Frequency Spectrum Monitoring System

A fast channelizer and a network-transparent architecture for Software-Defined Radio

Jan Hrach, Faculty of Mathematics and Physics, Charles University

Bachelor thesis, 2016

Introduction

Software defined radio is an approach to reception and transmission of radio waves. Our work covers the reception part. The principle is to sample the raw input signal as well as possible in an analog-to-digital converter and perform all the signal processing on a general-purpose computer. Thanks to this, simple hardware can be used for multiple purposes by just changing the computer’s software, the development is aplied by debuggers, verstoining systems and similar tools of software engineering and the whole system can make use of existing computer networks.

Reasonably useful universal radio hardware is now available from multiple vendors as RTL-SDR for as low as $35 and even more powerful devices are in the price range reachable to an individual, which makes SDR an attractive technology even to hobbists.

goals

We have analyzed two popular programs used with SDR – GmRadio and Giga projects – and identified some of their weaknesses which we aimed to fix.

GmRadio is a library implementing various digital signal processing functions. One of the implemented algorithms is spectrum channelization, which is extremely useful for reception of frequency division multiplexers. The algorithm was mysteriously slow and we decided to attempt a reimplementaion.

Giga is a GUI tool where the user can explore signal power spectrum on chosen frequencies and then decide to demodulate radio signals with several analog demodulators. We have used Giga as a base-army knife while developing various SDR applications.

When Giga is used with a remote SDR accessed over the network, it transmits the entire SDR baseband. This is usually tens or even hundreds of megabits per second, which makes it impossible to use over the internet. We have decided to design a completely new client-server architecture with a more efficient communication protocol. We then implemented some other new features which we deemed useful.

The channelizer

One channel from a frequency division multiplexer (FDM) can be selected by hardware filtering it and decimating it. However, modern FDM networks use tens of channels at once and using this process separately for every channel is computationally intensive.

We have implemented the algorithm in the Multirate Signal Processing for Communication Systems book, which uses fast Fourier transform to compute one sample of all channels at once in O(N log N) instead of O(N^2) for the naive approach where N is the length of the FFT filter and n is the number of channels.

Usability

We have implemented a simple servlet interface in our program. The interface allows the user to obtain signal power of the channels, select which channels to output and set frequency correction, which allows uninterrupted reception even with unstable clock.

The SDR tool

We have created a client-server architecture to access SDR hardware over the network.

The server

The server is a C program reading data from SDR and exposing a TCP interface. Thanks to the use of libm64r hardware abstraction library, practically all available SDR devices are supported.

The server accepts connections like “setup streaming channel of width X on frequency Y”. It also computes power spectrum of the baseband and sends it to the client.

The main advantage against other solutions that use SDR over the network is that only selected narrowband channels are transmitted, which saves bandwidth. The remote access to the SDR is therefore possible only over lines with only a few MBps.

Multiple clients can connect to the server at once and all changes made by one of the clients are automatically sent to the others.

pluggable demodulators

Giga channels

Automatic frequency correction

The software compensates frequency drift automatically.

Intelligent squelch

The squelch does not care about absolute power level but compares the peak with noise floor. Additionally, when the squelch opens, a finite bit of history is replayed, so the beginning of the conversation is not lost.

Efficient network protocol

The server sends data from the radio and sends to the client only the required information – like waterfall data and narrowband channels. Less precise transport format can be selected instead of loss to save even more bandwidth.

Thanks to this, Kukuruza can be readily used to access remote SDRs over common internet lines.

Further information

Author: Jan Hrach, janhrach@kar.tu
Supervisor: Mgr. David Klaudařek, Ph.D.
Institute of Formal and Applied Linguistics

References

Pedro J. Henriques, Multiple Signal Processing for Communication Systems.

Source code of our software

https://bitbucket.org/rt64/project/sdr
https://bitbucket.org/rt64/code

Histogram

Plotting histogram of absolute values of samples helps you to correct set the gain of the radio and identify clipping and interference.

input buffer

The entire SDR baseband is kept for about a minute if a short interesting signal appears, the buffer can be dumped to a file for later offline analysis.