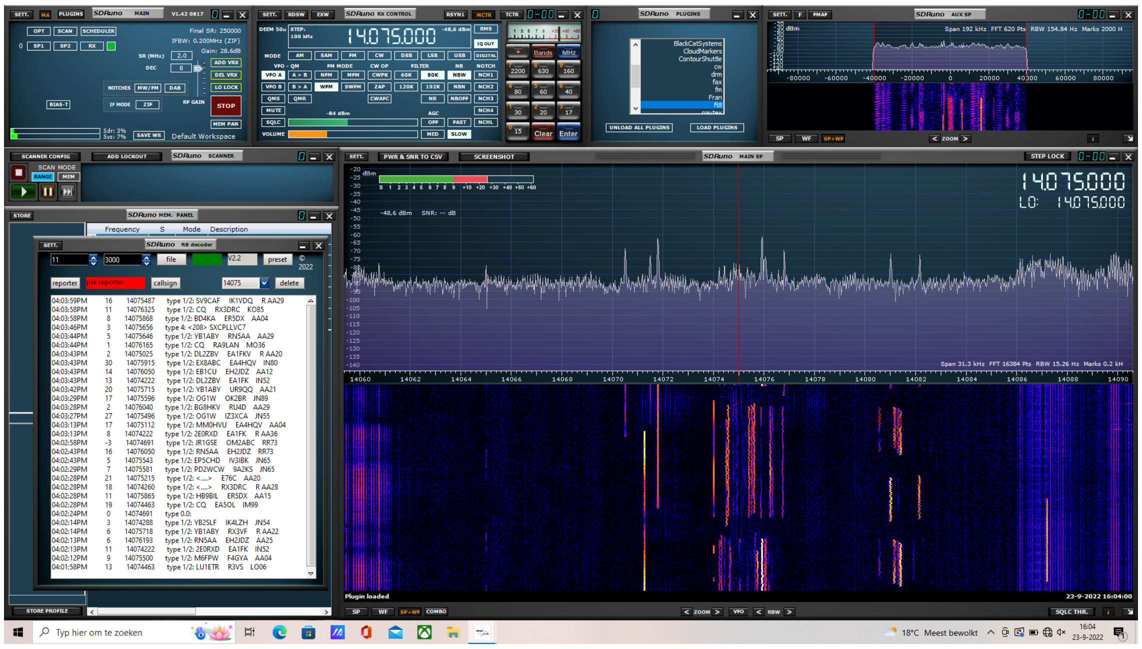
An SDRuno plugin for FT* decoding (V2.2)

Jan van Katwijk
Lazy Chair Computing
The Netherlands
J.vanKatwijk@gmail.com

September 23, 2022



```
04:03:59PM   16   14075487   type 1/2: SV9CAF   IK1VDQ   R AA29
04:03:58PM   11   14076325   type 1/2: CQ   RX3DRC   KO85
04:03:58PM   8    14075868   type 1/2: BD4KA   ER5DX   AA04
04:03:46PM   3    14075656   type 4: <208> SXCPLLVC7
04:03:44PM   5    14075646   type 1/2: YB1ABY   RN5AA   AA29
04:03:44PM   1    14076165   type 1/2: CQ   RA9LAN   MO36
04:03:43PM   2    14075025   type 1/2: DL2ZBV   EA1FKV   R AA20
04:03:43PM   30   14075915   type 1/2: EX8ABC   EA4HQV   IN80
04:03:43PM   14   14076050   type 1/2: EB1CU   EH2JDZ   AA12
04:03:43PM   13   14074222   type 1/2: DL2ZBV   EA1FK   IN52
04:03:42PM   20   14075715   type 1/2: YB1ABY   UR9QQ   AA21
04:03:29PM   17   14075596   type 1/2: OG1W   OK2BR   JN89
04:03:28PM   2    14076040   type 1/2: BG8HKV   RU4D   AA29
04:03:27PM   27   14075496   type 1/2: OG1W   IZ3XCA   JN55
04:03:13PM   17   14075112   type 1/2: MM0HVU   EA4HQV   AA04
04:03:13PM   8    14074222   type 1/2: 2E0RXD   EA1FK   R AA36
04:02:58PM   -3   14074691   type 1/2: JR1GSE   OM2ABC   RR73
04:02:43PM   16   14076050   type 1/2: RN5AA   EH2JDZ   RR73
04:02:43PM   5    14075543   type 1/2: EP5CHD   IV3IBK   JN65
04:02:29PM   7    14075581   type 1/2: PD2WCW   9A2KS   JN65
04:02:28PM   21   14075215   type 1/2: <....>   E76C   AA20
04:02:28PM   18   14074260   type 1/2: <....>   RX3DRC   R AA28
04:02:28PM   11   14075865   type 1/2: HB9BIL   ER5DX   AA15
04:02:28PM   19   14074463   type 1/2: CQ   EA5OL   IM99
04:02:24PM   0    14074691   type 0.0:
04:02:14PM   3    14074288   type 1/2: YB2SLF   IK4LZH   JN54
04:02:14PM   6    14075718   type 1/2: YB1ABY   RX3VF   R AA22
04:02:13PM   6    14076193   type 1/2: RN5AA   EH2JDZ   AA25
04:02:13PM   11   14074222   type 1/2: 2E0RXD   EA1FK   IN52
04:02:12PM   9    14075500   type 1/2: M6FPW   F4GYA   AA04
04:01:58PM   13   14074463   type 1/2: LU1ETR   R3VS   LO06
```

1 Introduction

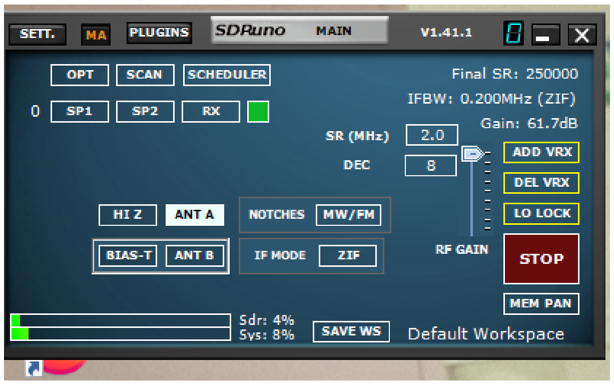
The FT8 plugin is a plugin for decoding FT8 messages and showing the decoded messages. The plugin is experimental in the sense that its development is going on.

The plugin - as a community plugin - can be installed by placing it in the folder for community plugins. Ask SDRplay.com where the folder need to be placed on your system, on my system the folder can be found as subfolder in "Documenten".

2 Selecting a samplerate

The plugin uses the IQOUT option of the SDRuno platform. This output is 192000 samples per second. The plugin itself will reduce the samplerate to 12000 (and filter accordingly). The plugin - when started - will do the selection of the IQOUT port.

The default setting for SDRuno (at least with me) is that the input samplerate is 2 MHz, and the main spectrum display will show a spectrum of the full 2 MHz.



Final SR: 250000
IFBW: 0.200MHz (ZIF)
Gain: 61.7dB
SR (MHz) 2.0
DEC 8
Sdr: 4%
Sys: 8%
Default Workspace

Figure 1: ft8Plugin: uno widget

It is useful to zoom in in the spectrum, after all the spectrum of the region of interest is only a few KHz wide.

3 The FT8 plugin and the reporter map

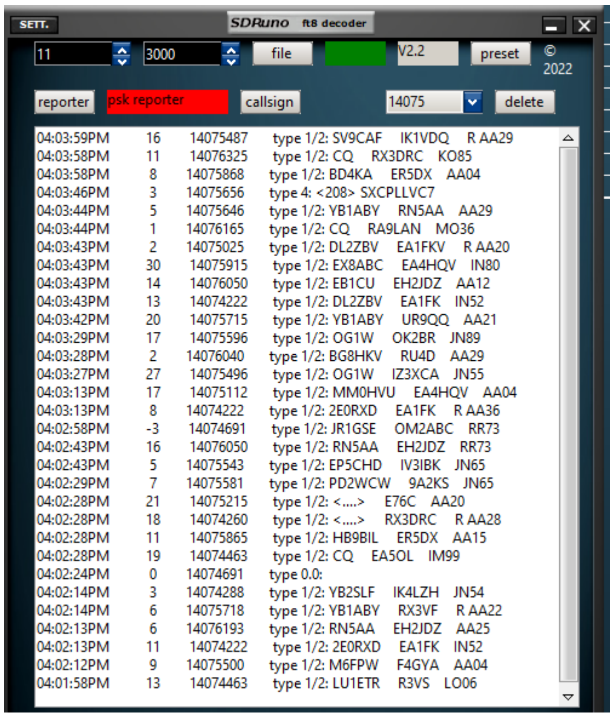
The widget for the FT8 plugin is straight forward and does not have a lot of controls. On top there are two lines with buttons and some info, and there is a larger space where the text of the messages received is written.

On start up the white box will show the callsign and the grid of the receiver - obviously only if specified[1]. Then it will show for each decoded FT8 message a few fields

- the time of decoding the message;
- the strength of the FT8 message, relative to the average signal strength during reception of the FT8 message;
- the frequency of the message with an accuracy of 3.125 Hz (tone distance is 6.25 Hz);
- the message type;
- the message itself.

On the top line in the control part of the widget there are two selectors and two buttons

[1]If no callsign and grid are specified no data will be transmitted to the pskreporter site.



```
04:03:59PM   16   14075487   type 1/2: SV9CAF   IK1VDQ   R AA29
04:03:58PM   11   14076325   type 1/2: CQ   RX3DRC   KO85
04:03:58PM   8    14075868   type 1/2: BD4KA   ER5DX   AA04
04:03:46PM   3    14075656   type 4: <208> SXCPLLVC7
04:03:44PM   5    14075646   type 1/2: YB1ABY   RN5AA   AA29
04:03:44PM   1    14076165   type 1/2: CQ   RA9LAN   MO36
04:03:43PM   2    14075025   type 1/2: DL2ZBV   EA1FKV   R AA20
04:03:43PM   30   14075915   type 1/2: EX8ABC   EA4HQV   IN80
04:03:43PM   14   14076050   type 1/2: EB1CU   EH2JDZ   AA12
04:03:43PM   13   14074222   type 1/2: DL2ZBV   EA1FK   IN52
04:03:42PM   20   14075715   type 1/2: YB1ABY   UR9QQ   AA21
04:03:29PM   17   14075596   type 1/2: OG1W   OK2BR   JN89
04:03:28PM   2    14076040   type 1/2: BG8HKV   RU4D   AA29
04:03:27PM   27   14075496   type 1/2: OG1W   IZ3XCA   JN55
04:03:13PM   17   14075112   type 1/2: MM0HVU   EA4HQV   AA04
04:03:13PM   8    14074222   type 1/2: 2E0RXD   EA1FK   R AA36
04:02:58PM   -3   14074691   type 1/2: JR1GSE   OM2ABC   RR73
04:02:43PM   16   14076050   type 1/2: RN5AA   EH2JDZ   RR73
04:02:43PM   5    14075543   type 1/2: EP5CHD   IV3IBK   JN65
04:02:29PM   7    14075581   type 1/2: PD2WCW   9A2KS   JN65
04:02:28PM   21   14075215   type 1/2: <....>   E76C   AA20
04:02:28PM   18   14074260   type 1/2: <....>   RX3DRC   R AA28
04:02:28PM   11   14075865   type 1/2: HB9BIL   ER5DX   AA15
04:02:28PM   19   14074463   type 1/2: CQ   EA5OL   IM99
04:02:24PM   0    14074691   type 0.0:
04:02:14PM   3    14074288   type 1/2: YB2SLF   IK4LZH   JN54
04:02:14PM   6    14075718   type 1/2: YB1ABY   RX3VF   R AA22
04:02:13PM   6    14076193   type 1/2: RN5AA   EH2JDZ   AA25
04:02:13PM   11   14074222   type 1/2: 2E0RXD   EA1FK   IN52
04:02:12PM   9    14075500   type 1/2: M6FPW   F4GYA   AA04
04:01:58PM   13   14074463   type 1/2: LU1ETR   R3VS   LO06
```

Figure 2: ft8Plugin: the widget

- The *first* selector sets the value for the number of iterations, to be applied in the LDPC decoding. Higher values *may* lead to better decoding but are expensive in terms of CPU load. Default value is 10, which should be enough for. A selected value will be stored and used the next invocation of the plugin.
- the *second* selector sets the bandwidth within the spectrum in which ft8 messages are searched for. Default value is 2 KHz, maximum is 6 MHz. The selected frequency is here the *center* frequency, so a width of 2 KHz means from -1 KHz to +1 KHz wrt the selected frequency. *Obviously, the larger the width, the more processing power is needed.*
- Next a button labeled *file* is there. As the name suggests, a file can be selected in which the data is stored. If the data is being written the label next to the button will be colored red, if no data is written the color of the label will be green.
- the second button on the first row is labeled *preset*. The plugin supports a number of preset frequency values. In previous versions this was a static list, in this version the user can add his/hers preferred frequencies to the list (and the default list is shortened).
- to the far right of the top line there is the inevitable copyright symbol.

The second line in the widget shows three (or four) buttons, a label and a combobox

- the button with the label *reporter* can be used to switch the *PSKReporter* on or off. The state is reflected in the label, *green* indicates that the reporter is "on", otherwise the label will color *red*.

- the PSK reporter sends the data to the server

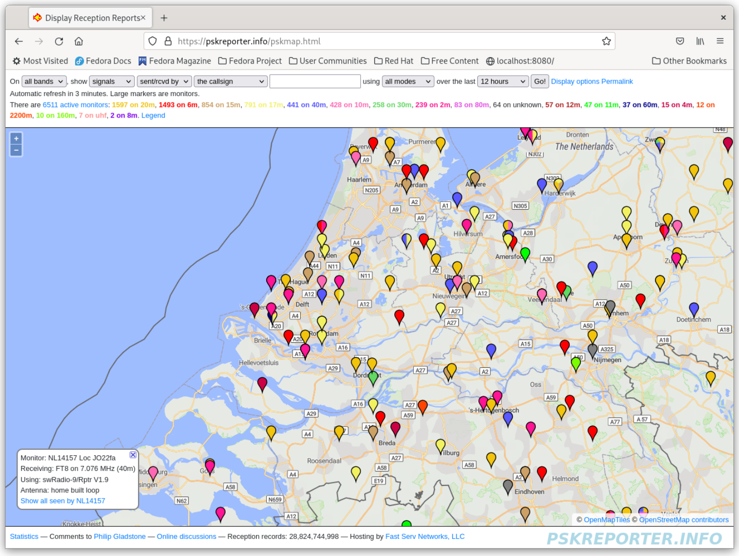
  `"https://www.pskreporter.info/pskmap.html`

  In order for the pskReporter info site to show the ”monitor”, i.e. data of the receiver that sends the messages, the ft8 plugin sends some information on the ”home” station, a.o. a callsign and the grid location to the pskreporter. The button *callsign*, when touched, shows a widget where a callsign and a grid can be entered. Values will be maintained between invocations of the plugin.

- if the psk report function is selected, a third button shows, labeled ”show”. If touched, the ft8 plugin shows the elements of the messages that are transmitter to the psk reporter. Transmission takes place in UDP mode, using port 4739 of the pskreporter server.

- FT8 is transmitted is well defined regions in different amateur bands. To ease switching between these bands, the plugin now contains a button - labeled - *presets* where one may select among a list of frequencies, known to be in these regions. By default the list merely contains four of five frequencies (of course the one I am using most of the time), and as said, your own preferred frequencoes can be added to the list. Of course, the added frequencies will be saved between sessions.

- the *delete* button will remove the last entered preset frequency.



On all bands, show signals sent/rcvd by the callsign using all modes over the last 12 hours Go! Display options Permalink

Automatic refresh in 3 minutes. Large markers are monitors.

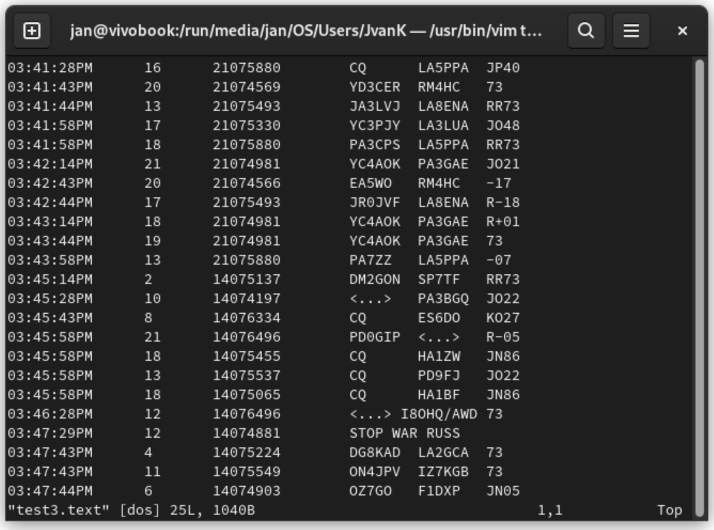
There are 6511 active monitors: 1597 on 20m, 1493 on 6m, 854 on 15m, 791 on 17m, 441 on 40m, 428 on 10m, 258 on 30m, 239 on 2m, 83 on 80m, 64 on unknown, 57 on 12m, 47 on 11m, 37 on 60m, 15 on 4m, 12 on 2200m, 10 on 160m, 7 on uhf, 2 on 8m. Legend

Monitor: NL14157 Loc JO22fa
Receiving: FT8 on 7.076 MHz (40m)
Using: swRadio-9/Rptr V1.9
Antenna: home built loop
Show all seen by NL14157

Statistics — Comments to Philip Gladstone — Online discussions — Reception records: 28,824,744,998 — Hosting by Fast Serv Networks, LLC

Figure 3: ft8Plugin: the map, example

The widget will show up to 30 of the last received messages. Of course all messages are saved in a file whenever a file is selected



```
jan@vivobook:/run/media/jan/OS/Users/JvanK — /usr/bin/vim t...
03:41:28PM      16      21075880        CQ      LA5PPA  JP40
03:41:43PM      20      21074569        YD3CER  RM4HC   73
03:41:44PM      13      21075493        JA3LVJ  LA8ENA  RR73
03:41:58PM      17      21075330        YC3PJY  LA3LUA  JO48
03:41:58PM      18      21075880        PA3CPS  LA5PPA  RR73
03:42:14PM      21      21074981        YC4AOK  PA3GAE  JO21
03:42:43PM      20      21074566        EA5WO   RM4HC   -17
03:42:44PM      17      21075493        JR0JVF  LA8ENA  R-18
03:43:14PM      18      21074981        YC4AOK  PA3GAE  R+01
03:43:44PM      19      21074981        YC4AOK  PA3GAE  73
03:43:58PM      13      21075880        PA7ZZ   LA5PPA  -07
03:45:14PM      2       14075137        DM2GON  SP7TF   RR73
03:45:28PM      10      14074197        <...>   PA3BGQ  JO22
03:45:43PM      8       14076334        CQ      ES6DO   KO27
03:45:58PM      21      14076496        PD0GIP  <...>   R-05
03:45:58PM      18      14075455        CQ      HA1ZW   JN86
03:45:58PM      13      14075537        CQ      PD9FJ   JO22
03:45:58PM      18      14075065        CQ      HA1BF   JN86
03:46:28PM      12      14076496        <...> I8OHQ/AWD 73
03:47:29PM      12      14074881        STOP WAR RUSS
03:47:43PM      4       14075224        DG8KAD  LA2GCA  73
03:47:43PM      11      14075549        ON4JPV  IZ7KGB  73
03:47:44PM      6       14074903        OZ7GO   F1DXP   JN05
"test3.text" [dos] 25L, 1040B                            1,1           Top
```

Figure 4: ft8Plugin: file output

4 A note on FT8 decoding

FT8 decoding is essentially based on a - more or less - ”intelligent” brute force approach (sounds like a ”Contradictio in terminis”). In a first step an attempt is made to preselect ”potentials”, i.e. potential messages based on matching the input with a costas array of 7 tones.

In the second step for each of these potentials an LDPC based error detection/recovery mechanism is applied. First the 79 subsequent tones are collected, the 58 data tones are mapped upon 174 soft bits (3 bits per tone) and the LDPC algorithm is asked to make that into ”hard” bits. Finally, in a third step the resulting message is subjected to a CRC check. If the check succeeds the bits are translated into a message.

Note that the parameters in the first stage may be too strict, and not all messages are in the set of potentials, or they may be too loose, and the set of potentials is too large. Some criteria have to be applied to do the selection, but it is not obvious that a single set of criteria is optimal under all circumstances.

An encoded FT8 message is 91 bits (including the CRC), and 83 error correction bits are appended. Of course, the probability that on receiving a transmission, the error correction bits contain - roughly - the same amount of errors as the message bits is pretty large.

The net effect is that the result of the application of the error correction may be that incoming garbage is transformed to something resembling a message, and the other way, a correct message may suffer from errors in the error correction bits and may be transformed into garbage. A CRC check is therefore absolutely needed.

Anyway, the decoder may - and probably will - miss some messages in the transmitted band, and I am still looking for the ”best” parameter settings for the different functions.

5 Future work

Current work is on optimizing parameter settings and extending the encoding. Detecting potential messages requires finding detecting peaks in the spectrum where the amplitude value is (relatively) large, compared to its environment. While that sounds easy, there is a pretty large ”noise” component, and parameters for detecting very small peaks are not always capabel of detecting large peaks.

In the current version an attempt is made to decode all message types, however, messages in some message types need further processing.

Further work is required to extract all calls that can be send to the PSKreporter.

6 Copyright and acknowledgments

The code in the plugin uses parts of the code of the FT4FT8 decoding software of Karlis Goba. Especially the LDPC decoder is (almost) a copy of his code, and the copyright to the parts taken or derived from his code are gratefully acknowledged.

The copyrights of the code of the plugin - not covered by the above - is by me. The software is available under a GPL V2. Note that under this license you are free to use the plugin, and I will make the source available to you on request, but the software is provided ”as is”, and no garantees are given that the software meets your requirements.

Note that - as the title of this document states - the plugin is still experimental, implying that the search for optimal parameter settings continues.