Spectrum Sensing to Achieve Frequency Rendezvous using Xilinx FPGAs

– DySPAN 2008 Demonstration –

Jörg Lotze[†], Barış Özgül[†], Suhaib A. Fahmy[†], Juanjo Noguera[*], Linda Doyle[†] and Robert Esser[*]

[†]*Centre for Telecommunications Value-chain Research*
University of Dublin, Trinity College
{jlotze,ozgulb,suhaib.fahmy,linda.doyle}
@tcd.ie

[*]*Xilinx Research Labs*
Xilinx Ireland
{juanjo.noguera,robert.esser}
@xilinx.com

1 Introduction

Nodes in dynamic spectrum access networks opportunistically use free areas of spectrum, also called *white spaces*, for communication. A common problem in such networks is achieving rendezvous between a transmitter and a receiver, *i.e.*, establishing a communication link on the same frequency band. One possible solution is to use a common control channel where each transmitter can propagate information about which channel it is using. However, the frequency of this dedicated control channel has to be known *a priori* by each radio connecting to the network, which generally means it is fixed. It also represents a single point of failure for the whole network and has scalability issues. A better and more flexible solution is to establish rendezvous without a control channel.

This demonstration shows rendezvous without a control channel. The receiver uses spectrum sensing techniques to scan for the transmitter, changes its frequency accordingly, and establishes the link. We use a coded Differential Quadrature Phase Shift Keying (DQPSK) link to transmit real-time video data. The transmitter frequency can be changed manually. When the receiver loses the connection, it searches the spectrum for the transmitter. Once found, it establishes a link, and continues to play the video. All baseband signal processing for both the transmitter and receiver is running on a Xilinx Virtex II Pro Field Programmable Gate Array (FPGA).

2 Sensing Algorithm

Energy detection (ED) is an established method in spectrum sensing [1] which can be employed by a receiving node to estimate the unknown frequency of a transmitter. *Cyclic-feature detection* [2] (CD) is another popular technique that can be applied since most telecommunications signals can be inherently modelled as cyclostationary random processes. ED is more generic as it does not require any signal-specific information, while CD is more robust at low signal-to-noise ratios.

The sensing algorithm in this demonstration is based on ED. However, the receiver can optionally use CD if the so-called *cyclic frequency* for the signal of interest is provided. The algorithm works as follows:

1. Apply the Fast Fourier Transform (FFT) to the received discrete-time data samples over a certain observation window.
2. Use the FFT to calculate an average power spectral density (PSD) function for ED (and an average spectral correlation function [2] (SCF) for CD).
3. Moving average filtering of the PSD (and SCF in case of CD).
4. Use a decision threshold to detect the signal and estimate the carrier frequency by finding the centre frequency of PSD (and SCF if CD is applied). The threshold value is determined as in [3].

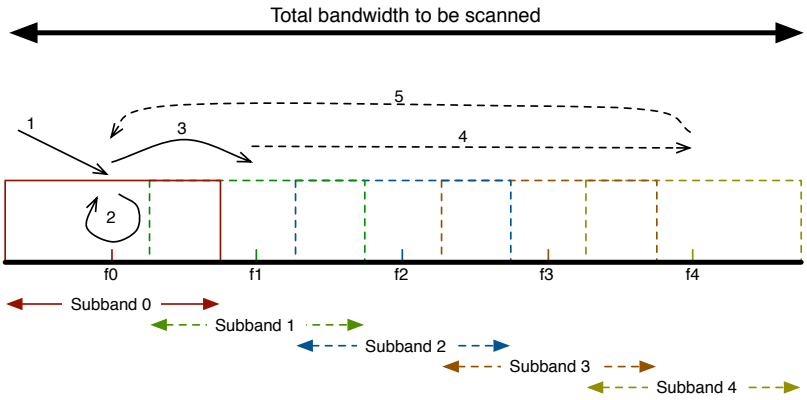

Figure 1: Carrier frequency acquisition mechanism.

The mechanism used to scan for the transmission frequency is illustrated in Figure 1. The overall frequency band of interest is divided into overlapping subbands where f_i denotes the centre frequency of the ith subband. Overlapping subbands are employed to prevent mis-detection at the edges when only part of the signal is received. Note that the sensing algorithm is capable of analysing the whole frequency band at once; the subbands are necessary due to the bandwith limitation of the Radio Frontend.

Referring to the numbers in Figure 1, the receiver functionality can be summarised as follows:

1. Receiver frequency is set to f_0 and scanning starts at subband 0.
2. Receiver searches for the signal to determine the carrier frequency. If a signal is detected, and the receiver estimates the carrier frequency by exploiting the sensing algorithm described above, locks onto the signal, and checks a known preamble after demodulation and decoding to determine if the signal is correct. If the detection is successful, the receiver periodically checks whether the signal is lost or not (possible if the carrier frequency changes).
3. In the case of no signal, incorrect signal or lost signal, the receiver scans the next subband.
4. The search continues across all subbands in the case of no signal, incorrect signal, or lost signal.
5. The receiver frequency is changed back to f_0 after scanning the full frequency band of interest.

3 Demonstration Setup

The software radio platform used for this demonstration is based on IRIS [4] which has been extended to run under embedded Linux on an FPGA and support hardware components in the logic fabric. The building blocks of a radio are Digital Signal Processing (DSP) components, either implemented in software using C++, on in hardware using VHDL. Examples for such components are modulators, decoders, or filters. Each of the components has a set of parameters which allow for re-use in different radio configurations. A subset of these parameters can be changed at run-time, either manually or by a radio controller.

The implementation platform is the Xilinx Xilinx University Program (XUP) Development Board which hosts a Xilinx Virtex II Pro FPGA. We use the Universal Software Radio Peripheral (USRP) radio frontend as air interface [5]. The transmit chain receives UDP packets, through ethernet, containing video data, streamed from a PC using the VideoLAN VLC Player. This is processed on the transmission chain on the FPGA, then transmitted by the USRP. At the receiver, the signal is processed by the reception chain in the FPGA, then passed, through UDP, to a VLC Player window on a PC which plays the video. An overview is shown in Figure 2. We employ the FPGA architecture as presented in our previous paper [6], with the Linux OS running on the embedded PowerPC (PPC) of the Virtex II pro FPGA.

The radio chains for this application are shown in Figure 3. Components implemented in hardware are shown in white. The *Differential QPSK modulator* and *DQPK demodulator* map information dibits

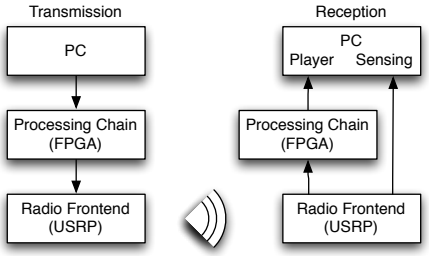

Figure 2: Demonstration Setup.

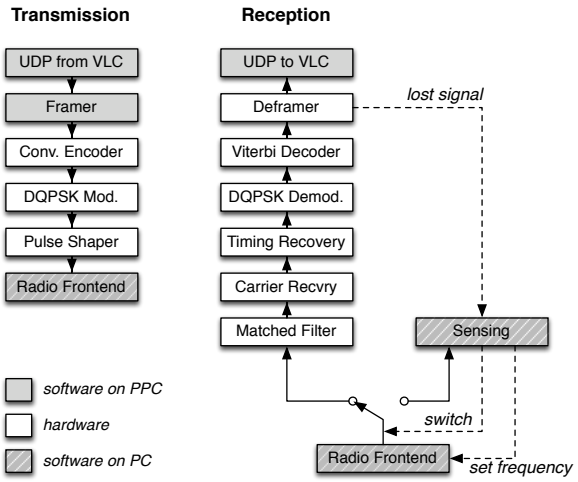

Figure 3: Transmitter and receiver radio chains.

(2-bit units) into phase changes on an I/Q plot, and vice versa, respectively. A root raised-cosine filter is used at both the transmitter (*Pulse Shaper*) and receiver (*Matched Filter*) to shape the pulses and thus reduce transmission bandwidth. When a modulated signal is received, the frequency of the local oscillator is typically off by some margin which is corrected by *Carrier Recovery*. *Timing Recovery* finds the correct sampling points for each symbol, eliminating the effects of timing shifts. We use the standard Xilinx *Convolutional Coder* and *Viterbi Decoder* cores provided as part of CoreGen. The *Deframer* correlates a fixed 64-bit preamble (inserted bythe *Framer* at the transmitter), with the streaming data, in order to identify the start of a valid frame of data. The *Sensing* component is implemented in software on a PC. It uses the algorithm mentioned in Section 2 to locate the transmitter frequency, then enables the FPGA-based processing chain for reception.

The demonstrator described here can be seen in operation at the IEEE Symposia on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), October 2008, Chicago, Illinois, USA.

References

[1] A. Ghasemi and S. E. Sousa, "Spectrum sensing in cognitive radio networks: Requirements, challenges, and design trade-offs," *IEEE Communications Magazine*, vol. 46, pp. 32–39, April 2008.

[2] W. A. Gardner, "Signal interception: A unifying theoretical framework for feature detection," *IEEE Transactions on Communications*, vol. 36, pp. 897–906, August 1988.

[3] T. J. O'Shea, T. C. Clancy, and H. J. Ebeid, "Practical signal detection and classification in gnuradio," in *SDR Forum Technical Conference (SDR)*, 2007.

[4] P. MacKenzie, "Software and reconfigurability for software radio systems," Ph.D. dissertation, University of Dublin, Trinity College, Ireland, 2004.

[5] *Universal Software Radio Peripheral – The Foundation for Complete Software Radio Systems*, Ettus Research LLC, Mountain View, California, USA, Nov. 2006. [Online]. Available: http://www.ettus.com/downloads/usrp_v4.pdf

[6] J. Lotze, S. Fahmy, J. Noguera, L. Doyle, and R. Esser, "An FPGA-based cognitive radio framework," in *Irish Signals and Systems Conference (ISSC)*, Jun. 2008, pp. 138–143.