This is a beta fork of the 64 bit sbitx code with many CW focused features including improved keying, waveforms, decoding and a CW audio peaking filter. More details are below..

A low risk way to test

Create a new directory and use it

Example

```
pi@sbitx~ $ mkdir cwbitx
pi@sbitx~ $ cd cwbitx
pi@sbitx~/cwbitx $ git clone https://github.com/rkayakr/sbitx/tree/cw_all
pi@sbitx~/cwbitx $ cd sbitx
pi@sbitx~/cwbitx/sbitx $ ./build sbitx
  to run this version from within ~/cwbitx/sbitx
pi@sbitx~/cwbitx/sbitx $  ./sbitx
```

If the window title bar displays "v5.01 cw" you are running this version

Your original sbitx app and settings are unchanged. Any contacts you make go into your existing log.
You can launch the original app from the usual thumbnail icon. The window title bar will display "v5.01".

If you don't like the CW version, just remove it. Be careful – there is no way to restore what you remove.

```
pi@sbitx~ $ rm -rf cwbitx
```

Summary of changes:

This version is **v5.01 cw**

Refactored keyer for all cw modes

- broke single large switch-case into seperate functions for each cw mode
- use inline functions to replace repeated lines of code, keeping original logic and function and greatly improving readability
- added reverse paddle (new command is "**cwreverse on|off**')

Data driven waveform (Blackman-Harris cw waveform shaping)

- shape the cw envelope using a pre-calculated array of values applied to leading and trailing edge of every cw symbol
- only a small change from the previous raised-cosine envelope, but we now join the likes of Flex radio, Elecraft and QRPlabs using Blackman-Harris
- sbitx cw spectrum should be a fine-looking signal at least close to center frequency!

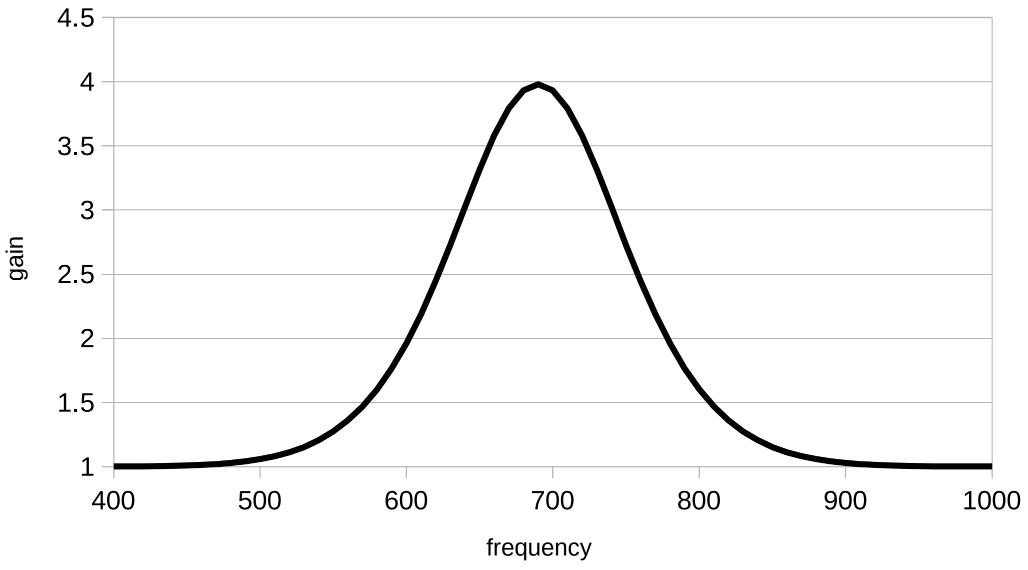
Enhanced cw decoder

- using original design concept for decoder, but almost all functions have been reworked and extended.
- ample frequency bins around signal center to help mis-tuned signals and with signals in noise
- better use strings of detections to improve detection accuracy
- performance may only be marginally better but code is now in better position for further improvement

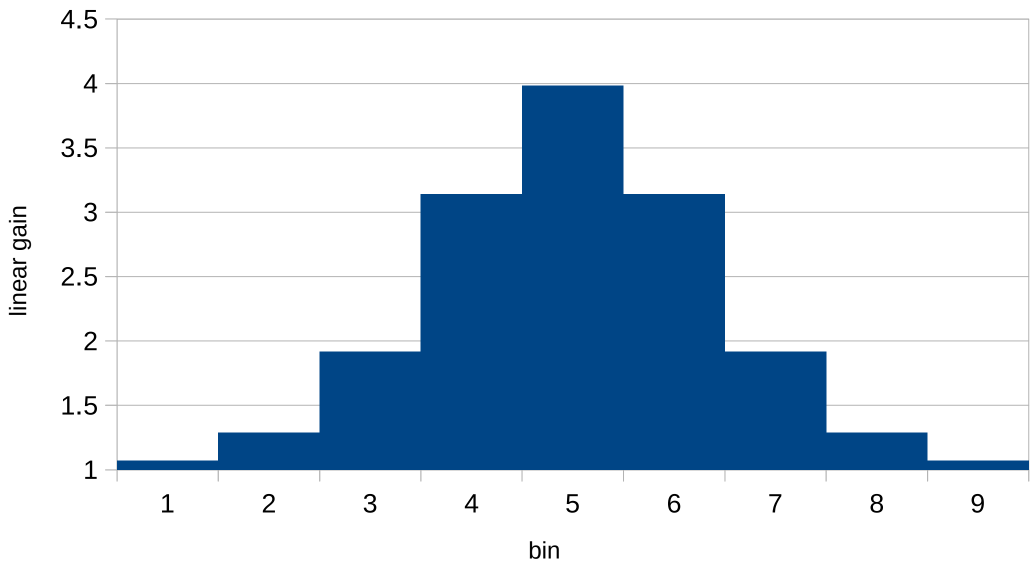
CW apf (Audio Peaking Filter)

This function is the compliment to most filtering that tries to reduce audio levels. In contrast, the apf provides gain defined by a Gaussian curve centered on the CW pitch, that is, the gain rolls off smoothly from the peak. For weak signals this can pop up the signal to make it easier to copy.
It is invoked by the command **apf gain width**
where the gain is in db and the width parameter controls how slowly the gain falls from the peak. Both values are floats. Too high a gain or too narrow a width will induce unpleasant artifacts. I suggest gains from 3 to 9 and a width of about 100. Try it out to find what's bet for you.
To disable the function enter **apf** with no arguments.

Here is a plot of a gain function with gain 6 and width 100. You can see that this curve is around the center at 680 Hz.



The sBitx operates in the frequency domain with bins of frequencies that are about 47 Hz wide. This graph how the sBitx function works. Gains are evaluated at bin centers and applied to a bin. The present sBitx function uses 9 bins, a center and 4 on each side, and has a gain floor of 1.0.


linear gain
bin