# Cognitive Radio Universal Software Hardware

George Eichinger
MIT Lincoln Laboratory
Lexington, Massachusetts
george.eichinger@ll.mit.edu

Kaushik Chowdhury, Miriam Leeser
Department of Electrical and Computer Engineering
Northeastern University
Boston, Massachusetts
krc@ece.neu.edu, mel@coe.neu.edu

***Abstract*—The FPGA is an integral component of a software defined radio (SDR), which provides the needed reconfigurability for dynamically adapting its transceiver and data processing functions. The current state of the art in SDR design relies on complete processing of the raw samples at the host computer, thereby impacting time critical tasks. Instead, we propose to move the processing closer to the front-end by interfacing an external FPGA board with the SDR. Our architecture, called CRUSH, is composed of a Xilinx ML605 FPGA Development Board connected to an Ettus Research USRP N210 through an HDL framework, with a custom interface to allow flexible data transfer between them and independent programming capability on the two devices. Our test scenario for spectrum sensing, a key step in determining channel availability before transmission in dynamic spectrum access networks, indicates significant benefits: CRUSH can implement FFTs at 100x improvement, and can perform a complete sensing cycle 10x faster than legacy SDRs for large FFT sizes that enable wideband sensing. By potentially reducing the load on the host, and allowing a powerful FPGA extension for off the shelf devices, CRUSH will enable advances in both protocol design and radio hardware. For DySPAN we will show a live demonstration of the CRUSH platform performing spectrum sensing. For this demo CRUSH will be receive only and operate in the 50 MHz to 2.2 GHz region. A laptop will be connected to CRUSH via Ethernet and will display the live results via a MATLAB GUI.**

***Index Terms*—Cognitive Radio; Spectrum Sensing; FPGA;**

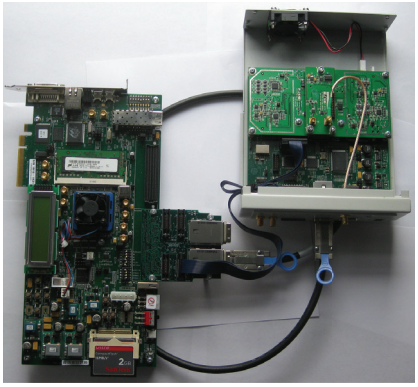

Fig. 1: CRUSH Platform

**Hardware Overview.** The CRUSH [1] system (Fig. 1) is comprised of three components (1) an Ettus Research USRP N210 software defined radio, (2) a Xilinx ML605 Development Board and (3) a custom interface board (CIB) to route signals between the USRP and the ML605. The interconnects between the boards are commercial off the shelf (COTS) cables. Other FPGA platforms exist such as WARP [2] from Rice University and the SDC Testbed [3] from Drexel. CRUSH is unique because it decouples the expensive, fast developing FPGA technology from the radio and configurable hardware; this allows either the FPGA or the radio to be updated independently.

A goal of the CRUSH platform is to minimize the impact on the existing USRP. The modified USRP firmware is designed to function with or without the presence of the ML605. The USRP firmware has been modified so that the data from the ADC lines are forked and sent over the interface between the USRP and the ML605 as well as to the host. The original Ethernet link is still intact so the USRP can be used as a standard radio in parallel with the accelerator.

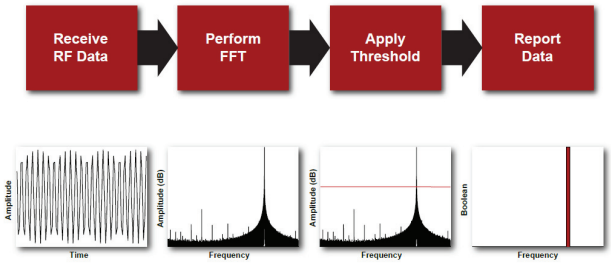


Fig. 2: Spectrum Sensing Algorithm

**Algorithm.** CRUSH is currently being used as a spectrum sensing accelerator. The spectrum sensing algorithm is an FFT followed by a threshold operation (Fig. 2). The first step is receiving and digitizing the RF data via the ADC. The result is the time domain digital representation of the RF data. Next we perform an FFT to convert the time domain data to the frequency domain. FFTs are referred to by their point size which represents how many output bins the FFT will produce; each bin represents the energy of a fraction of the total bandwidth. Once we have energy values for each bin, we apply a threshold. If the value of the bin exceeds this threshold, it is assigned 1. If it is below the threshold it is assigned 0.

**Results.** One of the main reasons for migrating to hardware based algorithm implementations for radio is improving latency. To prove that CRUSH reduces latency, we performed

This work is sponsored by the Department of the Air Force under Air Force Contract FA8721-05-C-0002. The opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

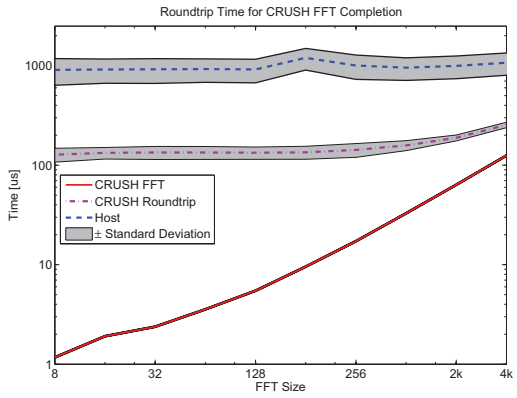


Fig. 3: Comparison of all times

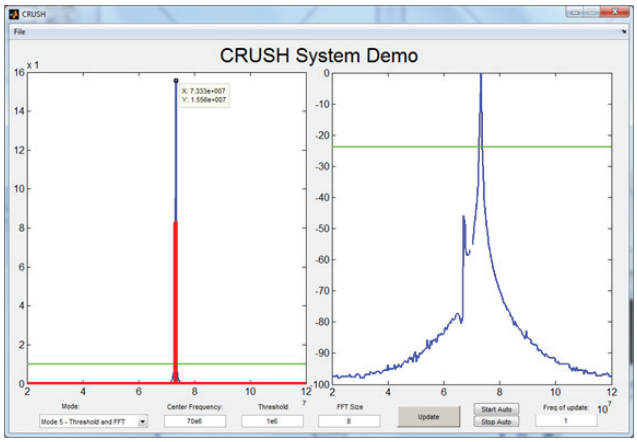


Fig. 4: MATLAB GUI for CRUSH Demonstration

an end to end test examining FFT sizes from 8 to 4096 and averaged over 400 packets per point size. We also did the same test purely in software on a 2.8 GHz processor as a comparison. Additionally, the timing for just the FFT in hardware was also included (Fig. 3). The blue dashed line represents the time the host takes to complete the given FFT size. The purple dash-dot line represents the end to end or roundtrip time on CRUSH and the solid red line represents the FFT completion time of CRUSH. Both the x and y axes are in log scale; the FFT size is on the x-axis and time in $\mu$s is on the y-axis. The gray filled portion represents $\pm$ one standard deviation. Since both the host and roundtrip results included UDP packet transmission times, there were large standard deviations. Analyzing this graph, we can see that CRUSH is approximately 10x faster than a host only version of the same spectrum sensing algorithm.

**Resource Allocation.** After the implementation of the CRUSH framework and the spectrum sensing application that uses a simple threshold based energy detection, the ML605 still has ample space to implement additional algorithms. The CRUSH framework uses 1.2% Slice Registers, 7.8% Block RAM and 0.4% DSP48 blocks. Spectrum sensing adds 2.6% Slice registers, 26% Block RAM and 8.3% DSP48 blocks. The majority of the spectrum sensing hardware is Block RAM used for calculating the FFT. Overall with both the framework and spectrum sensing implemented, the system still has 96.1% Slice Registers, 66.3% Block RAM and 91.2% DSP48 blocks free. One candidate for using this space is to implement additional MAC functionality in reconfigurable hardware. This could lead to novel split-MAC designs, with time critical functions of the MAC performed on the FPGA, and policy decisions undertaken by the host.

**Demonstration.** For our demonstration of CRUSH, we will couple the RF input of the USRP to either a signal generator or an antenna. The CRUSH MATLAB GUI (Fig. 4) will visually show that CRUSH is correctly identifying the occupied spectrum space. The screen capture in Figure 4 shows a 73 MHz tone injected into CRUSH. The plot on the left is a magnitude plot and the right shows the same data plotted in dB. The blue line is the output FFT values from the CRUSH platform. The green line is the user selected threshold value. The red line represents the processed channel occupancy data. In this screen capture the system is correctly identifying the 73 MHz tone as occupied spectrum. On the bottom of the GUI the user can modify the parameters of the demo including the center frequency, the threshold value and the FFT size.

We will also demonstrate the impact of processing information gathered by the SDR closer to the radio front-end through the CRUSH platform in terms of end to end latency. Our setup shall show that not only does CRUSH correctly identify the presence of the signal, but does so 10x faster than the case with only the host being present. We will use the MATLAB based signal reconstruction to confirm the shape of the sensed signal, as well as use a synchronized digital clock to indicate the time in which the entire operation was completed.

**Conclusion.** CRUSH combines a powerful FPGA with a versatile RF front end. To enable this, we created a custom interface board that allowed high speed data transfer between two widely used COTS platforms. The signal processing is now closer to the receiver allowing for implementation of high speed, real time algorithms. This also decouples the FPGA computing resources from the SDR allowing for either to be updated independently. We implemented spectrum sensing as the first application on CRUSH. Spectrum sensing is now running at 100x the performance for FFT sizes of interest to cognitive radios. We have reduced the load on the host computer by running this algorithm in hardware and we have kept the system fully configurable. In the future we plan to integrate CRUSH into existing research on cognitive radio.

## References

[1] Eichinger, G., Chowdhury, K., Leeser, M., "CRUSH: Cognitive Radio Universal Software Hardware," in *Field Programmable Logic and Applications (FPL)*, August 2012.

[2] Amiri, K. et al., "WARP, a Unified Wireless Network Testbed for Education and Research," in *Microelectronic Systems Education*, June 2007, pp. 53 –54.

[3] Shishkin, B. et al., "SDC testbed: Software defined communications testbed for wireless radio and optical networking," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, May 2011, pp. 300 –306.