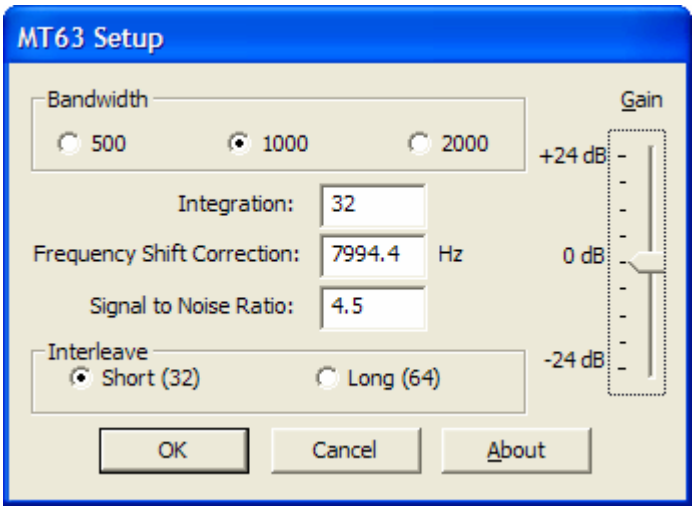
HamHub™ MT63 Interface

HamHub™ Designed by
TIMEWAVE TECHNOLOGY INC.
1025 Selby Ave, Suite 101, St Paul, MN 55104-6533
Phone: (651) 489-5080 ● Email: sales@timewave.com

The MT63 Interface is a wrapper for Pawel Jalocha's UNIX MT63 command line program. Most of this is taken from his documentation with specifics for the Interface inserted.

Mt63lx was released with a GNU public license. The HamHub™ interface was created with permission from Pawel Jalocha.

The MT63 works as described in the original documentation. There is a setup dialog used in place of the command line parameters:



If using PK-Term, this is listed as "Soundcard Setup" on the File menu.

Notes:

BANDWIDTH – This should probably be left set at 1000. See Page 3 below.
INTEGRATION – This should be left at the default of 32. See Page 5 below.
FREQUENCY SHIFT CORRECTION – This is new to the graphical interface. It is a frequency shift for sound card distortion. It is similar to the –R command line option. See Page 6 below.
SIGNAL TO NOISE RATIO – This is the correction factor for signal to noise. It is not available from the command line. A smaller number (around 4 or 5) will allow more data but will have more errors, whereas a larger number (around 9 or 10) will be cleaner, but will not pick up weak signals.
INTERLEAVE – This is the character interspacing. Most common is 32. See Page 4 below.
GAIN – This is digital gain added to the front end for decoding.

Notes:

1. The soundcards on PCs are NOT very accurate. MT63 is VERY frequency sensitive. The soundcard "Frequency Shift Correction" may have to be adjusted up to 150 Hz up or down to get a true 8000 Hz. Text will lose case when the frequency is more than 4 Hz off.
2. Whereas the command line default is a 1 KHz bandwidth, IZ8BLY's MT63 program defaults to 2 KHz. This HH Interface uses the command line defaults.
3. MT63's interleave requires a long period of diddle after the last text is sent to allow all of it to be sent. THIS IS NOT AN ERROR! It is necessary due to the protocols. Similarly, diddle must be received for a while before the first text appears.
4. MT63 uses the ASCII code page.

Original UNIX Documentation

This is the release 0.5 of the MT63 modem for LINUX.
Date: 08-JUL-2004, Author: Pawel Jalocha, SP9VRC, Pawel.Jalocha@cern.ch

Release notes:

0.5 -- This release changes the license of the program to the GNU GPL and fixes a few compiler warnings.

0.4 -- This is another "quick" release, the new feature is the CW identification which can be transmitted below the 64-tone digital signal.

Important notes:

1. To take full advantage of the MT63 modem, the sampling rate of your sound card needs to be either calibrated or dead precise on 8000.0 Hz. For now, the MT63 receiver doesn't tell you the rate mismatch, I may provide such facility in the next software versions.

2. When connecting the audio to your PC/laptop be careful not to damage the computer audio inputs. I suggest you first connect the grounds of the radio and the computer and only then connect or disconnect the audio cables.

The MT63 modem is intended for amateur radio as a conversation (RTTY like) mode where one station transmits and one or more other stations can listen. In short, the modem transmits 64 tones in its 1 kHz bandwidth: the audio range for the tones is 500-1500 Hz. The differential bipolar phase modulation is used to encode 10 bits of information per second on each tone. The user data in the form of 7-bit ASCII characters is encoded as a set of 64-point Walsh functions. The bits are interleaved over 32 symbols (3.2 seconds) to provide resistance against both pulse and frequency selective noise or fading. The character rate equals to the symbols rate thus the modem can transmit 10 7-bit characters per second.

This modem can as well run in two other modes obtained by simple time scaling, the possible modes are summarized here:

Bandwidth	Audio range	Symb. rate	Char. Rate	Interleave/char
500 Hz	500-1000 Hz	5 baud	5 char/s	6.4 or 12.8 sec
1000 Hz	500-1500 Hz	10 baud	10 char/s	3.2 or 6.4 sec
2000 Hz	500-2500 Hz	20 baud	20 char/s	1.6 or 3.2 sec

For each mode the interleave factor can be doubled thus each character becomes spread over twice as long period of time.

The first experiments with this mode were done on the EVM56K DSP evaluation board from Motorola and the package was named MT63ASC.ZIP.
This LINUX implementation is written to be compatible with that package.

The MT63 modem is made for single side band operation. The audio generated by the modem (sound card output) is applied to the SSB modulator. On the receiver side, the output of the SSB demodulator is put into the sound card input. The envelope of the MT63 signal is not constant as in other multi-tone systems - it is rather noise-like.
One must be careful not to overdrive the transmitter.

The receiver of the MT63 is self-tuning and self-synchronizing thus the radio operator is only required to tune into the signal with +/- 100 Hz accuracy for the basic 1000 Hz mode. The modem will tell the actual frequency offset after it is synchronized. The operator should not try to correct this offset unless he is able to tune very slowly his radio

receiver, because the MT63 as a low rate phase modulated system does not like sudden frequency changes.

The package contains the following programs:

mt63tx - MT63 transmitter
mt63rx - MT63 receiver
mt63trx - MT63 receiver and transmiter with a simple terminal
ratecal1 - soundcard rate calibrator
peakrms - utility to measure peak/RMS ratio of an audio file
addnoise - utility to add noise to an audio file

mt63tx and mt63rx can work on a soundcard as well as on raw audio files in signed 16-bit format. For example to transmit the MT63 signal through the /dev/dsp1 you can type:

mt63tx -d1

and to transmit and at same time save the audio onto a file mt63.raw you type:

mt63tx -d1 -smt63.raw

Later you can decode the mt63.raw by typing:

mt63rx -ptm63.raw

To decode and listen to the audio at same time:

mt63rx -d1 -pmt63.raw

The file facility is made for off-line performace evaluation. You can pass the mt63.raw file through off-line HF simulator and then put in back into the mt63rx decoder.

The mt63trx can only work in real-time, you start it like:

mt63trx -d2 (receive standard 1000 Hz mode on /dev/dsp2)

to let it run on /dev/dsp2. To select the modem bandwidth you use the –B option, like this:

mt63trx -d2 -B2000 (receive 2000 Hz mode on /dev/dsp2)

to select the 2000 Hz mode.

To double the interleave factor add -i:

mt63trx -d2 -B2000 -i (receive 2 kHz mode on /dev/dsp2 with double interleave)

The -i and -B options apply as well to the mt63tx/rx commands.
For the mt63rx there is one more option -I which tells the synchronizer integration time.
For example:

```
mt63rx -d -B1000 -i -I64
```

means: receive 1000 Hz mode on /dev/dsp with double interleave, the synchronizer would intergrate over 64 symbols to measure the time and frequency offsets.

Note, that none of the mt63xxx programs takes care for the mixer settings.
It is the operator job to setup proper audio levels on output and input.
I use the "aumix" for this, for example:

```
aumix -d/dev/mixer1 -I
```

allows me to setup interactively the /dev/mixer1.

To help you with setting up the input level the receiver displays the RMS of the input signal and the number of samples above the 3/4 of the full ADC range. Ideally you should setup the mixer so that the RMS is greatest possible but with very few or none overranges.
Some overranges are allowed, they can even help you to cut off part of the pulse noise :-) but you should not allow them to go above 1%.

Operating the terminal of the mt63trx:

Ctrl-T	turns on the transmitter.
Ctrl-R	turns to receive but only after all the keyboard buffer is transmitted.
Ctrl-A	immediate (almost) transmitter shutdown.
Ctrl-D	drop the receiver lock and re-acquire
Ctrl-^	quit the terminal program

While you receive you may already type the text to be transmitted.
It will apear in the lower window and it will be stored in a buffer.
When you press Ctrl-T the transmitter will send the stored text and after it will continue with what you type.
Each character sent will be printed in the upper window.

Driving the radio transmitter:

The signal envelope is not constant (!) - it is rather noise like. It's again the operator who should care not to overdrive the SSB transmitter. If you have the ALC meter it is best to start with the setting which make NO movement of the ALC. If you only have the output power meter I suggest you start with the average power equal to 1/10th of the peak power.

Soundcard sampling rate issue:

The MT63 is a synchronous system and it relies on the sampling rate to be the same at the receiver and the transmitter. At least the sampling rates should not be different by more that 10^-4. MT63 samples at 8000 Hz thus if your card runs at 8000.5 it's probably OK but if it runs at 8005 Hz it's not good ! An extreme example can be my Soundman-16 (PAS-16 clone) which when asked to run 8000 Hz tell me, that it can only do 8008 Hz and it reality it runs at 7910.3 Hz which makes an error of more than 1% - far too much for the MT63 even at infinitely good S/N. My other two cards (DSP-16 and Ensoniq 1371) are more reasonable: they show an error of 0.3 to 0.5 Hz at 8000 Hz sampling.

To measure the sampling rate I have prepared "ratecal1" utility which uses timing signals sent on HF. I use the ones on 4996.0, 9996.0 and 14996.0 kHz but anything else which transmits beeps at precise seconds intervals (or any given intervals) is good enough.

To measure the sampling rate I tune my HF receiver to say 4995.0 USB so the pulses come up as a 1 kHz beeps. I connect the audio to the given soundcard (say /dev/dsp) and I type:

```
ratecal1 -d
```

After 10-20 seconds I can see the measured sampling rate of my card. For more details please read the top of the ratecal1.cc file.

When you know the true sampling rate you can correct for this when working with the MT63 mode with the -R option. Say, you measured that you card at /dev/dsp2 samples at 8010.5 Hz, so you type:

```
mt63trx -d2 -R8010.5
```

I do so with my PAS-16 clone and it works just fine for the MT63 despite the 1% error.

You may happen to have a card which refuses to run at 8000 Hz - but you are not all lost. Say the card at /dev/dsp1 only likes the 9600 Hz sampling and when asked for 9600, it samples at 9605.4 Hz (you measure this with the ratecal1). To run the MT63 receiver you type:

```
mt63rx -d2 -r9600 -R9605.4
```

the mt63rx will ask the sound driver to sample at 9600 and knowing that it really means 9605.4 it will apply the proper rate converter.

Ideally every operator should measure the sampling rate of his soundcard and then correct for this when transmitting and receiving. This way every signal on the air has same absolute timing and thus can be read by anyone.

About the C++ code:

All the code used to compile the modem executables is provided here.
The code is released under the GNU General Public License. For details see the included file COPYING.

IF you want to implement this code in your software, you will most likely need the files mt63.cc mt63.h mt63.dat dsp.cc and dsp.h - look into mt63.h for the description and comments of the classes MT63tx and MT63rx. They are not dependend on the platform as this is pure arithmetic (you will need -lm on linking). When you add the sound I/O and user I/O you have the modem ready.

The code can be compiled under pure MSDOS and I do this with the Borland C++ 3.1. However, as there is no sound support, you can run the mt63tx/rx/trx on raw audio files, which can still be usefull for decoding pre-recorded audio files or off-line experiments.

The mt63xxx programs can read/write audio files in raw, 16-bit, signed data format sampled at 8000 Hz (unless you change the rate with -R). To convert these to .wav or other formats it is probably best to use the "sox" utility, specifying "-s -w -r 8000" before the .raw file produced by the mt63tx for example.

Pawel, SP9VRC.