JSDR-3.0 SDR
A Linux software package for cheap sdr reception

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

June 1, 2012

1 Introduction

JSDR is the name of a set of support programs for software defined radio. The programs implement receivers in the sdr domain, covering with suitable - but cheap - hardware a range of 100K .. 1700 MHz for a variety of decoder modes.

The set was originally developed as a SW receiver for the Elektor SDR card (described in the Elektor May 2007). It was extended with software for the preselector for that card (described in Elektor December 2009). Later, support for the italian pmsdr kit was added. Since the pmsdr kit supports a wider range of frequencies (up to 165 Mhz through the 3-d harmonic of the Si570 oscillator), a separate version, specific to AM/FM, was derived. Recently, a DAB stick was acquired, and inspired by the software, made available through the osmocom.org site for rtl2832u based DAB sticks, both the sw receiver and the fm receiver were equipped with support for these sticks. A stripped version of the FM receiver was made into a spectrum viewer, such that the spectrum of a selectable band in the frequency range 50 .. 1700 Mhz is shown. Since the Elektor card was - at the time - priced app 100 Euro, the pmsdr app 250 Euro and the DAB stick less than 30 Euro, the total cost of the complete set of hardware was less than 400 Euros, cheap and less than the laptop on which the software was developed.

The software is written in C++, it was developed under Linux (recent versions of Fedora and Ubuntu, both 32 and 64 bits versions), using Qt (and Qwt) for the gui. Thanks to the portability of Qt and C++ and thanks to Mingw-32 as Windows development platform with all its libraries (e.g. portaudio and libusb-1.0), porting it to Windows turned out to be surprisingly simple (though not trivial).

The software has been running under Windows XP, Vista, W7, Fedora 14/15,16 and Ubuntu 10.04/11.10/12.04. Currently I am running the software on three platforms, Fedora-16 (32 bits), Ubuntu 12.04 (64 bits) and Windows-7 with Mingw-32.

The software is being developed as a hobby project and all sources are available under an Open Source License, the GPL V2 and V3. It builds upon many open source libraries (e.g. Qt, Qwt, fftw3, libusb-1.0, libftdi, libsamplerate, libsndfile, libportaudio) while other existing software available under similar open source licenses, e.g. gmfsk, fldigi, cuteSDR, served as source of inspiration (and provided some lines of code) for the various parts.

Since the three programs have the same basis, their "look and feel" is the same. Indeed, many underlying support modules are the same, and all three programs are based on using Qt and Qwt as mechanisms for creating GUI's.

The programs share the interface for entering and modifying frequencies: a keypad with which a frequency can be entered, keys for shifting the frequency, and a display at which specific frequencies can be selected by a mouse click. The SW receiver supports selecting frequencies on both displays, the FM receiver and the spectrum viewer provide support for automatic stepping through a range of frequencies. The $f++$ and $f--$ buttons allow manual stepping, $fc++$ and $fc--$ allow automatic stepping. Touching the $fc++$ or $fc--$ button more than once will alter the stepping period. Touching the $fc++$ when automatic stepping is going on with a negative step size, will decrement the step time. Similarly for $fc--$, which then decrements the step time for the positive direction.

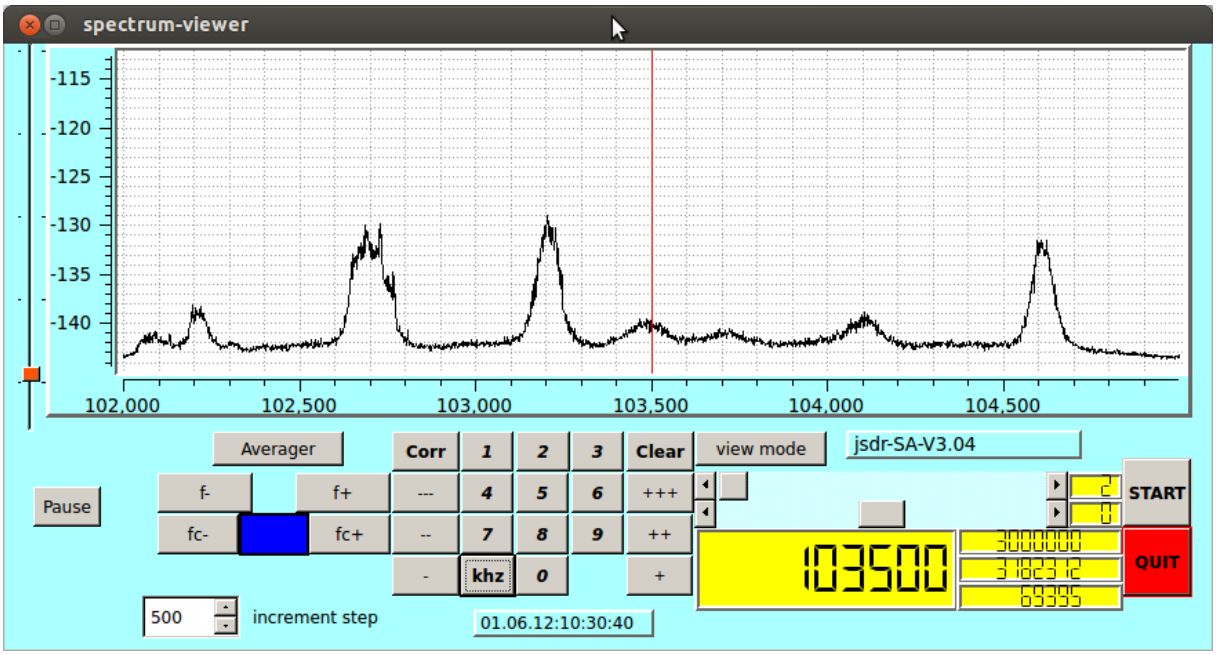
Furthermore, all three programs have a similar mechanism for initialization through an ini file. The ini file contains settings for an initial value for selected sliders, buttons and switches and are generated easily by running the program with the "-d" flag. Default place for ini files is the $(HOME) environment. An axample of an ini files for each of the three programs is included in the distribution. The "-i filename" option provides the opportunity to apply a non-default ini file.

2 The programs

2.1 The spectrum viewer

A simple DAB stick can be used for sdr purposes: its tuning range is wide, its frequency can be set and a stream samples (8 bit though) can be collected through an USB-2.0 port. The spectrum viewer supports DAB sticks with the rtl2832u and one of the tuners fc0013, fc0012 or e4000. Software support for the stick and tuner is derived from available osmocom software.

Since the tuning bandwidth is wide, and the speed of the datastream is selectable between ca 1MS/second to 3.2 MS/second, it is an ideal vehicle to deliver a sample stream for showing a fairly wide spectrum.



The figure shows a spectrum of a part of the FM broadcast band, with a spectrum width of 3 MHz.

The samplerate for the inputstream is set either by default (2MS/second) or through a command line parameter (-C xxxx). The actual frequency can be set through the keyboard and altered by either typing a new frequency, or by stepping though the frequencies, either manually (f++ or f– button) or automatically (fc++ or fc– button). Step size as well as step time can be set. Scanning the full FM broadcast band with a step size of 100K, and a step time of 1 second will take a couple of minutes. Boundaries can be set on initialization such that stepping will be repeated when the upper or lower boundary is reached, depending on the step direction. These boundaries are either defaulted to 86500 KHz to 1000000 Khz or set through values in the ini file.

2.2 The SW receiver

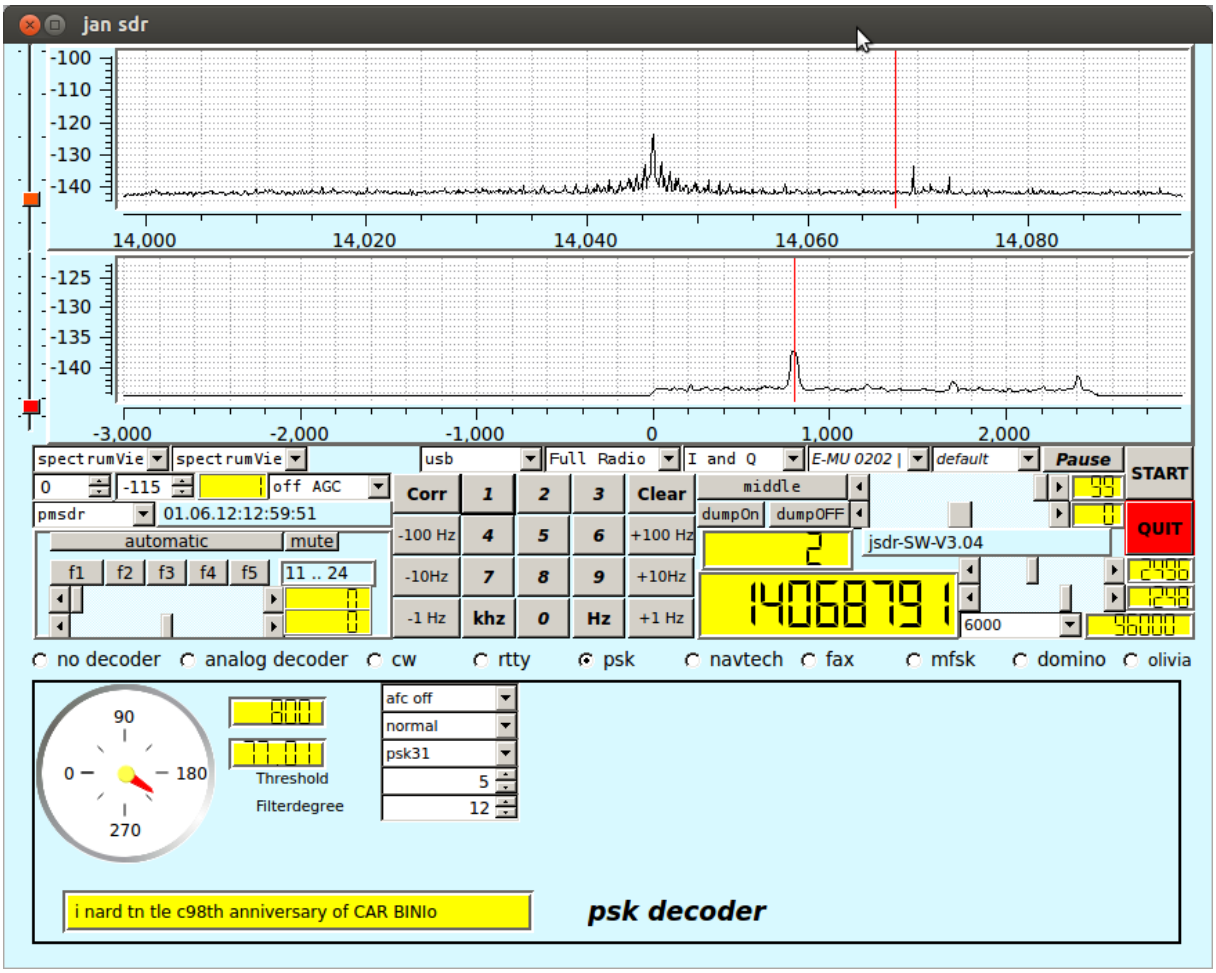
The first program in the set was (a predecessor of this) sw receiver. The program is still the most complex program of the three (complex from the user's point of view). The program was originally built with the Elektor SDR card in mind, i.e. for frequencies from app 150 KHz to 30 MHz. It was extended with support for the pmsdr kit. This kit supports - through the 3-d harmonic of the Si570 oscillator - frequencies up to to 165 MHz (although with reduced performance), with the pmsdr kit we therefore get a receiver with a continuous tuning range from 100KHz to 165MHz. Finally, support for the above mentioned DAB Stick was added. With the DAB stick, we get a receiver with a continuous tuning range from 45MHz to 1700 MHz.

When connected to either the elektor card or to the pmsdr, input samples will be

read though the attached soundcard, and - obviously - an input device should be selected before clicking on the start button. When connected to the DAB stick, input samples are coming though the USB bus. Default inputrate is 96K, this can be set to 192K or 48 K through a command line parameter (48000 use -S option, 192000 use -R option. Default outputrate is 48K. When reading from the DAB stick, predefined input speed is 2880000 Samples/second which is then filtered and decimated.

The receiver has two displays, one showing the spectrum of the input samples, the second one showing the spectrum of the filtered, shifted and decimated samplestream, the stream entering the selected detector. Clicking on any of the displays will set the input frequency to the frequency pointed to. each of the displays can be set to show a waterfall rather than a regular spectrum.

A bandfilter can be selected around a specified frequency, both the bandwidth and the centerfrequency of the filter can be manipulated. Frequencies can be specified with an accuracy of 1 Hz. The filter is an FFT implementation of a FIR filter (it actually consists of two filters) of order over 1000. A number of standard filter settings, e.g. for am usb, lsb, fax, is predefined and a selection can be made by a simple mouse click. Practical band selection is from app 100 Hz to (half) the working sample rate. One of a number of predefined working rates can be selected.



A variety of decoders is available. For each of the decoders a separate subscreen (the bottom part) is shown, with selectors, sliders and buttons specific to the selected decoder. The screen will appear when selecting the decoder.

- As default, a "no decoder", is available. This decoder does essentially nothing, just passing the decoder input to the output, while showing the input samples on a small complex plane;
- An analog decoder is selectable. This decoder implements the the common analog decoder modes such as am, fm and ssb.
- An rtty decoder can be selected. This decoder implements rtty decoding with lots of user selectable parameter settings.
- A CW decoder can be selected. This decoder implements adaptive CW decoding, again with lots of user selectable settings.

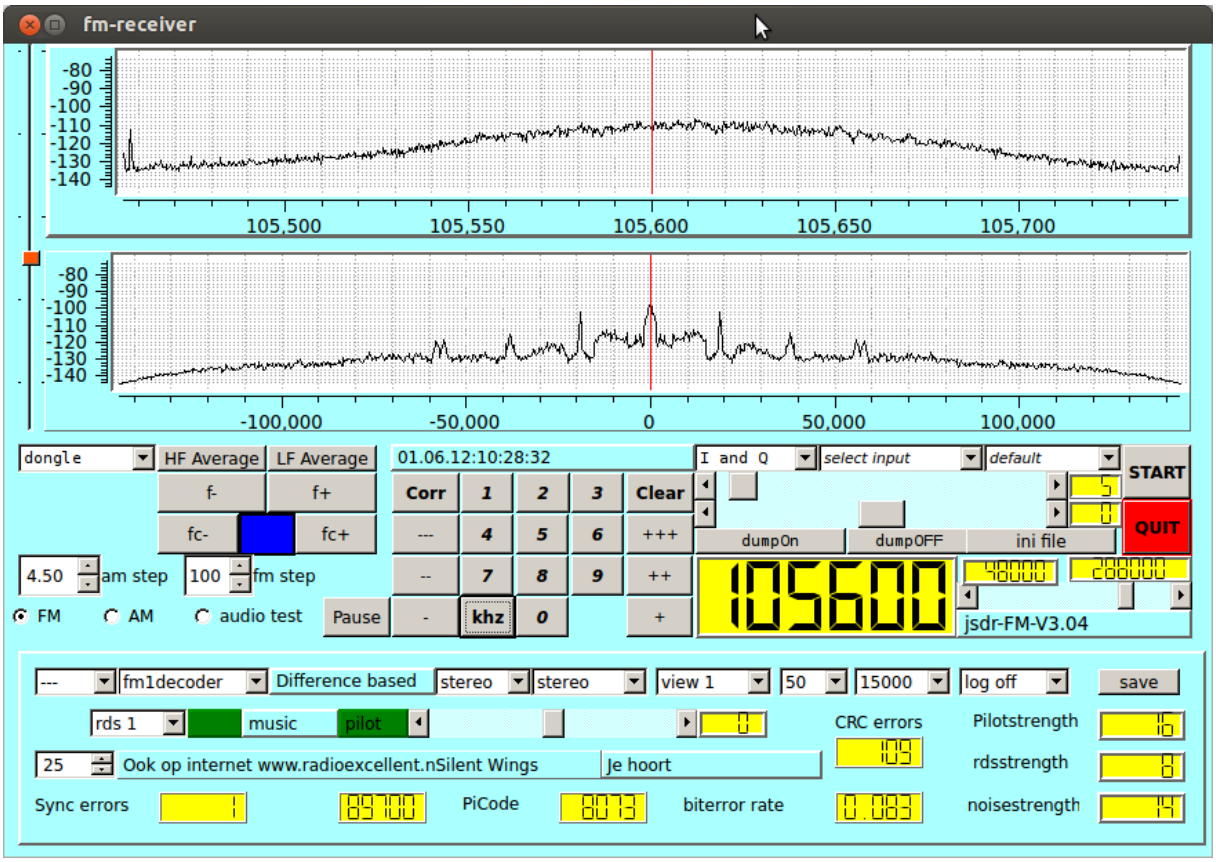
- A psk decoder can be selected. This decoder implements both bpsk and qpsk decoding for a variety of speeds.
- An mfsk decoder can be selected. This decoder implements the common mfsk modes, and is equipped with additional visual tuning aids.
- A domino decoder can be selected. This decoder implements the common domino modes.
- An olivia decoder can be selected. This decoder, essentially a wrapper around a package made by Pawel Jalocha, implements the standard olivia modes.
- A wheatherfax decoder can be selected. This decoder, a partial reimplementation of the work of Christoff Schmitt, is implementing wheatherfax decoding. The received picture can be stored into a file.
- A navtex decoder, allowing coastguard and other amtor-B messages to be decoded and made visible.

Obviously, there is a provision for dumping the input samples into a ".wav" file, and processing these (or other) ".wav" files. When processing ".wav" files, sample rate conversion is supported, so files created during a session when running 192K are valid input when being processes during a session running e.g. 48K and vice versa.

Finally, the receiver provides the opportunity to *not* select a radio device, but to process data that is entering through the soundcard. It is even possible to just pass this data on (after decimation) to a decoder, allowing normal receiver output to be processed as well.

2.3 The AM/FM receiver

The third program is dedicated to AM/FM reception. The program implementing the receiver supports - next to file input - the aforementioned pmsdr kit and DAB stick.



The software, when connected to the pmsdr kit, supports a continuous tuning range from 100K to 165 MHz. The lower frequencies mostly provide AM broadcasts, the FM band obviously provides FM broadcasts. When connected to the DAB stick, a continuous tuning range from app 45 MHz to 1700 Mhz is supported.

As with the spectrum viewer, automatic (or manual) stepping through a predefined frequency range is supported, as is - obviously - manually setting whatever frequency in the domain accepted by the connected device.

For each of the two reception modes a separate subwindow is defined. For the AM it is simple and not shown here, it contains a selector for the passband that is to be demodulated, it contains a switch and setting for the AGC and a selector for any of a few different detector implementations.

The focus in the implementation is on the FM receiver. A selection can be made from 5 different implementations of an FM demodulator. Stereo reception is of course possible, one can select stereo, the left channel, the right channel, and the L+R part as well as the L-R part of the signal. In general, the quality of the L-R channel is indicative for the overall quality.

RDS decoding is supported: a selection can be made out of 2 different implementations. Furthermore, an input bandfilter can be selected for a number of predefined widths. The standard de-emphasis filters can be selected, and the width of a low pass fil-

ter is selectable. As an aid in understanding the demodulated signal, there is a choice in what the second display will show, the full demodulated signal or one of its constituents. As additional feature, a log can be made of some relevant variables.

What must be noted is that the samples coming from the DAB stick are 8 bit samples, resolution is therefore limited. However, as the picture shows, even then RDS decoding, at least partially, is possible for some of the stronger FM broadcast stations.

Some command line options are specific to parameter settings for the DAB stick:

- The *-F* option tells that 288000 samples/second will be handled. In the current settings, this is only useful for the DAB stick. When set to 288000 samples/second, performance (as well as CPU usage) increases. Default is 192000 samples/second.

- The *-G* flag is specific to the DAB stick. It tells that the band with which the FM/AM decoding is done is to be taken from 0 .. 2 * samplerate / 2, rather than a direct mapping from - samplerate / 2 .. + samplerate / 2. Noise around the zero IF coming from the DAB stick can be eliminated this way.

- The *-m number* option is used to map the selected internal sample rate to the rate of the DAB stick. The *number* should be even. In some cases the combination 288000 samples/second internally and an external samplerate of 2880000 is a little heavy for my laptop and output starts stuttering. The *number* is adapted when its application would lead to an external samplerate outside the range of 960000 .. 3200000 samples/second.

As with the SW receiver, the input samples can be written onto a (.wav) file, and wav files can be used as input for the receiver.

3 The distribution

The distribution consists of a ".tgz"file. When unpacked, a tree is generated with directories:

- *docs*. This directory contains this document.

- *ini-files*. This directory contains example ini files for the three programs.

- *swreceiver* which contains the main sources for the sw receiver,

- *fmreceiver* which contains the main source needed for the fm receiver,

- *spectrum-viewer* which contains sources for building the spectrum viewer.

- *righandling*, which contains all files related to the radio devices,

- *scopes*, which contains all files for the implementation of the displays,

- *filters*, which contains the implementation of the various FIR and IIR filters,
- *soundcard*, which contains the interface code to the portaudio library, as well as the file reading functions,
- *utilities*, which contains a few support functions, and
- *viterbi*, which contains an implementation of the viterbi algorithm, used in the implementation of qpsk and mfsk modes.

4 Sources of inspiration

The software uses many ideas - and in some cases lines of code - from others. The main sources of inspiration for various parts - other than Qt or Mingw - are

- *Signals, Samples and Stuff: A DSP Tutorial* by Doug Smits (QEX 1998, 4 parts), really the set of papers that made me decide to do some programming in the field of SDR.
- *The Scientist and Engineer's Guide to Digital Signal Processing* By Steven W. Smith, Ph.D. California Technical Publishing P.O. Box 502407 San Diego, CA 92150-2407. This book, found on the internet, provided me with a first introduction to digital processing. All 640 pages are available on the internet.
- *Practical Analog and Digital Filter Design* by Les Thede, Artech House, which provided me with the basics for IIR filters, their design and their implementation.
- *Implementation of FM Demodulator Algorithms on a High Performance Digital Signal Processor* by Franz Schnyder, Christoph Haller. Diploma Thesis Hochschule fur Technik Rapperswil, Nanyang technological University 2002. Gave a readable overview on 4 (5) different algorithms for the decoding of FM.
- *CuteSDR Technical Manual*, October 2011 by Moe Wheatly, which gives a good view on various techniques used in the cuteSDR software and helped to improve some of the algorithms used.
- sources of many existing programs, in particular
  - osmocom rtl-sdr, a set of programs and drivers for the rtl2832u based DAB sticks (see osmocom.org), which forms the basis for the interface to the stick.
  - RTTY, an FSK decoder program for Linux (Jesus Arias, 2001). This program gave the first hints to process CW and RTTY.
  - fldigi (W1HKJ), that gave the hint on how to handle CW with an adaptive algorithm.

  - gmfsk (Tomi Manninen OH2BNS). A great program that gave a real insight in decoding the psk, mfsk and domino.
  - hamfax (Gerhard Schmitt), which was the source for the fax implementation.
  - digiRadio (Alberti Perotti and others), and
  - FM Stack (Michael Feilen), that gave the inspiration for the implementation of FM software.
  - The pmsdr interface software (Martin Pernter, Andrea Montefusco), and the Si570 data sheets. The current pmsdr interface has some parts that are directly derived from this interface software.
  - The elektor interface software (Burkhard Kainka) and the hamlib software that learned me how to handle the Elektor card.

- and numerous anonymous sources on the internet.

5 Availability and licensing

This software is available under the GPL. The software was written by me, updated by me, using many ideas - and even lines of code - from others, as far as I can trace all available under the GPL.

```
/*
 *
 *    Copyright (C)  2010, 2011, 2012
 *    Jan van Katwijk (J.vanKatwijk@gmail.com)
 *    JFF Consultancy
 *
 *    This file is part of the JSDR (ESDR).
 *    Many of the ideas as implemented in this software are derived from
 *    other work, made available through the GNU general Public License.
 *    All copyrights of the original authors are recognized.
 *
 *    JSDR is free software; you can redistribute it and/or modify
 *    it under the terms of the GNU General Public License as published by
 *    the Free Software Foundation; either version 2 of the License, or
 *    (at your option) any later version.
 *
 *    JSDR is distributed in the hope that it will be useful,
 *    but WITHOUT ANY WARRANTY; without even the implied warranty of
 *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *    GNU General Public License for more details.
 *
```

```
*    You should have received a copy of the GNU General Public License
*    along with ESDR; if not, write to the Free Software
*    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
*
*/
```