An FPGA-based Autonomous Adaptive Radio

Jörg Lotze*, Suhaib A. Fahmy*, Juanjo Noguera† and Linda E. Doyle*

*CTVR, Trinity College Dublin, Dublin 2, Ireland
jlotze@tcd.ie, suhaib.fahmy@tcd.ie, linda.doyle@tcd.ie

†Xilinx Research Labs, 1 Logic Drive, Citywest Business Campus, Saggart, Co. Dublin, Ireland
juanjo.noguera@xilinx.com

ABSTRACT

We show the use of a novel, high-level FPGA-based cognitive radio framework in implementing a demonstration of video transmission over a frequency-agile radio link. The demo uses spectrum sensing to detect the transmission frequency and adapts the forward error correction scheme to the channel conditions autonomously.

1. INTRODUCTION

Despite field-programmable gate arrays (FPGAs) providing an enticing platform for high performance embedded systems, they remain difficult to use. FPGAs allow for the design of custom datapaths that can exploit inherent parallelism in algorithms, resulting in significant speedup over software. Another key capability of FPGAs, particularly useful in the area of cognitive nodes, is partial reconfiguration, where parts of the FPGA application can be replaced at runtime while other parts continue to function. This can allow high performance cognitive nodes with many configuration options to be realised efficiently on a small, lower-powered device. This demo presents the use of a novel FPGA-based cognitive radio framework that allows radio designers with no hardware experience to leverage the performance and flexibility advantages afforded by FPGAs.

Autonomous, self-organising networks are built upon smart nodes that can establish communication with peers without the presence of a fixed network control channel [3]. Nodes must be able to not just communicate using the necessary protocols, but to assess current communication activity within the network. This requires a radio platform that provides sufficient computational power for signal analysis and complex communications protocols while also being flexible; able to modify some communication parameters or effect a complete change in processing behaviour. One technology that is ideally placed to address this need is cognitive radio.

This demonstration showcases a flexible, embedded FPGA-based cognitive radio framework that we have developed [1]. This framework enables the high-level design of cognitive radio systems, with abstraction between control and hardware implementation. This paper presents the following:

- Our FPGA-based cognitive radio framework allows easy design of adaptive nodes that exploit the performance of hardware and maintain the flexibility of software.
- An autonomous network node discovering an unknown transmission frequency using spectrum sensing.
- Adaptation to channel conditions by switching the error correction scheme, minimising power consumption and resource usage while maintaining link quality.

2. COGNITIVE RADIO FRAMEWORK

We have developed a framework for implementing cognitive radio nodes on FPGAs based on IRIS [1]. Radio chains are described using XML, stitching together components from a pre-existent library, containing both hardware (FPGA) and software (general purpose processor) components. The framework we developed handles all hardware reconfiguration automatically, with the radio designer determining the autonomous control algorithm in the *Decision Engine*, in software. It does not depend on the implementation of the components (i.e. software or hardware), and accesses all interfaces at an abstracted level. This allows radio designers to take advantage of the processing power and run-time reconfiguration capabilities afforded by FPGAs, without the need for hardware design experience.

The FPGA runs an embedded Linux operating system on a embedded PowerPC core, enabling the execution of the IRIS engine and software components. Interfacing to hardware is managed through a Linux driver.

3. DEMO APPLICATION

In this application, the transmitter starts transmitting a video stream at an unknown arbitrary frequency, using DQPSK modulation and convolutional coding. A receiver wakes up, and enters detection mode, where it uses spectrum sensing to locate transmitted signals within the frequency band. Once a transmission has been found, the radio reconfigures into reception mode and attempts to demodulate the signal. If this fails, the receiver switches back to sensing mode and searches for another signal. Once a correct transmission is found, the receiver continues to demodulate.

The transmitter can change frequency during transmission. When the receiver detects that the signal is lost, it switches back to sensing mode and searches for the transmit signal again, as described above.

During reception mode, the receiver monitors the Bit Error Rate (BER), and based on this determines when to change the strength of the code, and when coding can be switched off entirely. This adaptation is continuous, based on channel conditions. It results in power and resource savings while maintaining the link quality.

Baseband processing at both the transmitter and receiver is performed on an FPGA. Note that changes in the cod-

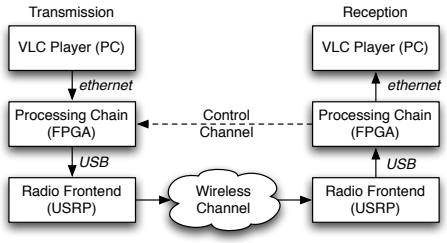

Figure 1: Demonstration Setup.

ing scheme, and switching to sensing require considerable changes in the instantiated FPGA hardware. These changes are achieved by applying a new configuration through run-time partial reconfiguration, which is managed, transparently, by the framework we are showcasing.

We use the Xilinx University Program Virtex II pro development board for this application, and the Universal Software Radio Peripheral (USRP) as radio frontend. VLC Player, running on a PC connected through Ethernet, is used as the source and sink for the video transmission. An overview of the demo setup is shown in Figure 1.

3.1 Sensing for the Transmitter Frequency

The sensing mode consists of two components, a Power Spectral Density (PSD) estimator component, implemented in hardware (FPGA logic), and a peak detection component, implemented in software (embedded PowerPC).

We use the thresholding method described in [2] to find peaks in the PSD after correlating with the expected spectral footprint. These are reported to the Decision Engine, which tunes the frequency of the radio frontend and switches to reception mode.

3.2 Adapting the Coding Scheme

The use of forward error correction (convolutional codes with code rate 1/2) allows for more robust transmission. On the other hand, using codes with lower constraint length, or no coding at all, result in considerable power savings on the FPGA. Running the system without coding also doubles the data rate. We allow three configurations of the processing chains in this demo: code polynomials with constraint lengths 8 and 6, and no coding.

We set a target BER, and use this to determine thresholds for switching codes, balancing robustness with efficiency. The Viterbi decoder provides a BER estimate. If this decreases below a threshold, the receiver signals to the transmitter over a feedback channel that a less robust code can be used, and reconfigures itself to that code. If the channel conditions are good enough to maintain the target BER without coding, both transmitter and receiver reconfigure to no-coding. In this configuration, the current BER is estimated based on the frame error rate computed in the Deframer. The system continues to operate in this fashion and therefore adapts to the current conditions autonomously.

3.3 Implementing Adaptation

The receiver Decision Engine in this demo is a hierarchical state machine. At the top level, it has two states: sensing and reception. Transitions between these states are triggered if the signal is lost, or if possible transmission frequencies are found, respectively. The reception state contains

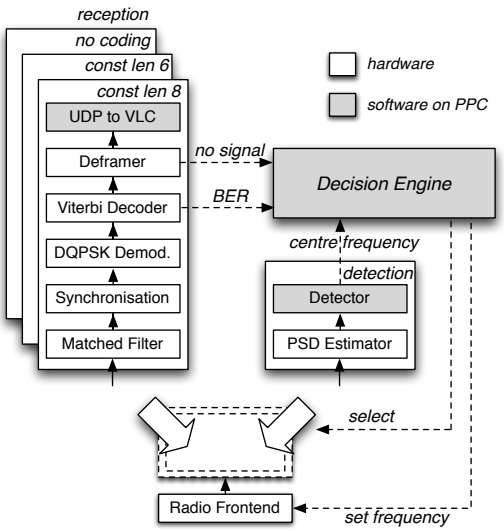

Figure 2: Reception and detection chains with Decision Engine and control signals. The Decision Engine selects which of the chains are loaded into the system and connected to the radio frontend.

three internal states, one for each coding method. Transitions between the internal states are triggered when the BER is above or below predetermined thresholds.

The configuration of the receiver is illustrated in Figure 2. The transmitter is similar, but without detection chain.

4. CONCLUSION

This demonstrator showcases the ease of designing cognitive nodes using our cognitive radio framework, which allows radio designers with no hardware experience to leverage the performance and flexibility advantages of FPGAs.

Spectrum sensing and adaptive forward error correction are two examples of features required in cognitive nodes, which are the building blocks of autonomous, self-organising networks in future wireless systems.

The demonstrator described shows a wireless video streaming application over an adaptive, frequency-agile communications channel. The receiver can find the transmission frequency without prior information, at start-up or when the frequency changes. The forward error correction scheme adapts to current channel conditions at run-time, by partially reconfiguring the FPGA hardware transparently, as required.

5. REFERENCES

[1] S. Fahmy, J. Lotze, J. Noguera, L. Doyle, and R. Esser. Generic software framework for adaptive applications on FPGAs. In *IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM)*, Napa, CA, Apr. 2009.

[2] T. J. O'Shea et al. Practical signal detection and classification in GNU radio. In *SDR Forum Technical Conference (SDR)*, Denver, CO, Nov 2007.

[3] P. D. Sutton et al. Cyclostationary signatures in practical cognitive radio applications. *IEEE J. Sel. Areas Commun.*, 26(1):13–24, Jan. 2008.