sequences $r_{xx}(k)$ and $r_{dx}(k)$. The algorithm is basically a recursive gradient (steepest-descent) method that finds the minimum of E and, thus, yields the set of optimum filter coefficients.

We begin with arbitrary choice for the initial values of *h(k)*, say $h_0(k)$. For example, we may begin with $h_0(k)=0,\ 0\le k\le N$. Then, after each new input sample *x(n)* enters the adaptive FIR filter, we compute the corresponding output, say *y(n)*, form the error signal *e(n)=d(n)-y(n)* and update the filter coefficients according to the equation

$$h_n(k) = h_{n-1}(k) + \beta e(n)x(n-k), k = 0,1,\ldots,N-1, n = 1,2,\ldots,$$

where β is called the step size parameter and *x(n-k)* is the sample of the input signal located at the *k*th tap of the filter at time *n*. This is the LMS recursive algorithm for adjusting the filter coefficients adaptively so as to minimise the sum of squared errors E.

Implementation

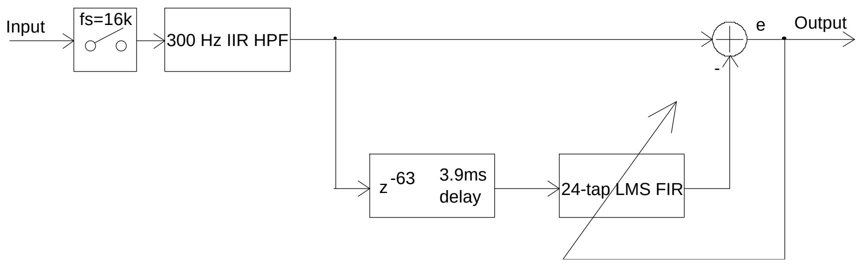

Figure 18: Block diagram of the QRM reductor

The signal flow diagram carrier type noise reductor, QRM reductor, is shown in figure 18. First the signal is fed via high-pass type filter that removes the unwanted hums that does not contribute to the speech signal. This filter is second order IIR-type realisation analog elliptic prototype filter. The filter is realised with one biquad section with proper pre- and post-scalings.

The signal is fed to the delay line and filter adaptation error calculation. Delay line and adaptive FIR-filter are implemented with one ring buffer in order to minimise explicit data movements. Adaptive filter is implemented with LMS coefficient adjustment with simple coefficient decay, i.e. the filter coefficient modification equation is

$$h_n(k) = \gamma h_{n-1}(k) + \beta e(n)x(n-k), k = 0,1,\ldots,N-1, n = 1,2,\ldots,$$

where coefficient decay factor γ controls the forgetting factor of the coefficients. Especially when the convergence parameter β is large, there is a noise build-up effect where hiss components are noticeably amplified. When the input signal goes zero, the coefficients of the standard LMS algorithm does not change, and when there is a lot of noise in the signal, the coefficients will tend to wander aimlessly and may become quite large, increasing the unwanted noise part in the signal. With this decay parameter γ, the LMS algorithm can slowly recover and reset itself over a period of several seconds.

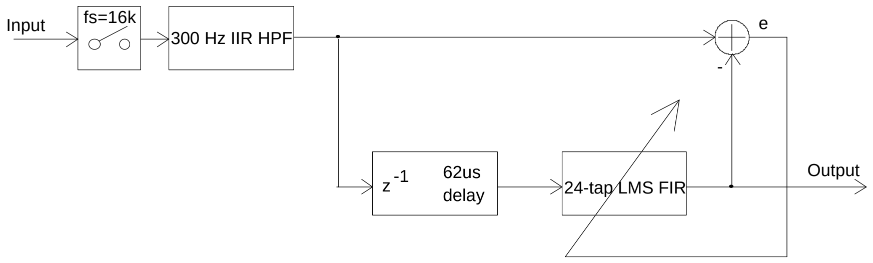

Figure 19: Block diagram of the QRN reductor

Random noise reductor, QRN reductor, is shown in figure 19. In this case, the delay is shorter because we want to reject those signal components that does not correlate, i.e. are pure white noise, and pass components that have small correlation, i.e. speech signal.

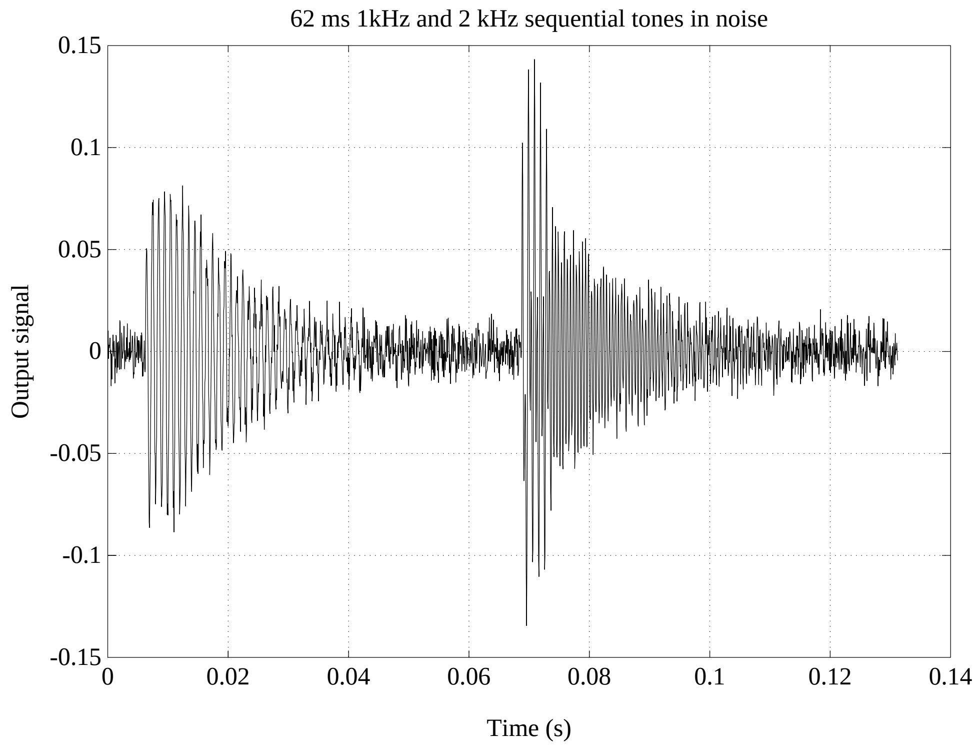

Figure 20: Sequential tone interference

In figure 20 is illustrated the automatic notch filter feature. First 1 kHz signal is fed to the notcher and the frequency is changed to 2 kHz. The LMS filter adapts in 40 ms to the new signal.

Using noise reductors

Bandpass filter programs is `QRMQRN.ASM`, and it must be loaded to the processor memory. This programs filters signals in the left line-input channel, and produces the output on the both left and right channel line outputs.

1200 AFSK modem

Frequency modulation

The simplest amateur packet radio network is based on FM modulated radios and 1200 bit/s Bell 202 telephone standard modems connected directly to the audio ports of those radio transceivers. This Bell 202 modem uses tones of two different frequencies to carry information, this kind of modulation method is called as *frequency shift keying* (FSK). FSK signal can be described as [12,13]

$$s(t,\vec{a}) = \sqrt{\frac{2E}{T}}\cos\left(\omega_c t + \phi(t,\vec{a}) + \phi_0\right),$$

where $0 \le t \le nT$, and E is the signal energy during one bit interval T, ω_c is the carrier frequency and ϕ_0 phase of the carrier at the start of the bit. It is assumed in non-coherent receiver case that ϕ_0 is a random variable that is limited to range $[-\pi, \pi]$. Information bearing phase ϕ can be stated as

$$\phi(t, \vec{a}) = \pi a_i h \frac{t-(i-1)}{T} + \pi \sum_{r=1}^{i-1} a_r h,$$

where $(i-1)T \le t \le iT$ and $\vec{a} = (a_1, a_2, \ldots, a_n)$ is *uncorrelated equal-distributed binary sequence* $(a_i = \pm 1)$ and h is *an modulation index.*

Modulation index determines in a very large extent the spectrum usage of the FSK-signal. In case where *h*=0.5, the resulting spectra of the signal is very narrow and this type of modulation is called as *Minimum Shift Keying* (MSK). When *h*=1.0, spectra is very broad and demodulating this signal is easy, this kind of modulation is called as *Sunde-FSK.*

Frequency modulation is used in data communication systems mainly because [13]:

- Signal demodulation can be done with non-coherent receivers (i.e. there is no need to carrier synchronisation), and thus the receiver structures are very simple and cheap.
- FSK-signalling is quite immune against many non-linearities because FSK-signals are usually constant amplitude signals.
- FSK-signalling is very efficient in using all the available signalling energy to transfer information. Therefore, multi-frequency (*MFSK*) modulation is widely used in cases where the available power levels are small.

Bell 202 FSK modulation format uses two frequencies, 1200 Hz and 2200 Hz to carry information. Spectra of this kind of signal is shown in the figure 21. It is clearly seen from the figure, that the resulting signal easily fits on the audio channel of the FM-modulated radios. This is maybe the reason for the success of this modulation format in amateur community—modems of this kind are very easy to interface to the excisting radios.

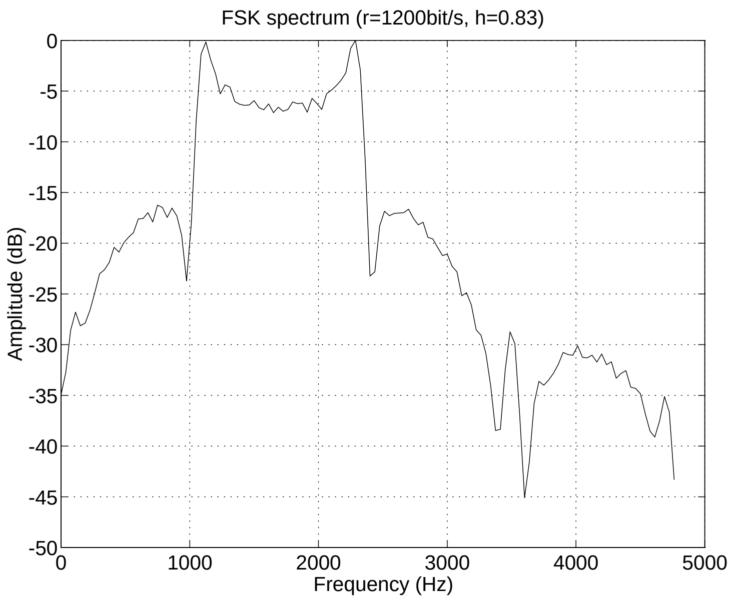

Figure 21: Spectrum of 1200 bit/s Bell 202 FSK-signal

FSK demodulation

One of the most easiest way of demodulating FSK signal is by two narrow bandpass filters tuned to the signalling frequencies and comparing the levels coming from the filters. In this implementation, the frequency uncertainty must be small, because the narrow bandpass filters are on fixed frequencies.

In order to demodulate FSK modulated signal, the frequency of the sinusoidal signal buried in noise must be tracked. If the signal is known to be characterised by some number of parameters that vary only slowly, then the formalism of *Kalman filtering* [10] tells how the incoming, raw measurements of the signal should be processed to produce best parameter estimates as a function of time. For example, in our case the signal is frequency-modulated sine wave, then the slowly varying parameter to be estimated is the instantaneous frequency. It can be shown [23] that the Kalman filter for this case is called a *phase-locked loop* [13].

Phase-locked loop demodulator

We have implemented FSK demodulator using digital phase-locked loop (DPLL) which is frequency locked to the received signal. It is quite easy from this frequency indication to make the decision of the symbol transmitted. The signal block diagram of modem implementation is shown in figure 22. Received signal, *y(t)*, is first filtered in order to reject unwanted noise components and to make the signal complex. *Complex signal* [13] has two components, *real* and *imaginary* part, and therefore can hold **both** the amplitude and phase of the sinusoidal signal at the same time. On the other hand, real signal can have only amplitude component (we cannot say to which direction the signal is going by observing only one point of the sinusoidal signal). Real signal can be converted to complex form with *Hilbert transform* [3]. Hilbert transformer looks the signal for a while and then makes the decision of the phase of the signal.

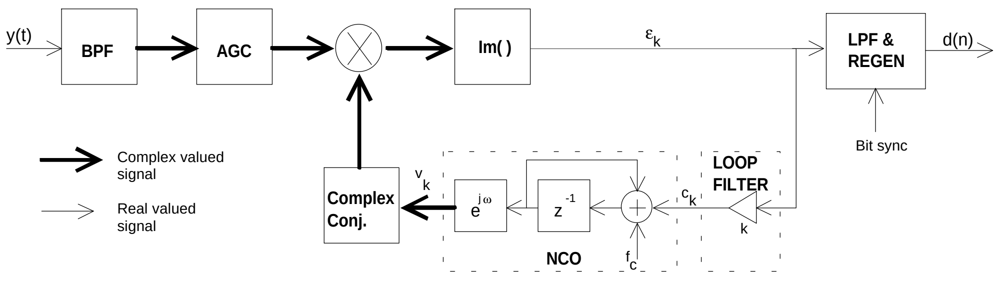

Figure 22: FSK demodulation with digital phase-locked loop

Using complex signals in phase-locked loops is useful, because in this case there is no image frequencies at the output of the phase comparator (mixer). In a theory, the received signal could have first bandpass filtered and then fed to the Hilbert-transformer, but this can be done in easier way. It can be shown, that when we filter the signal with two filters in parallel, the outputs from these filters form a complex pair, i.e. their outputs are in 90° phase difference, if the coefficients of the filters can be stated as

$$\begin{aligned} h_I(n) &= 2h(n)\cos(\omega_c n) \\ h_Q(n) &= 2h(n)\sin(\omega_c n) \end{aligned},$$

where *h(n)* is the impulse response of the baseband filter. This bandpass filter was designed using Hamming windowing method, because the passband should be maximally flat. The design parameters were: 37 taps, cut-off frequency 700 Hz. Coefficients from the SPW's FDS filter design program were multiplied with sine and cosine components at the center frequency (1600 Hz). The resulting amplitude response is shown in figure 23.

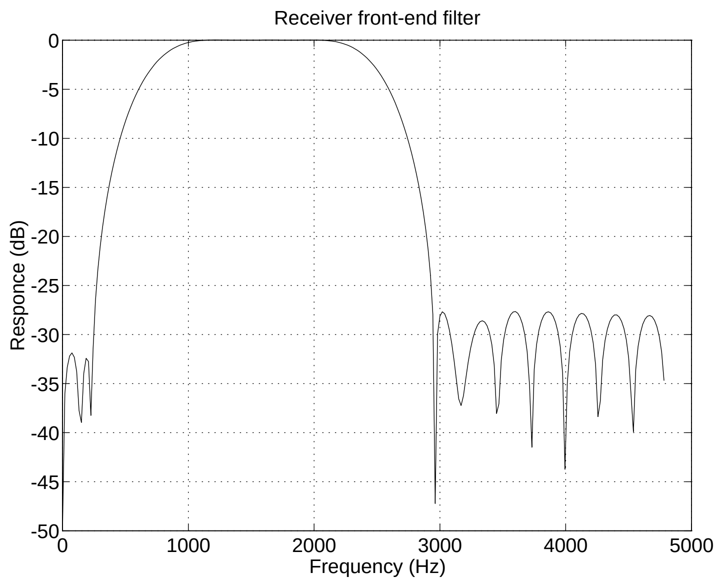

Figure 23: Amplitude response of the input bandpass filter

Phase comparator in the phase-locked loop is an imaginary component of the result of multiplication with complex conjugate of the local oscillator,

$$\Im\left\{A_y e^{j(\omega_y+\theta(t))} A_v e^{-j(\omega_v+\phi(t))}\right\} = A_y A_v \sin\left(\theta(t)-\phi(t)\right).$$

At discrete moment of time *k*, the operation of the phase comparator can be stated as

$$\varepsilon_k = A_y A_v \left(\theta(\tau_k) - \phi(\tau_k)\right)$$

by noting that $\sin x \approx x$ when the *x* is small. Phase comparator is therefore linear when the phase differences are small, which makes the analysis of the loop easy.

The local oscillator, *numerically controlled oscillator* (NCO), can be described as

$$\theta_{k+1} - \theta_k = c_k,$$

where *c* is the control signal of the oscillator. Taking z-transform of the NCO operating equation, we get the phase response of the *first-order phase-locked loop*

$$\Phi(z) = \frac{1}{z-1} C(z) = \frac{K}{z-1} E(z),$$

where *K* is the loop gain and *E(z)* is the z-transform of the phase difference signal. By z-transforming also the phase-difference equation and combining it with the previous equation, we get

$$\frac{\Phi(z)}{\Theta(z)} = \frac{K}{K+z-1} = \frac{Kz^{-1}}{1+(K-1)z^{-1}}.$$

There is one pole at $z = 1-K$, thus the loop is stabile when the loop gain is $0 < K < 2$. First-order DPLL is therefore **always** stable when there is loop gain. Phase response of the first-order loop is shown on the figure 24 with three different loop gain values. As it is seen from the figure, the loop behaves as a low-pass filter for phase changes, if $0 < K < 1$.

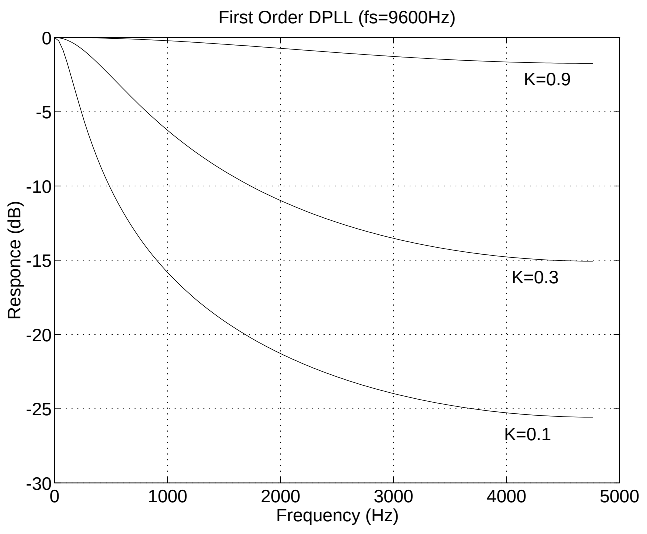

Figure 24: Phase response of the first-order DPLL with varying loop gains

Locking range of the loop is defined as

$$|\omega_o| \leq \frac{\pi}{T}|L(1)|,$$

where $L(\cdot)$ is the transform function of the loop filter. In the first-order phase locked-loop, the loop filter is only a gain function. From the locking range equation we get another limiting factor for the loop gain

$$2\frac{f_o}{f_s} \leq K.$$

Loop gain must therefore be in the range

$$2\frac{f_o}{f_s} \leq K \leq 1.$$

Loop gain can be also stated as in traditional analog form (V/Hz) because

$$K_{\mathrm{Hz}} = 2\pi f_s K.$$

Loop gain K=0.39 is thus 600 Hz/V when the sampling frequency is 9600 Hz. Output of the loop is shown in figure 25.

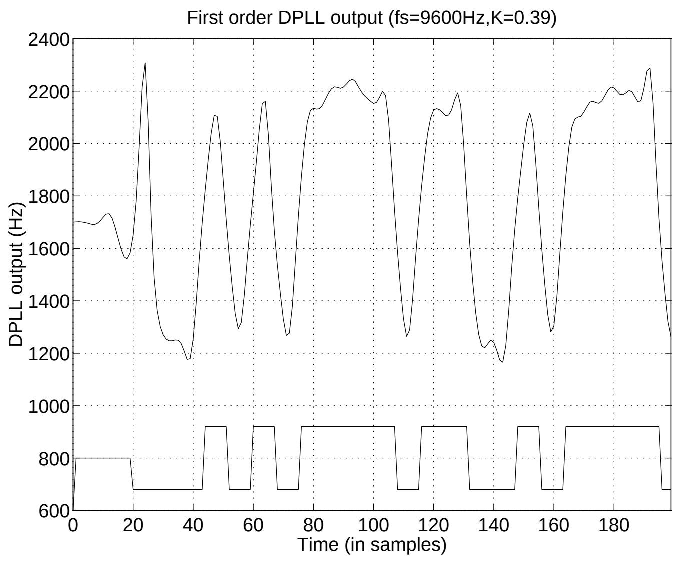

Figure 25: 1200 bit/s Bell 202 FSK-signal and demodulated result

Automatic gain controller

From the phase comparator equations we can be noted that the output of the phase-comparator is dependent of the amplitudes of the input signals. Therefore the input signal levels must be kept constant. This can be accomplished with a device called *automatic gain controller* (AGC). Gain control can be of course done in a straightforward way by first measuring the maximum level of the input signal and then dividing the input signal with this maximum level. In DSP56001 processor there are no special one-cycle divide operation. Therefore some other way of doing the amplitude stabilization must be found. One method is shown in figure 26. In this system we use a feedback-loop to stabilize the maximum amplitude of the input signal. The amplitude of the signal is measured by searching the maximum level of the input signal during 4×8=32 samples. This is subtracted from the set-point, and the result is integrated. Integration constant is set in such a way that the -3dB point in response is at 5 Hz. Thus the AGC does not react to the temporary noise spikes in the input signal.

Digital frequency synthesizer

Phase-locked loop needs a local controllable oscillator whose amplitude is constant but the frequency is variable. The most straightforward way to do this is with *digital frequency synthesizer* [28]. Values of the function sinθ are precalculated in memory, and the phase θ is a linear, increasing function of time, so

$$x(n) = \sin\left(\phi + \Delta\frac{2\pi}{N}n\right),$$

where ϕ is starting phase, Δ is the phase angle velocity and *N* is the size of sinewave table. From the equation it can be clearly seen that the frequencies to be synthesized must be multiples of 2π/N. If the difference between desired frequencies is very small, the sinewave table must be very large. The problem can be circumvented by *interpolating* the sample points between two successive wavetable samples [29].

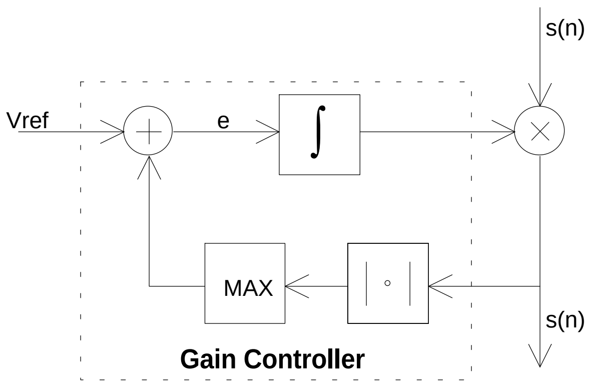

Figure 26: Automatic Gain Controller

Symbol synchronization

In order to make the decision of the received symbol at the optimal time, we must somehow synchronize to the transmitted symbol flow. Normally the data flow has a spectral component at half the data rate. When we direct the data signal through non-linearity, e.g. absolute value function, we get spectral peak at the data rate. If this signal is filtered with a narrow bandpass filter, we get clean synchronized timing pulse for decision making. Symbol synchronization circuitry is shown in figure 27.

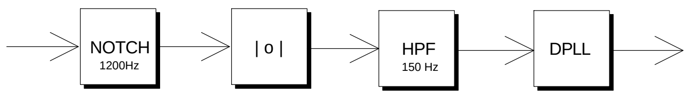

Figure 27: Symbol synchronization system

Phase-locked loop can be used as an bandpass filtering device. When the phase output from the phase-locked loop goes over 2π, it is time to make a decision of the received symbol. Phase-locked loop maintains its synchronism even when the signal momentarily disappears.

Carrier detect

Carrier detection is based on average error power from the decision device. When the decision error power is low it is assumed that there are ongoing data transmissions. Decision error power is filtered heavily, and the level detection is made to decide if the carrier detect condition is met.

Modem operation

Modem filter program is `FSK.ASM`, and it must be loaded to the processor memory using command `dl -g fsk`. 1200 bit/s FSK modem wants the input signal to be in the left line-input channel, and produces the output on the left channel line output. The input level to the modem is about $1V_{pp}$. Using KISS serial interface, some modem parameters can be changed on the fly, see table 3 on the page 25.

9600 bit/s G3RUH modem

The original 9600 bit/s G3RUH modem was designed by James Miller. It provides 9600 bit/s data rate with simple FM radios [44]. G3RUH modem is connected directly to the modulator and discriminator of the radio, and to the modem disconnect pins on the TNC.

G3RUH modem is actually a *baseband* modem, where the filtered digital signal is fed directly to the frequency modulator. The result is two-level frequency shift keying, 2-FSK. Frequency deviation is only 3 kHz, in order to fit the signal to the bandwidth of FM-radio's IF-filters. Signal from the modem is amplitude modulated *PAM-signal* (Pulse Amplitude Modulation).

PAM signal

In PAM-signalling, a sequence of time-translates of a basic pulse is amplitude-modulated by a sequence of data symbols. Such signals can be expressed by [12,13]

$$s(t) = \sum_k a_k p(t - kD),$$

where the modulating amplitude a_k represents the *k*th symbol in the message sequence, so the amplitude belongs to a set of *M* discrete values. The index *k* ranges from -∞ to +∞. *D* is the duration of single data symbol. The unmodulated pulse *p(t)* may be rectangular or some other shape, subject to the conditions

$$p(t) = \begin{cases} 0, \, t = 0 \\ 1, \, t = \pm D, \pm 2D, \ldots \end{cases}$$

This condition ensures that it is possible to recover the message by sampling *x(t)* periodically at $t = KD,\, K = 0, \pm 1, \pm 2, \ldots$, since

$$x(KD) = \sum_k a_k p(KD - kD) = a_K$$

The rectangular pulse satisfies the previous equation if $t \le D$, as does any time limited pulse with *p(t)=0* for $|t| \ge D/2$.

Pulse shaping filters

In the real communication systems the transmission path distorts the pulse shape in some way, so in receiver we have a pulse as follows

$$y(t) = \sum_k a_k \tilde{p}(t - t_d - kD) + n(t_i),$$

where t_d is the transmission delay, $\tilde{p}$ stands for the pulse shape with transmission distortion and $n(t_i)$ represents some Gaussian noise.

Recovering the digital message from *y(t)* is the task of the demodulator. An auxiliary synchronization signal may help the regeneration process by identifying the optimum sampling times

$$t_K = KD + t_d$$

If $\tilde{p}(0) = 1$ then

$$y(t_K) = a_K + \sum_{k \ne K} a_k \tilde{p}(KD - kD) + n(t_K),$$

whose first term is the desired information. The last term is the noise contamination at t_K, while the middle term represents cross talk or spillover from other pulses of the signal—a phenomenon called *intersymbol interference* (ISI).

One can easily assure oneself that the effect of $n(t_K)$ can be reduced by lowpass filtering the signal[7]. On the other hand, lowpass filtering increases ISI. Consequently, some compromises must be made considering ISI, bandwidth and signalling rate.

[7]Assuming here white-noise.

G3RUH modem uses *raised cosine filter* [12,13] to shape the pulses to match the limitations of the transmission link [43]. Its frequency responce is

$$P(f)=\begin{cases}1,\ 0\le f\le 5/16R\\ \text{cosine shape},\ 5/16\le f\le 11/16R\\ 0,\ 11/16R\le f\le\infty\end{cases},$$

where *R=1/D* is the bit rate (bit/s). Usually raised cosine type filter is specified by its *cutoff frequency* f_c and *rolloff parameter* α. The cutoff frequency is naturally half of the bit rate R/2, and the rolloff factor is this case can be obtained by resolving the formula

$$\begin{cases}\frac{5}{6}=(1-p)\frac{1}{2}\\ \frac{11}{16}=(1+p)\frac{1}{2}\end{cases}.$$

In the G3RUH case, the filter parameters are $f_c=\frac{R}{2}=4800\text{Hz}$ and $\alpha=\frac{6}{16}=0.375$. Transmit pulse shaping filter used in this G3RUH modem implementation is shown in figure 28.

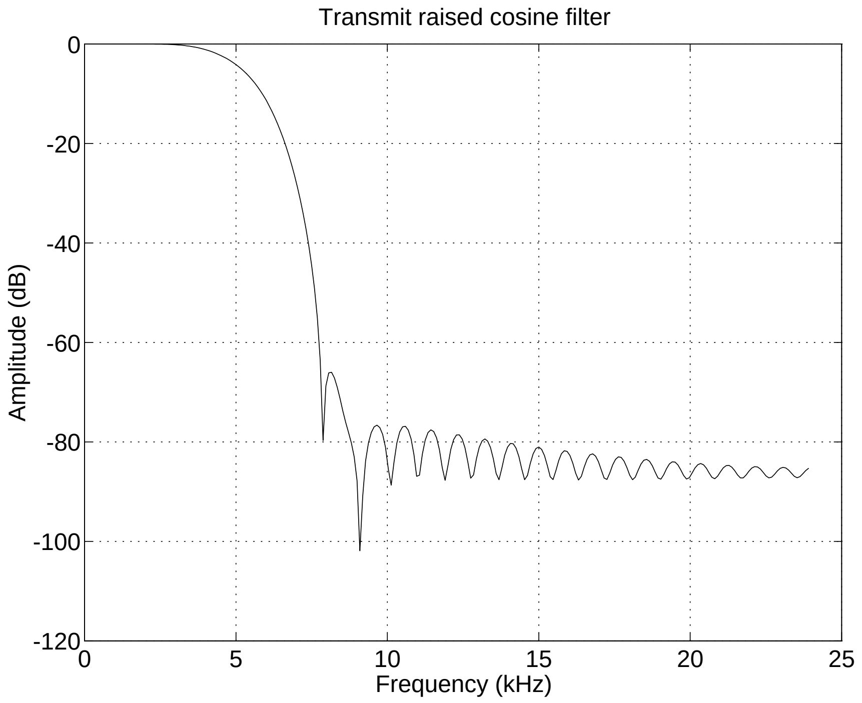

Figure 28: 9600 bit/s G3RUH raised cosine filter responce.

PAM demodulation

PAM demodulation is performed simply by sampling the receiver filter output periodically at the right times. The performance of the modem can be little improved by adding a special shaping filter somewhere on the transmission path adapted to the used transmission path. In G3RUH modem this filter was implemented on the transmitter with a novel way to simplify the hardware implementation, but it therefore requires that the transmitter must be adapted to the receiver which is difficult in multipoint links. Because continuously altering filters are easy to implement with DSP techniques, it is easy to implement the shaping filter on the receiver side and to make it automatically adaptive to get better match with different transmitters.

In this implementation a *Fractional Spaced Equalizer* (FSE) [12,13] is used where the filter taps are spaced a fraction of the symbol interval apart (in this case there is a filter tap for every half of the transmitted symbol). FSE can effectively compensate more severe delay distortion and deal with amplitude distortion with less noise enhancement than conventional equalizer. The adaptation algorithm used is LMS algorithm which is described in the previous QRM and QRN reductor section. Signal flow-diagram of the 9600 bit/s G3RUH modem is shown in figure 29.

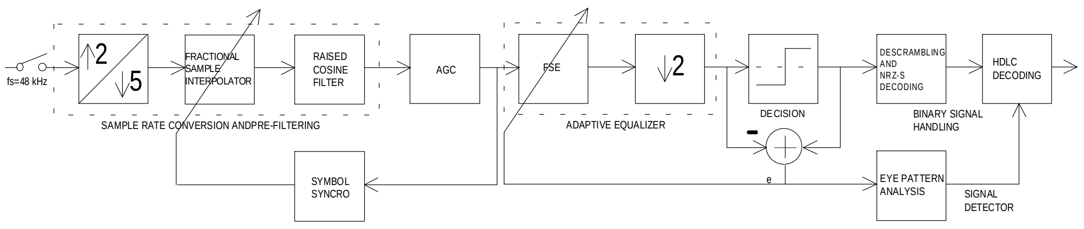

Figure 29: PAM demodulation with fixed-rate sampling and adaptive equalizer

Fixed rate sampling

Because DSP CARD 4 uses fixed sampling rate sigma-delta A/D converter, the time to make a decision cannot be altered by adjusting the sampling moment of the A/D converter. Also the fixed sampling rates of this sigma-delta A/D converter can cause problems, because we want to have the input signal sampled at the rate of 19200 Hz and this sampling rate is not available from the A/D converter.

The sampling rate of the already sampled signal can be altered using a special signal processing technique called *multirate signal processing* [4,26]. We can easily change the sampling rate to a fraction *I/D* by first *interpolating* the signal (by inserting *I* zeroes after every sample) and then filtering the signal in order to remove image frequencies and after filtering, *decimating* the signal (by taking only every *D*th sample). Using a fraction 2/5, we get 48 kHz sampling rate altered to 19.2 kHz.

The sampling moment can be altered after the sampling by interpolating new samples between two successive samples [14]. This technique can be used to adjust the instant a decision is made. This interpolation can be made using *sinc interpolation*. The impulse responce of this kind of filter is a appropriately delayed version of sinc-pulse[8], described by the equation

$$h_i(n) = \mathrm{sinc}\big(\pi(n - d - N/2)\big),$$

where *d* is the desired delay (-0.5 – 0.5 samples) and *N* is the lenght of the filter. This kind of interpolator look like an ordinary FIR-filter (it is actually a FIR-filter with a sinc-type impulse responce).

By convolving interpolation filter and raised cosine filter together, we get a single FIR filter which is easy to implement. This combined filter can be efficiently implemented using a *polyphase* filter structure [4], figure 30. In polyphase filter, the coefficients of the filter are time variant, e.g. they are changing all the time. By adjusting which coefficient set is used, the sampling time can be changed.

[8] $\mathrm{sinc}(x) = \dfrac{\sin(x)}{x}$

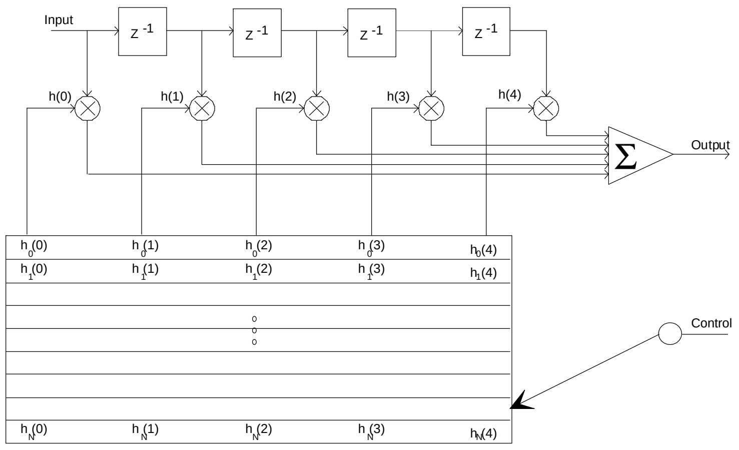
Input
z -1
z -1
z -1
z -1
h(0)
h(1)
h(2)
h(3)
h(4)
Σ
Output
Control
h0(0) h0(1) h0(2) h0(3) h0(4)
h1(0) h1(1) h1(2) h1(3) h1(4)
hN(0) hN(1) hN(2) hN(3) hN(4)

Figure 30: Polyphase filter structure

Symbol synchronization

Symbol synchronization or timing recovery is one of the most critical receiver functions in synchronous communication systems. The receiver clock must be continuously adjusted in its frequency and phase to optimize the sampling instants of the received data signal and to compensate for frequency drifts between the oscillators used in the transmitter and receiver clock circuits. The timing information is usually derived from the data signal itself and is base on some meaningful optimization criterion which determines the steady-state location of the timing instants. A crude distinction can be made between three different kinds of methods [42]

1. The *threshold crossings* of the received baseband data signal are compared with the sampling phase. A correction of the sampling phase is initiated as a result of this comparision.
2. *Signals derivate* at the sampling instants is correlated with estimated data to produce the updating information required for the timing loop.
3. A *spectral line* at the clock frequency if filtered out by a narrowband loop. Since such lines are not ordinarily encountered in systems, some nonlinear processing of the signal is used to generate such lines.

In this 9600 bit/s G3RUH implementation, we use the first method, mainly because then the realization becomes very easy. The symbol synchronization is a special closed-loop system that searches the minimum of ε (timing error) and adjusts correspondingly the sampling system. The timing error is formed by a special function

$$\tilde{T}(kT,\tilde{\varepsilon}) = y(kT,\hat{\varepsilon})\left[D\left(y(kT,\hat{\varepsilon}+\Delta\varepsilon)\right) - D\left(y(kT,\hat{\varepsilon}-\Delta\varepsilon)\right)\right]$$

where *y(·)* is the signal after receiving filter and *D(·)* is the decision operation. When the sampling occurs too late $\tilde{T}(kT,\hat{\varepsilon})$ is positive, and when the sampling occurs too early $\tilde{T}(kT,\hat{\varepsilon})$ is negative. The given error sequence is filtered by a linear random walk filter to filter out the random and synchronizer self noise. The random walk filter is simply an accumulator, and when the accumulator reaches some predefined point, the sampling time is advanced or retarded.

Carrier detect

Carrier detect is implemented by determining the eye pattern opening. That condition is checked by filtering the decision point and zero crossing point values and calculating their difference. When the difference is over some predetermined threshold, the DCD condition is met.

Modem operation

Modem filter program is `G3RUH.ASM`, and it must be loaded to the processor memory using command `dl -g fsk`. 9600 bit/s G3RUH modem wants the input signal to be in the left line-input channel, and produces signal on the left channel line output. The input level to the modem is about $1V_{pp}$. Using KISS serial interface, some modem parameters can be changed on the fly, see table 3 on the page 25.

Using the DSP CARD 4

Normally the DSP CARD 4 is connected to the host computer that controls the operation of the DSP CARD 4. There are mainly two different tasks to be performed:

1. Controlling the programs that are to be executed in the DSP CARD 4
2. Communicating with the selected application program

Task 2 is very application dependent and the instructions for that are given with the particular application. Task 1 things are fully on the behalf of Alef Null consortium. There are two programs[9] for the host computer for controlling the programs residing and executing in the DSP CARD 4: `DLIB` for the EPROM library maintaining and `DL` for program downloading.

Library Manager

The available commands for the library manager are:

```
DSP CARD 4 ROM library maintainer (Apr 08 1995)
usage: dlib -<command>[numarg] <rom_image> [<load_image>] [comment]
    -c<numarg> <rom_image> <load_image> [comment] - replace/add
                                                    a new image
    -d<numarg> <rom_image>                        - delete image
    -b         <rom_image> <load_image> [comment] - replace/add
                                                    boot image
    -p<numarg> <rom_image>                        - set autoboot
                                                    program
    -l         <rom_image>                        - show rom_image
                                                    status
```

Every EPROM library must have a special boot program in it (often this is LEONID monitor program), this can be set by the `-b` option. For example making a new EPROM, first we add the boot program by the command

```
DLIB -b newrom leonid AutoLoad
```

Now we have created a new file `newrom.bin` that contains the boot program (`leonid.lod`) and it is labelled by a text `AutoLoad`. The next thing to do is to add application programs to the newly created EPROM image. This can be done with the `-c<numarg>` replace/add command as following

[9]These programs are written entirely with C language for the PC environment, but they are quite easily (we hope, sic) modifiable to other platform, e.g. MacIntosh® computers.

```
      DLIB -c1 newrom sertst SerialTest
```

This adds a linker output file `sertst.lod` to the `newrom.bin` image file and gives a text label `SerialTest` to it. Other programs can be added in the same way, for example

```
      DLIB -c2 newrom lpc 2400 LPC
      DLIB -c7 newrom fft FFT
```

It is possible to set up things in a such way that when the DSP CARD 4 boots up, it starts automatically one program from the EPROM library if no external commands are given in a limited time. This can be made with `-p<numarg>` set autoboot command as follows

```
      DLIB -p1 newrom
```

The listing of the EPROM library can be obtained with the `-l` show romimage status command. The output with `-l` option from the EPROM library manager after the operations we have performed above is

```
      DSP CARD 4 ROM library maintainer (Apr 08 1995)
      ROM: AutoLoad        09.11.1992

       1*  SerialTest   22 09.11.1992
       2   2400 LPC    180 12.11.1992
       7   FFT          22 09.11.1992
```

From this listing it can be seen a comment describing the EPROM library, date of the monitor program (used for version determination) and the application programs. The lengths of application programs are shown in bytes. This lenght shown is the actual lenght in EPROM, lenght in the processors memory when loaded is a little smaller (2%–5% smaller) because the information for the loader is stripped off. An asterisk after the program number marks the program to be started immediately after reset.

Downloader

The available commands for the serial line loader are:

```
DSP CARD 4 program downloader (Apr 08 1995)
usage: dl -<command>[numarg] [<rom_image>|<load_image>]
    -f         <rom_image>  - program FLASH EPROM
    -c<numarg>              - change program
    -r         <rom_image>  - read FLASH EPROM
    -g         <load_image> - load RAM and go
    -x                      - reset DSP CARD 4
    -p<numarg>              - set current port
```

There are currently only three commands available: `-g` load and go, `-p` set port and -x reset commands. Set port command can be used to the desired port number (1 for COM1, 2 for COM2) in the host computer to be used for the serial communication. Reset command can be given to reset the DSP CARD 4 and query the version number of the LEONID monitor program. If the command

```
      DL -p2 -x
```

is given, then the reset command is written to the port COM2 and the responce may be the following

```
      Leonid monitor version: 08.04.1995
```

Now we know that the communication between the host computer and DSP CARD 4 is ok and the version date of the monitor program.

Load and go command is usefull to download assembler output files (.LOD files) to the DSP CARD 4. For example the loading of QPSK modem to the DSP CARD 4 can be performed with the command

```
      DL -g qpsk
```

This command resets the DSP CARD 4 and downloads the given program `qpsk.lod` and after loading starts the execution.

If the connection to the DSP CARD 4 does not work, or the connection on the later time somehow breaks, the `DL` tries three times before it ends the connection and gives the error report

```
        No response from the DSP CARD 4
```

Data transfer is checked using CRC checkword. If an error is detected, `DL` will automatically retry the same packet. If there are three errors in succession, `DL` will respond with a message

```
        Bad CRC
```

and the downloading is terminated.

Construction and testing of the DSP CARD 4

The DSP CARD 4 was designed using PADS-PCB CAD software on a four-layer printed circuit board. Four layers are absolutely necessary to ensure good grounding for DSP chip and the memories. Furthermore it simplifies the routing and decreases the logic noise both outside of the board (RFI) and also at the codec.

Some facts about bees and flowers

Warning

Soldering the board is quite straightforward except of two components: the DSP chip itself and the codec. Both of these chips are surface mounted devices (SMD). The codec is in a PLCC package, which can be soldered quite easily on the contrary to the DSP, which is in a ceramic quad flat pack (CQFP). Because of these chips, constructing the DSP CARD 4 is not considered to be a first tutorial to soldering.

Motivation

Selecting these case styles without sockets seems to be a good compromise where good grounding and cheap price are most important aspects. CQFP (and PQFP in the near future) is the cheapest package for the DSP. As can be seen from the routing between the memories and the DSP, it is also very well suited for four layer boards where all signals can be routed on one side. Using sockets and PGA package would have increased the total cost about 30%.

Winding the transformer

The transformer has two windings, the primary uses diam. 0.5 mm and secondary 0.3 mm wire. These diameters are not very critical, the wire should be thick enough to minimise losses and thin enough to fit on the toroid. The number of turns both in primary and secondary is 25 if Amidon FT50-61 toroid is used.

The small switcher generates a lot of harmonics. To put the most of those harmonics in the load, the transformer must be have wideband capabilities. The coupling between primary and secondary isn't 100%. Practically you experience this as not ideal coupling as leakage inductance. The leakage inductance introduce series impedance in the transformer. When there isn't enough damping in the circuit, it's ringing on the fast switching. Ringing generate a lot of RFI and can destroy semiconductors.

The 2 transformer windings start both near one of the long sides of the printed circuit board and run parallel the short sides of the board. Glue after test the transformer with hot glue to the printed circuit board.

A good transformer for the DSP CARD 4 can be wound by winding the primary and the secondary at the same time. Take the two wires (primary and secondary) and twist them together. One twist in 1.5 cm is ok. Wind the transformer in a single layer. Start winding at the half the wire! It's better not to use transformer wire when you want a good isolation between the primary and secondary. Isolated wire wrap wire is much better. The best isolation is kynar. Wire wrap wire is 0.3 mm and can be used for the primary and the secondary. Use different colours for primary and secondary!. The number of turns isn't very critical, but don't divert more then 2 windings from the design. The length of a transformer winding is about 50 cm long. When you start with 2 wires of 70 cm you have some spare wire length. Strip the wires so that you haven't more then 1 cm slack in the wire after mounting. Remove the rest of the wires after soldering the transformer in the printed circuit board.

Soldering the DSP

If you have access to a SMD soldering station and SMD soldering paste, please use them, it is surely easier. But if you don’t (as we didn’t), read on! It is best to solder the DSP first, to avoid difficulties from other components being on the way. Use quite a wide tip (2.5 - 3 mm) and a good soldering iron with thermostat.

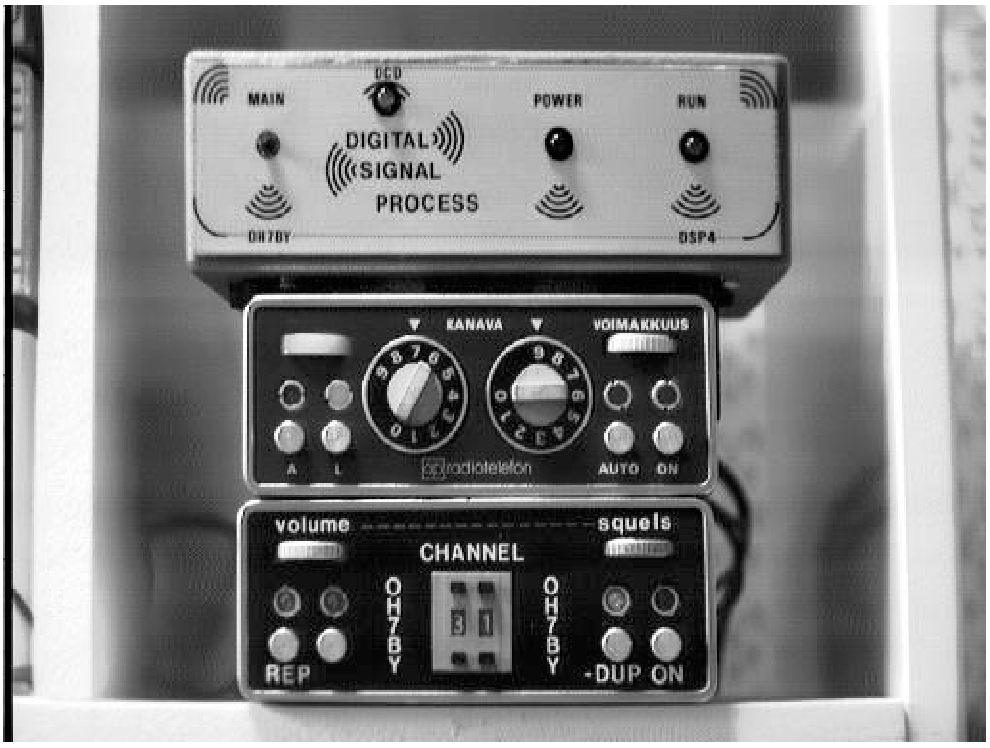

Figure 31: OH7BY's Satellite station with DSP CARD 4. At the top is DSP in a box, below is 2m transmitter and the lowest box is a 70 cm receiver. Both radios have previously been auto radiophones.