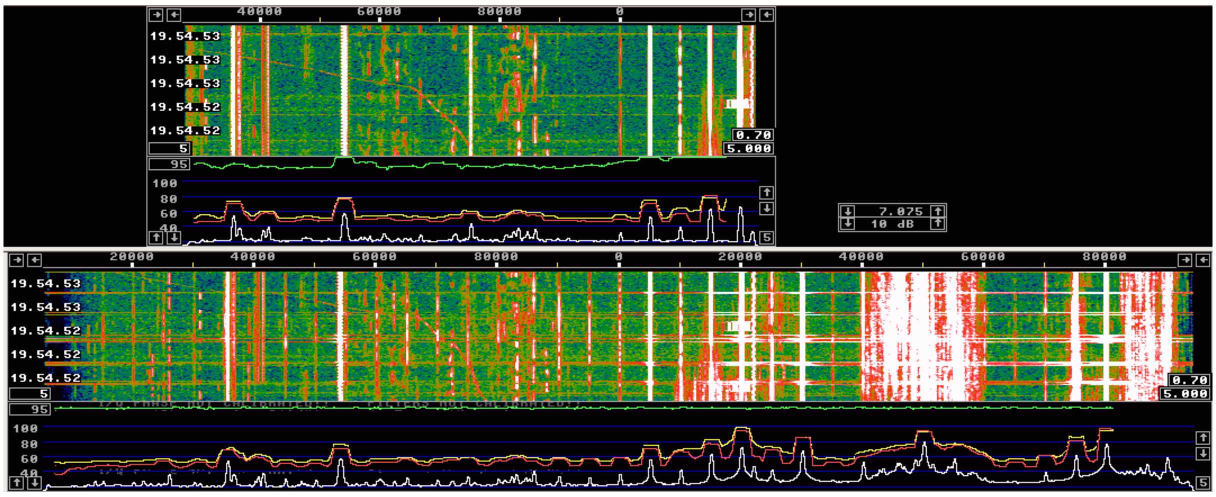


Fig. 8.*The attenuation is 50 dB for a NF of 41 dB. The false carriers at every 5 kHz in the SDR-14 are stronger, one or two more lines on the data bus have became active to produce more interference power. This image was captured too soon after starting Gimp (a Linux program to capture screen dumps) and therefore the waterfalls have not been properly updated. There are many enough signals however to allow a comparision in those time intervals shown in both waterfalls.*

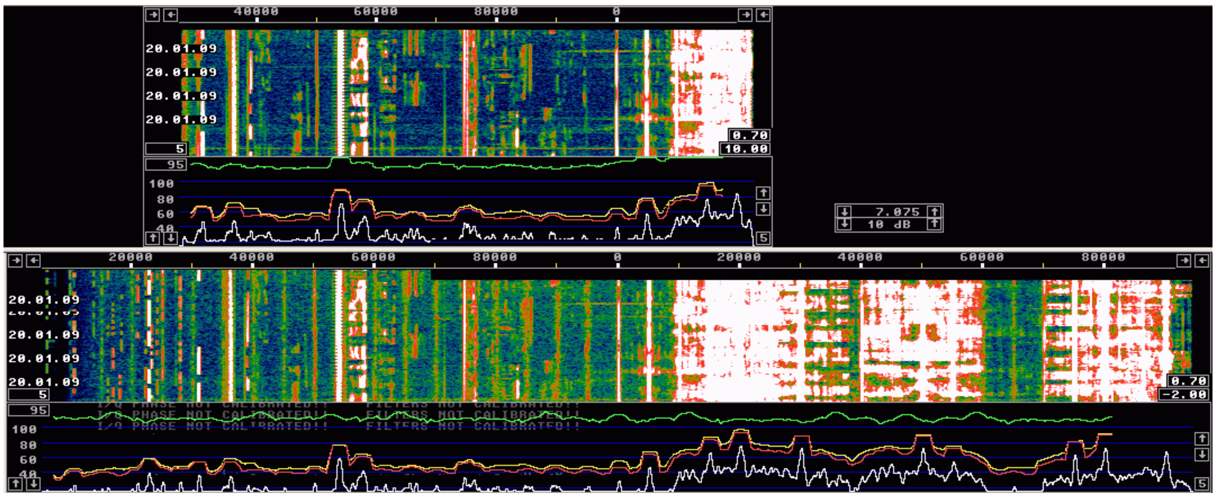


Fig. 9.*The attenuation is 40 dB for a NF of 31 dB. The zero point for the colour scale is shifted by 5 dB. The noise floor has increased slightly due to the noise picked up by the antenna, but it is quite clear that the SDR-14 has a degraded noise floor and that it suffers from false signals caused by the AM broadcast stations. The SDR-14 uses USB to send data to the PC in a 16 bit format. Figures 5 to 8 did not use any output shift (OL_RCF=0) but in order to accomodate the data within 16 bits the output was left-shifted by two bits for this screen dump. To compensate for the reduced system gain the zero point of the Linrad colour scale was changed by 12 dB.*

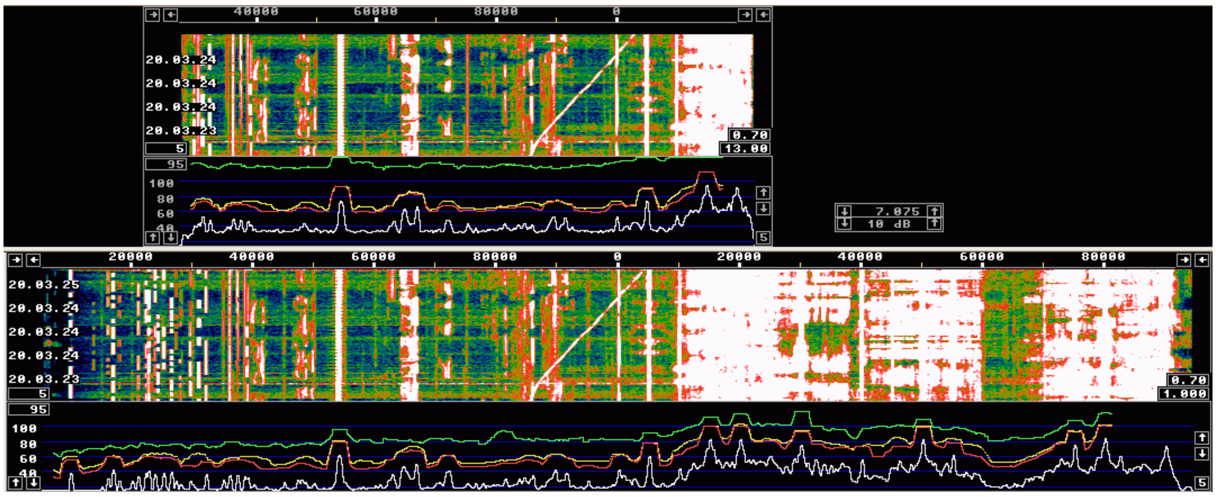


Fig. 10.*The attenuation is 30 dB for a NF of 21 dB. The zero point for the colour scale is shifted by 8 dB compared to figure 9. The noise picked up by the antenna contributes significantly to the noise floor. Only the stronger false signals caused by the AM broadcast stations are visible.*

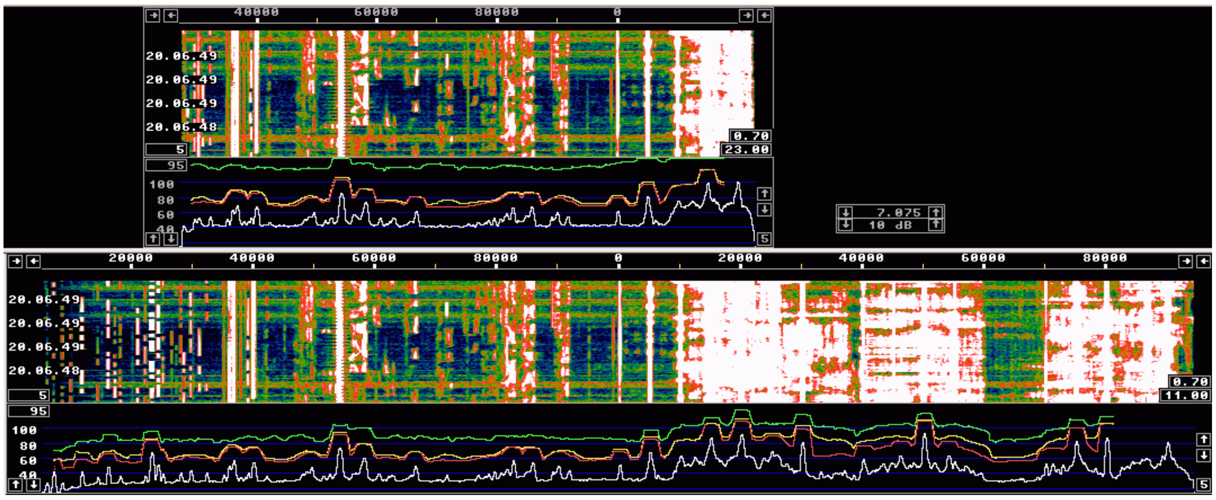


Fig. 11.*The attenuation is 20 dB for a NF of 11 dB. The zero point for the colour scale is shifted by 13 dB compared to figure 9. The noise picked up by the antenna now dominates the noise floor, but the stronger false signals caused by the AM broadcast stations are still visible.*

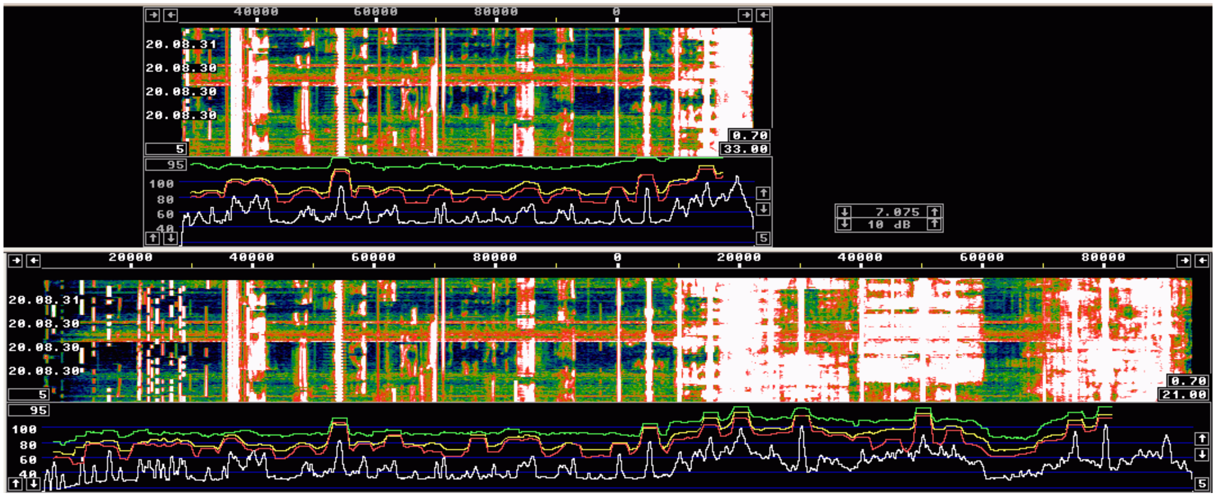


Fig. 12.*The attenuation is 10 dB for a NF of 7 dB. The zero point for the colour scale is shifted by 23 dB compared to figure 9. With the antenna noise placed 10 dB higher than in figure 11, the false signals are now below the noise floor. At this point the SDR-14 has a margin to saturation of about 10 dB.*

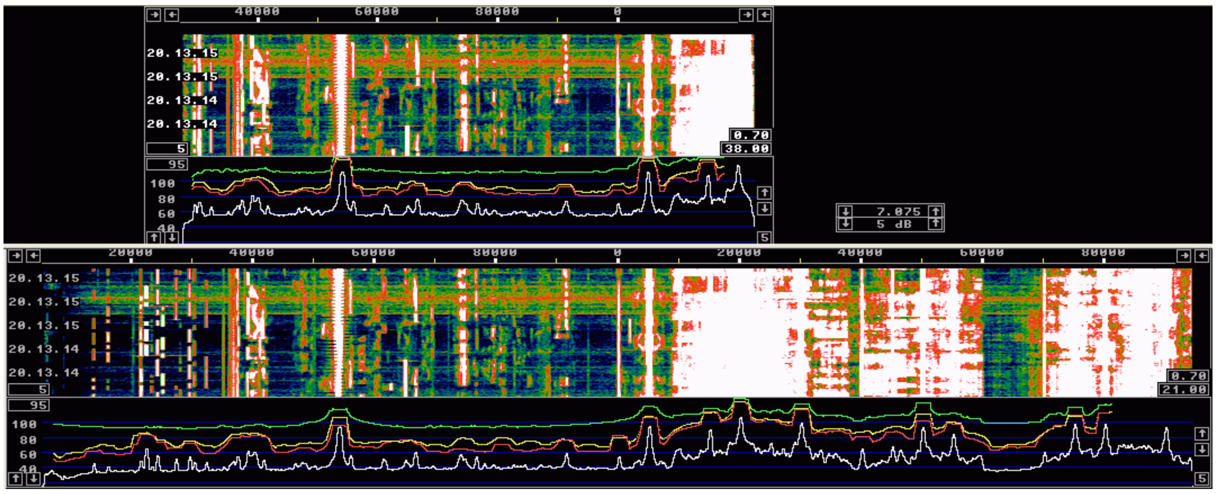


Fig. 13.*No attenuation. The NF is 5.5 dB. The zero point for the colour scale is shifted by 33 dB compared to figure 9. The output of the SDR-14 is shifted by 4 bits here and the WSE RXHFA preamplifier gain is reduced by 5 dB. The AD6644 of the SDR-14 saturates occasionally but the interference generated is not visible. The antenna noise floor is so high that much stronger interferences are needed to be visible.*

Conclusions

Receivers using VHF-sampling A/D converters suffer from interference caused by signal pick-up from the data bus and/or by non-linearities in the digitalisation process. Such interference is essentially independent of the signal levels and it is necessary to put an amplifier in front of a unit such as the SDR-14 to lift the noise floor high enough to make the false signals disappear below the noise.

The need to lift the noise floor by about 25 dB reduces the seemingly excellent dynamic range by lifting the noise floor from -152 dBsat/Hz to -127 dBsat/Hz. This is still quite respectable, not very different as compared to conventional amateur transceivers on HF bands.

On VHF and UHF where one wants an ultra-low noise figure for EME, satellite and weak signal communication, one has to lift the noise floor by about 15 dB anyway to make the mast-mounted preamplifier dominate the noise completely. Some noise from additional amplifiers and converters easily places the noise floor at about -127 dBsat/Hz

In case the dynamic range of 127 dB is inadequate one can remove the false signals by dithering. A quasi-random signal (noise modulated FM outside the interesting frequency band) with an amplitude that spans 50% of the A/D converter transfer function will reduce the point of saturation by 50% or 6 dB. The intermodulation-like interference would be spread out to cause a noise floor increase in the order of 3 dB so the improvement in dynamic range would be about 15 dB compared to removing the interference with a simple amplifier that just lifts the noise floor.

The digital technology develops however. It is possible to add and subtract random noise at various places to improve performance of a VHF sampling A/D converter. Chips like LTC2207/LTC2206 use such randomisation and should have a much better performance than the AD6644.

Besides the AD6644 the SDR-14 contains an AD6620 that reduces the sampling rate from 66.667 MHz to some value that fits a PC computer. The AD6620 does not allow filter that removes all spurs when a wide passband like 200 kHz is sent to the PC. The purpose of the preselector is to remove all resampling spurs and to protect the AD6644 from the needless signal voltages due to signals far away from the amateur band. It is obvious that a narrower preselector would make the dynamic range margin much larger on the 40 m amateur band.

The preselector also has an adverse effect. It removes the noise outside the desired passband that would have helped to randomise the interference by dithering.

Receiver testing

One of the key figures of merit for a receiver is the dynamic range. Generally it is the level above the noise floor undesired signals may have to not cause S/N degradation for a weak desired signal.

Measurements on conventional receivers start from the noise floor and increase the undesired signal until degradation of the desired signal is observed. When testing a digital receiver one should do the measurement the other way around by placing the undesired signal at the saturation limit, then look for the placement of the noise floor that will just mask the interference.

With reference again to the QEX article by Peter Chadwick it is necessary to shift the dynamic range up or down with amplifiers or attenuators on conventional amateur radio receivers. It is the same with digital receivers, one should just use different criteria for how to set the optimum level. An analog receiver should be adjusted for the antenna noise to just lift the noise floor a little, between 6 and 15 dB depending on circumstances. A digital receiver should be set for the highest signal level that does not cause saturation.

Appendix 6

Frequency stability of the output from a software defined receiver.

DRM, Digital Radio Mondiale.
Finding DRM decoders is non-trivial because there are patent rights involved. When using an SDR it could be a good idea to have the DRM decoder in another computer and run the audio through a cable. This way an SDR would behave exactly as any analog receiver modified for DRM.

As it turns out, the cable strategy does not always work because an SDR may add too much phase/frequency modulation to the signal passing through it.

The setup used for the figures on this page is as follows:

Signal source: 10 MHz reference oscillator (TCXO)
RX hardware: Perseus HF receiver connected to a Compaq 6510b.

The audio signal is analyzed on a second computer with a 48kHz soundcard. This computer can decode DRM transmissions and it can also be run with Linrad for spectrum analysis.

Winrad.

Figure 1 shows the frequency vs time with Winrad 1.31. Winrad was started about 2100. The baseband graph shows the output frequency from From 21.35 to 21.57. At that time, something went wrong with Winrad. Presumably due to some other activity by Windows. At about 21.59 Winrad was restarted and one can see how the initial frequency rises by nearly 0.5 Hz during the first 5 minutes to compensate for the initial error in the ratio between input and output.

At 22.15, the window that allows hardware control for the Perseus was moved by about 100 pixels upwards. That caused something to go wrong, after that the output is no longer a narrowband signal. The problem is also visible on the Winrad screen. As long as Winrad has not been disturbed by other tasks for the PC, the DRM decoder works fine in the other computer via an audio cable.

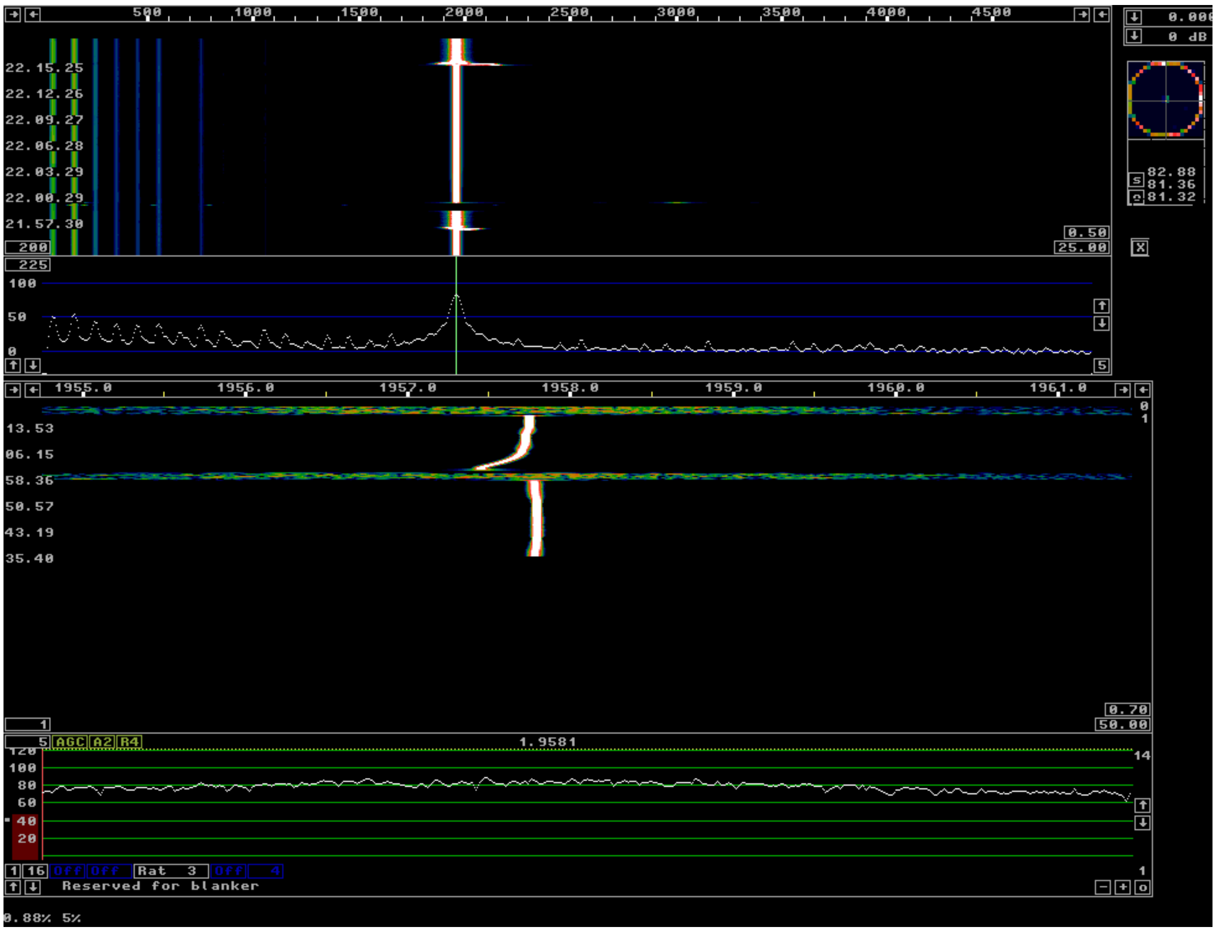


Fig 1.Winrad 1.31.

Linrad.

Figure 2 shows the frequency vs time with Linrad-02.47. Linrad was started at about 22.40. Perseus is set for a sampling rate of 1 MHz. The last few minutes, from about 23.45, an attempt was made to disturm Linrad by running many different programs simultaneously. Acrobat reader(scrolling up and down), Internet explorer (surfing various sites), the disk defragmenter, system monitor and several others. There are occasional glitches in the output, but Linrad recovers immediately. The DRM decoder works fine in the other computer via an audio cable.

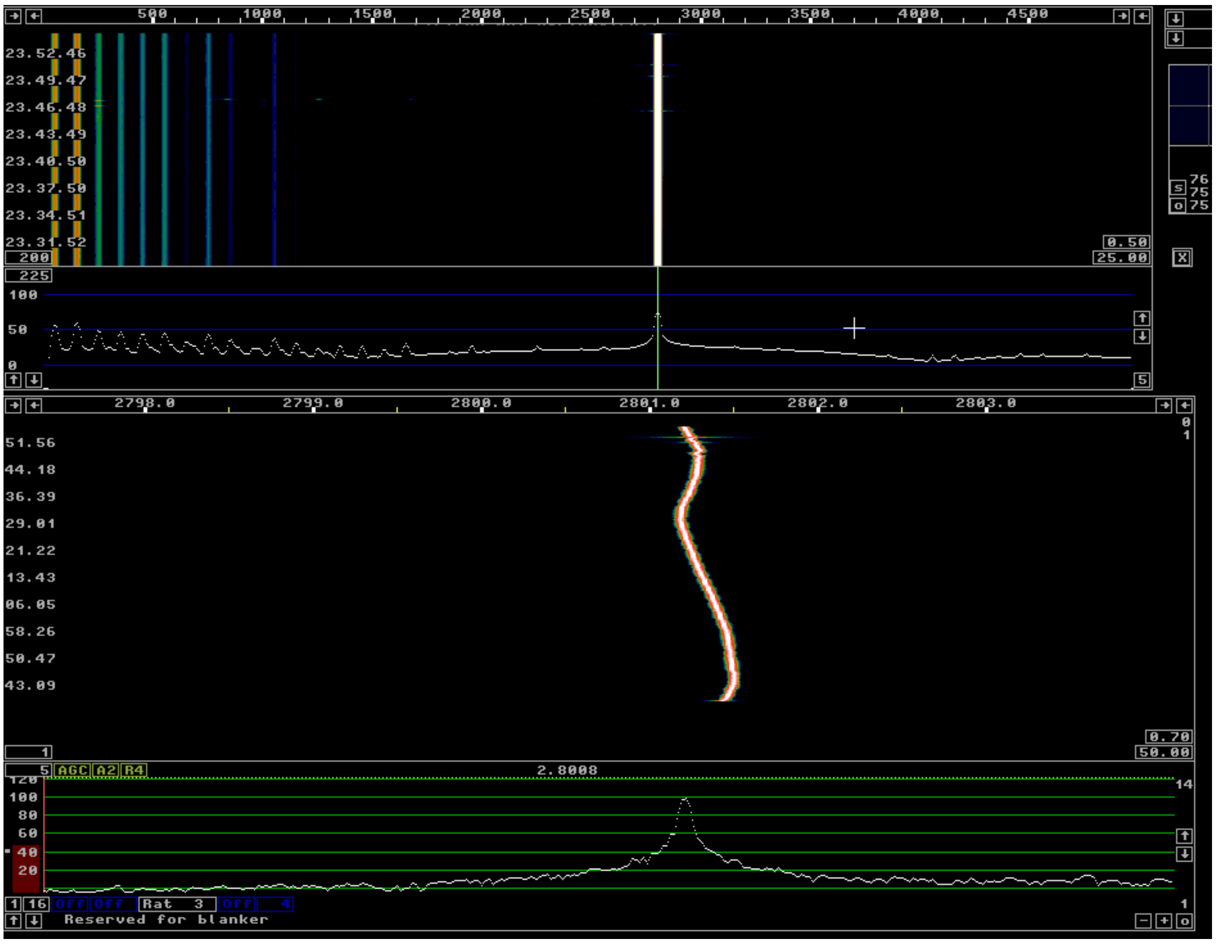


Fig 2.*Linrad-02.47.*

Perseus.exe
Figure 3 shows the frequency vs time with perseus.exe v1.0f. Note that the screen resolution is 16 times smaller as compared to figures 1 and 2. The loudspeaker output is frequency modulated and occupies a bandwidth of about 5 Hz. The DRM decoder does not work at all via the audio cable even though the spectrum as seen by the DRM decoder looks OK.

To receive DRM with perseus.exe one should use a virtual audio device that presents the DRM decoder with a signal that is derived as a non-variable fraction of the Perseus hardware sampling rate.

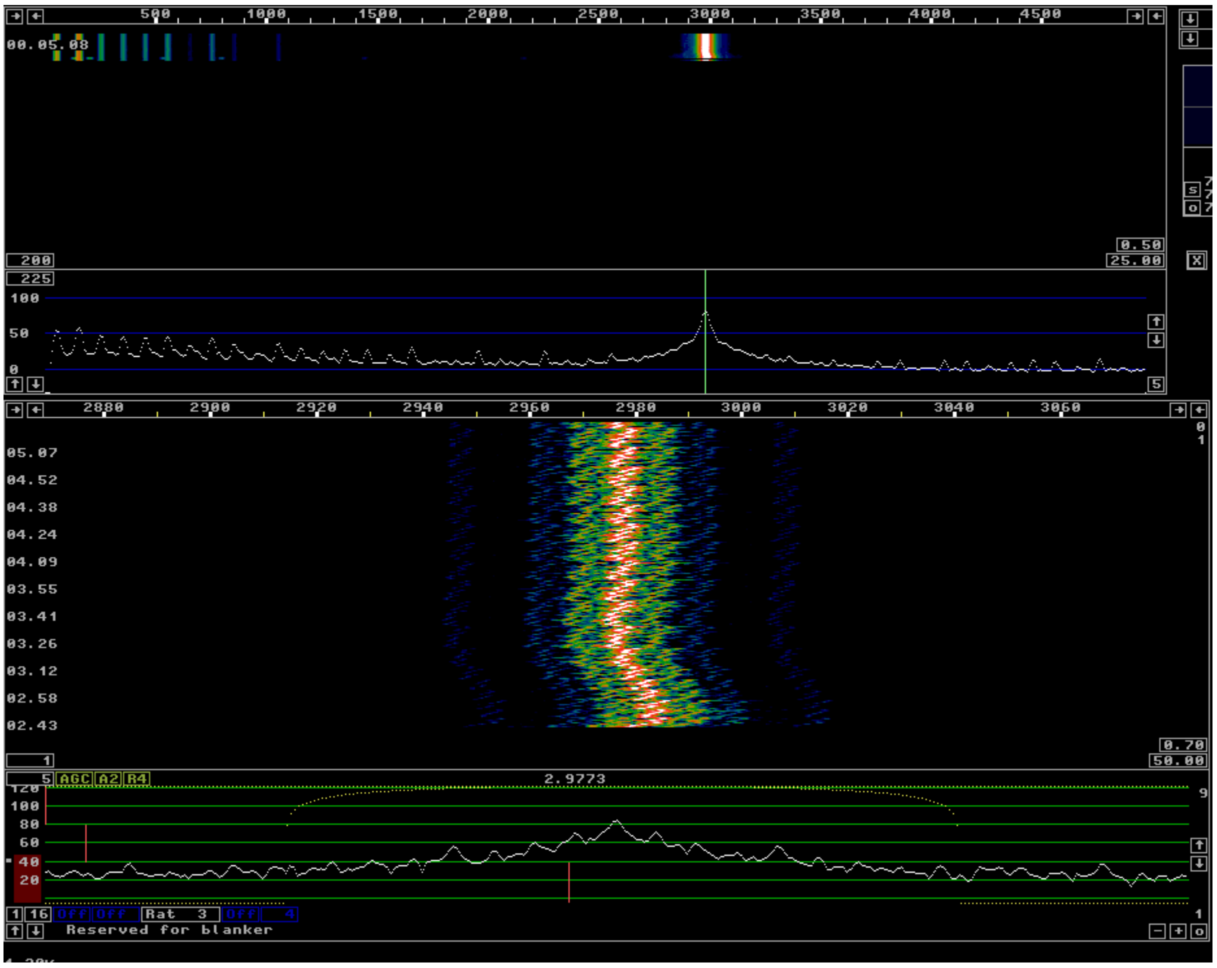


Fig 3.*Perseus.exe v1.0f.*