WFM-RPI*
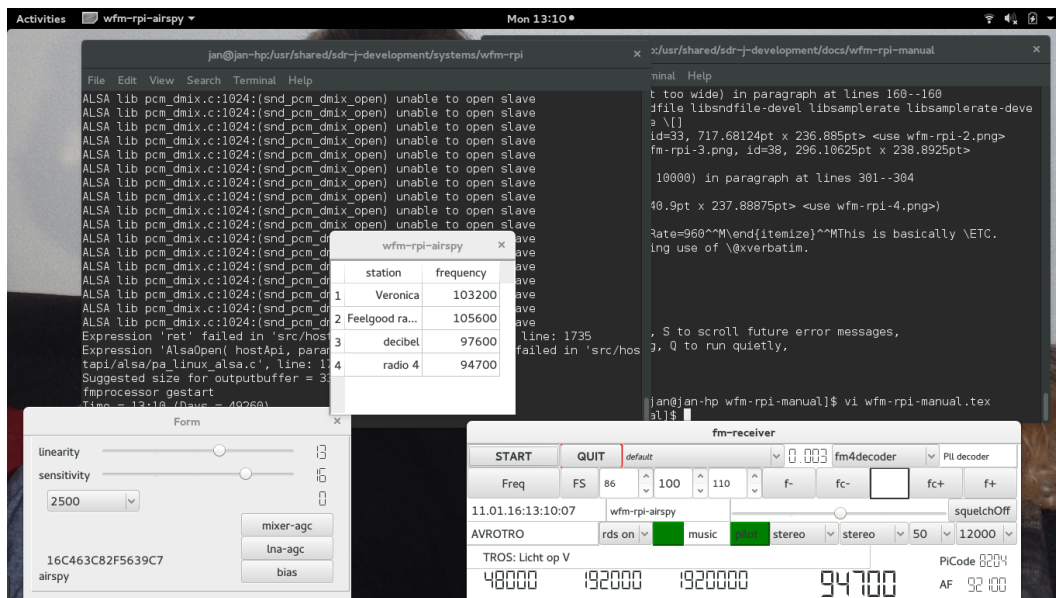
Software for SDR : the WFM-RPI set

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

January 11, 2016

---

1 Introduction

WFM-RPI is a program for decoding FM, using a device like a DABstick, an SDRplay or an AIRSPY. The software is built such that it will run on a decent PC with Linux or Windows as well as on a Raspberry PI 2, hence the name.

To simplify design (and implementation), executables are device specific, i.e. an executable is/can be made for each device.

To ease listening, two features are implemented

- remote listening, if so configured, the audio output is send to port 20040 of the local host,

- a station list can be maintained, i.e. a table where one might add station names for easy lookup and selection.

2 Installation under Windows

Unpacking the zip-file will be in a folder named *windows-bin-dab*. This folder contains the executables of this fm software and the dab software and a number of dlls[1].

Mirics SDRplay For using the SDRplay one has to install the appropriate Mirics driver through a Mirics installer program, to be obtained from the developers at *http://www.sdrplay.com/downloads.html*.

DABstick driver For using the DABsticks one has to download and install the appropriate *rtlsdr.dll* library, and install another usb driver for the RTL2832 based sticks. Dowloading the ".dll" file is best done from the site of the developers *http://sdr.osmocom.org/trac/wiki/rtl-sdr*. Pls ensure that you download the 32 bit version. Place the ".dll" file in the *windows-bin-dab* folder or adjust the searchpath.

The Zadig program is available for installing the required USB library. There are many examples on the internet how to run Zadig. Basically just run the Zadig program with the dabstick inserted in one of the USB ports. The Zadig program (should) detect(s) the DABstick, and will suggest WinUSB as a replacement.

Airspy For running the airspy the airspy.dll, version 1.07, should be in the searchpath, e.g. the *windows-bin-dab* folder. See e.g. https://github.com/airspy/host/releases.

3 Installation under Linux

Unpacking the sources of the distribution is in a single directory *wfm-rpi*

The file *wfm-rpi.pro* contains directives for the qmake program to create a Make-file. For use with cmake, a CMakeList.txt file is included.

3.1 Required packages and Libraries

One needs, next to the GNU compiler suite (g++),

- Qt-4.x or Qt5.y (x at least 7)

- libusb-1.0,

- libportaudio. Ubuntu 14.04 LTS still provides libportaudio-1.18 as standard package. Replace this with 1.19, which is easily done using the standard package handler.

---

[1]It is assumed that common dlls as required for many programs, such as MSVCR100.DLL are available on the system

- libsndfile and libsamplerate,

- libfftw3f, we use floats rather than doubles.

3.2 Ubuntu

For Ubuntu a validity check is given in the shell script below

```
#!/bin/bash
#
echo "Preparing the environment for Ubuntu"
echo "ensure that the udev rules are adapted for the usb devices
echo " "
echo "install packages"
sudo apt-get install gcc g++ \
libqtX-dev
libfftw3-3 libfftw3-dev \
alsa-base libasound2 libasound2-dev alsa-utils libasound2-plugins \
libportaudio2 libportaudio-dev \
libsndfile1 libsndfile1-dev \
libsamplerate0 libsamplerate0-dev \
libusb-1.0-0 libusb-1.0-0-dev
```

(replace libqtX with the name of the qt4 or qt5 lib you want to use).

3.3 Fedora

For Fedora, depending on the distribution, the following script might help

```
#!/bin/bash
#
echo "Preparing the environment for Fedora"
echo " "
echo "install packages"
sudo yum install gcc gcc-c++ \
qt qt-devel \
fftw fftw-devel \
alsa-lib alsa-lib-devel alsa-tools portaudio portaudio-devel \
libsndfile libsndfile-devel libsamplerate libsamplerate-devel alsa-plugins-samplerate \
libusb1 libusb1-devel
```

3.4 Choosing the configuration

Configuring for qmake When using qmake, one should adapt the "wfm-rpi.pro" file to have the appropriate device selected. Selection is determined by uncommenting a line

```
#
# choose ONE device
#
CONFIG += airspy
#CONFIG += sdrplay
#CONFIG += dabstick-new
#CONFIG += dabstick-osmo
```

```
CONFIG += streamer
```

Note that there are two entries for dabsticks. Read the README.DASTICK.

Configuring for cmake   When using cmake, one should adapt the CMakeLists.txt file

```
# if you want support for one of these devices, uncomment ONE of
# set(SDRPLAY true)
set(AIRSPY true)
#
# For the "standard" rtlsdr library (might be in your distribution
# choose DABSTICK-OSMO, for a more experimental one, i.e.
# the one from Leif Albrink, choose DABSTICK-NEW
# set(DABSTICK-OSMO true)
# set(DABSTICK-NEW true)
# but do not uncomment more than one line
set(STREAMER true)
```

The setting of STREAMER indicates whether or not the option of sending the audio output to port 20040 of LocalHost is honored or not.

librtlsdr   In most cases *librtlsdr* is available in the repositories for the distro and can be installed through the mechanisms available with the Linux distribution. Note that in some cases (i.e. Ubuntu 14.04 and Arch Linux on the RPI 2) use of the standard package requires adding the kernel module *dvb_usb_rtl28xxu* to the blacklist. An alternative is to create a library oneself, and edit the file librtlsdr.c by adding a define DETACH_KERNEL_DRIVER.

In case the library is *not* available, 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
```

Mirics SDRplay   Mirics Ltd provides on its site an installer for a downloadable shared library, *libmirsdrapi-rsp-x86_64-1.1.so*. The library will be installed in */usr/local/lib*.

Note however, that older versions are named differently, comment or uncomment

```
 DEFINES        += SDRPLAY_LIBRARY_NEW
```

AIRSPY   Please install "host-1.07" software if you intend to use the AIRSPY. In some cases you have to apply a patch - documented on the internet - to ensure unloading of the appropriate kernel modules.

For the version 1.07 sources, see e.g. https://github.com/airspy/host/releases.

3.5   Qmake

Qmake will use the .pro file as basis. The *wfm-rpi.pro* file contains lines that have shown to be working on Fedora, Ubuntu systems and on Arch Linux (running on my RPI 2).

Other Linux distributions most likely will provide the full set of packages required for building, probably on other locations, in which case the .pro file might need to be adapted to the particularities of the different Linux distributions.

When all libraries (including the corresponding "include" files) are in place, for use with QMake, one executes

```
qmake-qt4  or qmake or qmake-qt5
make
```

The *qmake-XXX* will generate a Makefile, running *make* will generate an executable and put it in *linux-bin*.

3.6 CMake

A CMakeLists.txt file is included with which an executable can be generated. Obviously, the list of required packages does not change, please ensure the packes are there. The CMakeLists.txt assumes that you are running Qt-5. Feel free to adapt the CMakelists.txt file to run with Qt-4, as stated earlier, the software can be compiled with either.

3.7 USB ports

It might be wise to ensure rights for reading and writing usb port and soundcards before running the program. This is usually done through *udev* rules. One may use the instructions given on the aforementioned osmocom page, installation instructions for the SDRplay, the AIRspy of the dabsticks.

4 Running the WFM software

The executable is named *wfm-rpi-xxx.exe* for Windows and *wfm-rpi-xxx* for Linux, where "xxx" is to be replaced by the device name. For Windows *three* executables are precompiled, one for each of the supported devices.

Under Windows the executables are found in the folder *windows-bin-dab*, under Linux in the directory *linux-bin* or any other place where you might have put it.
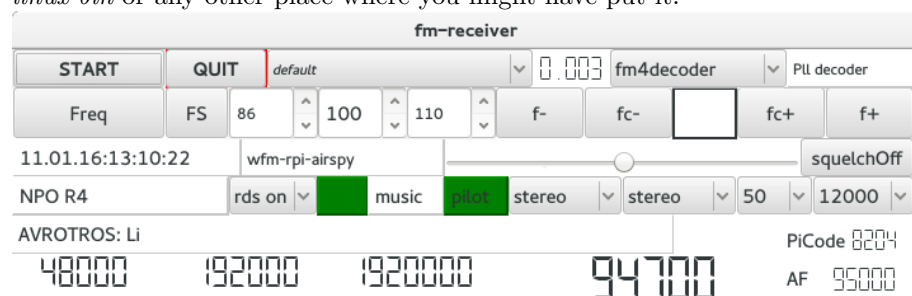
Once the GUI is visible, it is sufficient to click on the "START" button to let the program run. Some controls may be of interest

- as can be expected, clicking on the "QUIT" button will cause the program to exit;

- clicking on the button labeled "default" will allow selection of another sound channel, as the name suggests, the "default" channel is selected by default'

- clicking on the button labeled "fm4decoder" will allow selection of a different demodulation algorithm. In the current version there are 5 different algorithms.

- clicking on the left most button labeled "Freq" allow selection of a frequency. When clicked upon, a separate keypad will show on which the frequency - in Khz or Mhz - can be typed in. After clicking on the "Kz" or "Mhz" button on this keypad, the frequency is selected and the keypad will disappear.

- clicking on the "FS" button allows entering a name *for the currently selected frequency* onto the program list. Selection of a frequency on the program list is by just clicking *once* on the row in the program list where the frequency is.

- Scanning frequencies is by clicking on the fc+ or fc- button, the frequency *range* and the stepsize are detemined by the spinboxes that are - default - set to 86 and 110 Mhz resp. 110 Khz.

- A single step in the frequency is made by the f+ and f- buttons.

- Rds on or off is selected by the designated button.

- The buttons labeled in the picture "stereo" resp "stereo" allow selection of stereo or mono decoding, resp. some combinations of left and right channels.

- the de-emphasis is selected by the button default set to 50,

- the lowpass filter value is set by the rightmost button on this row.

Keypad for frequency selection

The program list

	station	frequency
1	Veronica	103200
2	Feelgood ra...	105600
3	decibel	97600
4	radio 4	94700

Entering data in the program list was described earlier, *removing* an entry from this list is by clicking twice on the entry.

The list is maintained between program invocations.

4.1 The FM ini file

Some 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-wfm-rpi.ini*.

Most of the settings in the ".ini" file follow from settings on the GUI of the FM receiver. The rates for the dabstick and the sdrplay can be set in the ini file

```
dabRate=960
sdrplayRate=960
```

This is basically for testing purposes. Note that the setting for the airspy is fixed: the rate of 2500000 samples/second is in the driver software decimated to 1920000 samples/second.

5 Remote listening

Included in the source tree are the sources for a pretty simple listener in the directory *sound-client*. The soundClient - available as executable in the *windows-bin-dab* folder - is identical to the one used for remote listening to the dab-rpi program.

For its construction a simple ".pro" file exists.

6 Final remarks

The SDR-J software uses a number of libraries, made available through (L)GPL style licenses and parts of the code is based on ideas of others. In all cases attempts are made to indicate the rightfull owner of the copyrights. The software itself is available as is, under a GPL V2 license.

The SDR-J software is essentially a hobby project. It is - obviously - not finished, after all it is software and it is most likely that it will never be finished. Many enhancements (and experiments) are

still waiting to be done. Contributions in any form, e.g. by suggestions for extensions, by contributing to code, or by donations for equipment are welcome.