Receiving with DAB sticks: *The fm and dab receivers of the SDR-J suite*

Jan van Katwijk
JFF Consultancy
The Netherlands
J.vanKatwijk@gmail.com

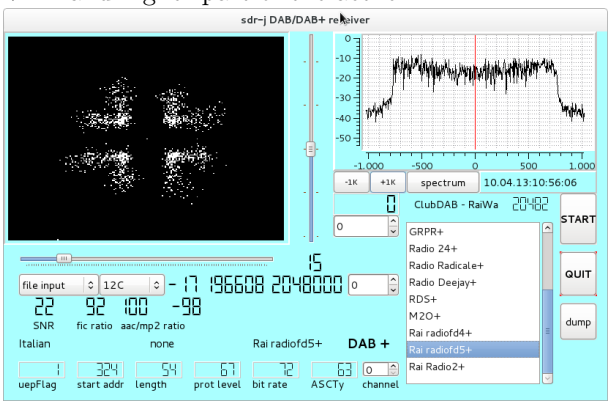
1 Introduction

DAB- and DVB-T sticks provide excellent opportunities for experimenting, they are cheap, have a wide frequency range and deliver I/Q data with a fairly high speed to a computer through an USB port.

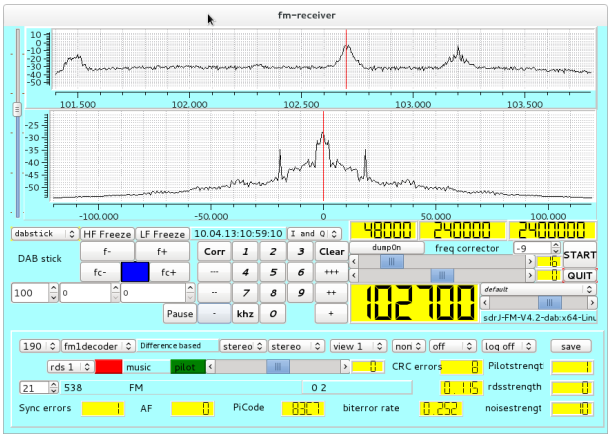
Software for controlling RTL2832 based DAB and DVB-T sticks is available through GPL-ed libraries from the osmocom project, and in a previous phase we modified some of our programs for use with such sticks. The original programs, however, were built originally around slow (i.e. 96K and 192 K samplerate) devices for data input.

Next to a more or less "classical" SW receiver, an FM sdr was developed and made available. This FM sdr has been restructured to take full advantage of the received spectrum. Furthermore, a full *software* DAB/DAB+ receiver was developed. It seems quite obvious that with a DAB stick one is able to listen to DAB, however, this implementation is a full software one, only taking a stream of I/Q samples from the stick.

Anyway, the result is a couple of programs with which it is fun to listen to music and to explore the VHF and higher part of the aether.



The picture shows the GUI of the DAB receiver. In this particular case the input (DAB+) is read from a pre-recorded file, simply because in the Netherlands there is no DAB+.



The above picture shows the GUI of the FM receiver. At the top display a spectrum of over 2M is shown, the second display shows the spectrum of the demodulated signal.
It can be seen that the GUI of the DAB receiver is simpler than that of the FM receiver, the latter has more settable parameters. One of its advantages is that it can be used to derive the frequency offset in the oscillator of the stick.

2 Installation

The software comes in two versions. For Windows 64 there are executables, for Linux there are sources, i.e. one has to create the executables. Although it is most likely possible to generate, no attempt was made to generate an executable for 32 bits windows versions. If an urgent need is shown, I might look into it. What must be realized, however, is that both programming are pretty resource consuming. While the FM sdr will function with lower rates[1], the DAB receiver uses a samplerate of 2048000 samples/second, and is really heavy in processing requirements.

2.1 Windows

Most libraries are linked statically. The obvious disadvantage is that the resulting executables are pretty large, the advantage is that they only need access to a (relatively) few non-standard dll's. These dll's are made part of the distribution.

Unpacking the distribution into a directory of one own's choosing is almost all that has to be done. For windows, however, one has to install an adapted usb driver. For this purpose the Zadig program is available. There are many examples on the internet of how to run. Basically just run the zadig program with the dabstick inserted in one of the USB ports. The program (should) detect(s) the DABstick, indicates which driver is installed (if any) and suggests a replacement.

2.2 Linux

The software was developed under Fedora, *almost* all required libraries are available as standard packages in Fedora repositories. One needs, next to the gnu compiler suite (g++),

- Qt-4.7 or Qt-4.8
- Qwt 5.2 (i.e. qwt5-4 in recent fedora distributions)
- libusbx,

[1] Rates can be set through the ini file

- libportaudio[2].
- libsndfile and libsamplerate
- libfftw3
- libfaad.
- the ffmpeg packages which contains -lavcodec -lavdevice -lavutil -lavformat -lswresample -lswscale -lavfilter
- librtlsdr.

For all of these packages, one needs both the library and the development package.

All but the *librtlsdr* is indeed available and can be installed through the mechanisms available through the Linux distribution. librtlsdr, however, needs to be created and installed through another mechanism. A description of the library and how to build it is to be found on the osmocom site ¡http://sdr.osmocom.org/trac/wiki/rtl-sdr¿. It consists of the steps of acquiring the sources, then running an installation script.

Note that while the Windows implementation uses the KJMP package for decoding DAB packages[3] the Linux version has a choice between KJMP and ffmpeg. Since ffmpeg is available as standard package in at least the Fedora distribution, the default is set to ffmpeg for DAB. For DAB+ there is a choice between ffmpeg and faad, with the latter as default.

By changing some constants in the file "dab-constant.h" one may choose for KJMP for DAB and ffmpeg for DAB+.

Other Linux distributions most likely will provide the same packages, probably on other locations, therefore the ".pro" files, may need to be adapted to the particularities of the different Linux distributions. These files contain the meta information for qmake to generate a makefile.

In case KJMP is used, the file "mp2procesor-kjmp.h" needs to be uncommented, while "mp2processor-ffmpeg.h" needs to be commented out, and the other way around.

When all libraries are in place, one executes

```
qmake-qt4
make
```

to generate the executables.

It might be wise to ensure rights for reading and writing usb port and soundcards before running the program. One may use the instructions given on the aformentioned osmocom page.

3 Running the programs

The programs can be run be either clicking on them in a windowing environment or using a commandline with the name of the program.

Starting the program will set the program in an "idle" state, waiting for a command to start processing. Before, however, processing can start, a *device* as "radio device" has to be selected. A device can be

- *no device*. No input device will be selected, input will consist of "nothing";
- *dabstick*. A dabstick as input device is assumed and the I/O and control are prepared. DABstick output - and thus program input - consists of a raw byte stream, containing I/Q bytes.

[2]Some older versions of fedora have portaudio 1.18 as standard package. Replace this with 1.19

[3]KJMP is developed by Martin Fiedler, and even extended by him for use with this receiver.

- *file input*. A menu will be displayed for selecting a file. The data in this file will be input. Note that the format of the input is a raw stream of I/Q data bytes as delivered by the DAB stick. The filename should end on ".raw".

Both programs will use an ".ini" file for maintaining state information between successive invocations. This state information consists of the some settings and some settable configuration parameters. Absence will not harm the programs, just some (seemingly) suitable default values will be chosen and - after normal program termination - such ini files are created. The files will be named $(HOME)/.jsdr-xxx.ini, where "xxx" is replaced by either "dab" or "fm-dab". It is common practice to just start the programs and quit them to obtain ini files with default values.

Both programs give the opportunity to save the raw data read from a DABstick into a file, and read it back later on with the *file input* element of the device menu. Both programs provide a "dump" button for this purpose, the button is only meaningful in case a DAB stick is the selected input device. Pressing the button will cause a menu to appear on which one can specify (or select) a filename. Pressing the button for a second time (or selecting the "quit" button) will stop dumping and closing the file. Note that - at least currently - the resulting file does not contain information on the actual datarate. It is quite well possible - though pretty useless - to store the DAB stick data of a part of the FM broadcast band, with a rate of 960000, and subsequently use it as input for the dab receiver.

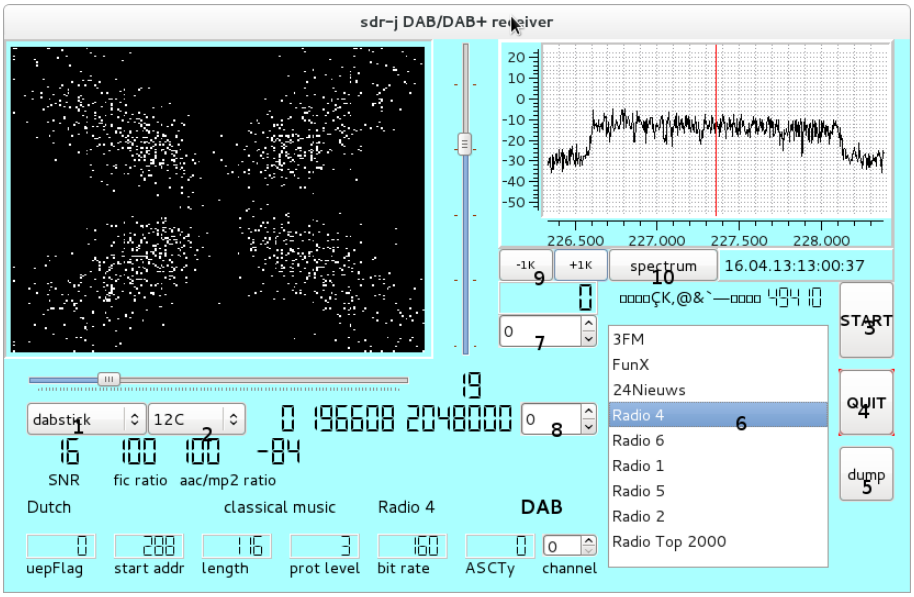
3.1 The DAB receiver

After *selecting a device* (selector labeled "1"), the user of the DAB receiver may select a *channel* (selector labeled "2"). The channels map onto the standardized frequencies for the DAB channels in band III. Processing will start with pressing the *start* button (labeled "3").

When started, the DAB receiver will try to synchronize with the incoming data[4] and, when synchronized, it will try to identify the name of the *ensemble*. When in sync, three numbers, right from the device and channel selector, are displayed in larger digits:

- the detected offset in KHz. A simple mechanism is applied to try to correct for the deviation of the DAB oscillator. When synchronized, the offset in KHz will be displayed.
- the length of the frames that are actually detected, which should be 196608.
- the samplerate, which should be 2048000.

[4]This may take some time, depending on the quality of the received signal



When correct data could be found, the names of the stations covered by the ensemble will be displayed. Selecting such a station will start further decoding (label "6").

The bottom line of the GUI will show (technical) information on the selected station. The display top-left shows received ofdm symbols[5]. The more a large X is shown, the better the quality of the received signal. The spectrum display shows the spectrum of the received signal, it has a width of 2048000 Hz.

Dumping the DABstick data will begin after pressing the "dump" button (labeled "5") (and stopping after pressing it again). The buttom "spectrum" (labeled "10") allows switching between spectrum and waterfall displays. For experimental purposes, there are some more spinboxes and selectors. We have

- spinbox labeled "8". One might select an offset of the sampling frequency here. This offset is in steps of 50 samples/second
- spinbox labeled "7", One might alter the oscillator frequency of the connected DAB stick (if any) in steps of 1 KHz
- the buttons labeled "9". One might alter the "soft" i.e. correcting oscillator value in steps of 1 KHz.

Many of the settings for the DAB receiver will be read from the ".ini" file on starting the program. If no ini file exists (yet), suitable defaults are chosen. The default[6] ".ini" file for the dab receiver is $(HOME)/.jsdr-dab.ini". This file contains entries

```
[General]
channel=12C
traceback=5
device=file input
latencyLevel=10
Concurrent=1
displaySize=512
decay=10
freqCorrection=-50
```

[5]the frames consist of 76 blocks, the symbols following the differential decoding of the second block in each frame is shown

[6]By passing a "-i filename" to the command line, when starting the program, a non-default ini file is selected

In general, it is safe to experiment with the values in this file, however no garantee is given that the program will function correctly for any chosen value. The settings that can be influenced by the GUI are written out on normal program exit.

- *channel* and *device* indicate what the tuning was the last session.
- *traceback* is an indication for the search depth of the viterbi decoding. The number should be positive and less than or equal to 20. In general it holds that the higher the number the deeper the search (which might improve the result, but will cost lots of processing power). Note that this setting cannot be influenced through the GUI.
- *latencyLevel* indicates the latency. A pretty high latency is required since there is quite some processing to be done in the final MPEG decoding. Again, note that the value cannot be altered through the GUI.
- *Concurrent* is a flag telling to do this MPEG decoding in a different thread (value 1) or in the same thread as the rest of the dab decoding (value 0). On a machine with many computing cores, there is a slight gain in efficiency when concurrent execution is selected. This setting can not be altered through the GUI.
- *displaySize* the number of elements in the spectrum display. This width cannot be altered through the GUI. It is unwise to use values other than powers of 2, furthermore, the higher the value the larger the penalty in resource consumption. The smallest value with reasonable results is 128.
- *decay* is the averaging factor when displaying the spectrum. Default a value of 5 is chosen, which makes the spectrum appear somewhat more quiet. The number should be positive. This value cannot be altered through the GUI.
- *freqCorrection* can be used to correct the DAB stick's oscillator. Different DAB sticks need different corrections[7], the value cannot be set through the GUI.

3.2 The FM receiver

The fm receiver is meant for experimenting, it therefore contains a pretty large amount of sliders and selectors. The GUI is structured into three parts:

- The *top part* contains two displays, a display displaying the spectrum of the dabstick data, and a spectrum of the decoded data;
- The *middle part* contains sliders, selectors and buttons for device and frequency selection;
- The *bottom part* contains sliders, selectors and buttons for fm specific choices.

Before the start button (labeled 2) will be effective, an *input device* has to be selected (selector is labeled "1"), and an *output device* (selector is labeled "9") needs to be selected.

3.2.1 Device and frequency selection

Device selection (labeled "1") is as mentioned in the previous section. A particular frequency (in Khz) can be selected using the numeric keys in the GUI. With the "Corr" and the "Clear" button on removes the last digit resp. the typed frequency. A selection should be terminated by clicking on the "KHz" button;

Altering a frequency can be done in several ways:

[7]The correcting value is only dependent on the DAB stick, the FM receiver contains a mechanism for selecting an appropriate corrector value

- by clicking with the mouse on a position in the top display;
- by clicking on buttons left and right from the keypad: the selected frequency will be modified by the number of KHz mentioned on the button;
- by clicking on the f++ or f−− buttons (label "4"): the selected frequency will be modified by the number of KHz mentioned in the selector (labeled "11").
- by clicking on the fc++ or fc−− buttons (label "5"): the frequency will continuously change in the frequency range (in Mhz) specified (the two spinboxes below the label), using steps as mentioned in the selector (labeled "11"). The time between successive steps can be modified by repeatedly clicking on the fc++ resp. fc– button.

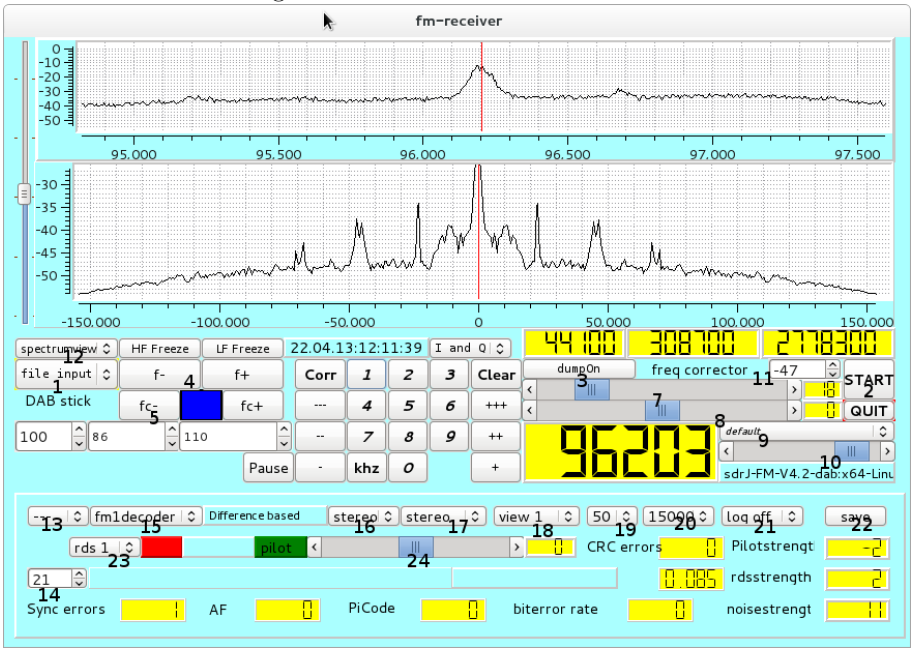
Note that mouse clicks on the bottom one of the two displays has a different effect: clicking on the left mouse button will enlarge (part of) the spectrum, clicking on the right mouse button will undo an enlargement step.

The two sliders below the "dump" button (label "3") are resp. the HF attenuation (labeled "7") and the HF balancing (labeled "8"). Below the output selector, one finds a slider implementing volume control (labeled "10"). The three numbers in the LCN displays above the dumpOn button present resp. the rate of the output, the rate of the FM processing and the rate of the DAB stick.

The combobox labeled "12" allows making a choice between a spectrum display as shown in the figure and a spectrum displayed as waterfall. Finally, a rather particular selector is the "freq corrector" (label "11") with associated value (initially) 0. DABstick frequencies might be a little off. With this selector one can determine a setting with a pretty accurate offset for the DAB stick frequency select. Its operation is discussed later on.

3.2.2 Typical FM selections

The bottom section of the GUI contains a numbers of sliders, selectors and displays supporting typical functions for the FM signal.



The FM receiver supports additional filtering of the decimated samples. The selectors at the left of the bottom part of the screen are to support this. The selector labeled "13" indicates whether additional

filtering is in place or not, and allows selecting the band for filtering. With the spinbox below this selector (labeled "14") the filter depth (a classical FIR filter is used) can be selected. Obviously, when switched on, some more computer power is used since this filtering operates on the FM sample stream.

On the top row of the bottom screen part we further see

- a decoder selector (labeled "15"). A choice can be made out of 5 (more or less) different FM decoders. The name of the selected decoder is displayed to the right of the decoder selector.
- a stereo/mono selector (labeled "16");
- a selector for the output (labeled "17"): one might select the individual channels or a combination of the stereo signal. Obviously, this selector is only effective when stereo is the selected mode.
- a view selector (labeled "18"): one might select among several different views on (elements of) the decoded signal that are displayed on the bottom one of the two spectrum displays.
- the deemphasis filter (labeled "19"). Choices are "none", "50" or "75".
- a low pass filter applied on the decoded signal (labeled "20"). Choices are "none" or a selection of predefined values.
- a logging function switched on or off (labeled "21"). This function, when selected will display the values for the pilotstrength etc. once every second.
- a logging save function (labeled "22"). This function, in combination with the logging function will save the logged values into a file.

In the middle row there is an RDSs selector (labeled "23"): one might choose among no rds, or one of two rds decoders. The rds decoders, when selected, will display RDS data into the label fields.

The slider in the same row (labeled "24") is a slider to balance output.

3.2.3 Fine tuning the DAB stick

One particular LCN display is worth mentioning: right from the rds label fields, left from the "rdsstrength" indicator, the LCN display displays the DC component in the decoded FM signal. As well known, the DC component is proportional to the offset in tuning of the FM signal. A meaningful correction value (see previous section) can be found by tuning on a local FM transmitter and changing the freq corrector (labeled "6") such that the DC component is minimized. The setting of the freq corrector will be stored in the ini file with the label *freqCorrector*.

3.2.4 The FM ini file

A (pretty large) number of settings will be stored in an ini file. This file will be read on start of the program and the values will be written out on (normal) termination of the program. The name and location of this file are $(HOME)/.jsdr-fm-dab.ini". Many of the fields of the ini file are either discussed with the dab receiver or relate directly to a switch, selector or slider of the GUI.

```
[General]
displaySize=512
averageCount=5
repeatRate=10
fm_increment=100
spectrumAmplitudeSlider=62
attenuationSlider=16
IQbalanceSlider=0
```

```
inputModeSelect=I and Q
VolumeSlider=66
fmFilterSelect=190
fmFilterDegree=21
fmMode=stereo
fmDecoder=fm1decoder
fmRdsSelector=rds 1
fmView=view 1
fmChannelSelect=stereo
fmDeemphasisSelector=none
fmStereoSlider=0
logging=log off
deviceSelect=dabstick
streamOutSelector=default
frequency=102700
IncrementIndex=0
min_loop_frequency=86000000
max_loop_frequency=110000000
dongleOffset_Khz=0
dongleOffset_Hz=0
freqCorrector=-9
fmRate_Multiplier=7
dabRate_Multiplier=9
fmLFcutoff=15000
```

A few additional remarks:

- dongleOffset can be set, e.g. when using an upconverter. In this case the appropriate handler for the DAB sticks will add the dongleOffset to its oscillator frequency.
- audioRate indicates - as can be expected - the rate of the audio output channel. Valid values are 24000, 44100 and 48000. Associated rates are the fmRate and the dabRate. The fmRate indicates the rate that is used when processing fm samples. It should be an integer multiple of the audioRate and it should be larger than 150000. The fmRate is computed by multiplying the audioRate with the fmRate_Multiplier. The dabRate indicates the rate used for input from the DAB stick. It should be a number within 900000 .. 3200000. The dabRate is computed by multiplying the fmRate by the dabRate_Multiplier. If any of these numbers turns out to be "out of region", defaults will be selected.

3.2.5 Keyboard control

As an experiment, some functions of the fm-dab receiver can be controlled through the keyboard (as well as the controls in the above paragraphs). The facility is highly experimental, since it interferes with keyboard handling as is done by the Qt system itself.

- "Q" is terminate process (i.e. the "quit" key;
- "R" is return to normal mode;
- "S" is start the program (i.e. the "start" key;
- "Ctrl": set in CONTROL mode and listen for 5 seconds to any of
  - "Ctrl" return to normal mode;

  - "Up" Increment frequency with 1 Khz;
  - "Down" Decrement frequency with 1 Khz;
  - "PageUp" Increment frequency with 100 Khz;
  - "PageDown" Decrement frequency with 100 Khz;
  - "+" alternative for f++ button;
  - "-" alternative for f−− button;
  - "U" increment frequency with the width of one (bottom) screen;
  - "D" decrement frequency with the width of one (bottom) screen;
  - "A" ask for a new frequency

- "Alt": set in ALT mode and listen for 5 seconds to any of
  - "A" not yet implemented;
  - "F" not yet implemented
  - "T" not yet implemented

4 Final remarks

This software is made available through a GPL license. It uses large numbers of libraries, made available through (L)GPL style licenses and parts of the code is based on ideas of others and in some cases even uses code lines of others. In all cases attempts are made to indicate the rightfull owner of the copyrights.