SDR - Spectrum Sensing

by Christina Baaklini, Michael Collins, and Nicole DiLeo
Overview

- FPGA Sequential Circuit Design
- Scanning Receiver Readings
- Implementation of MATLAB Code in C++
FPGA Sequential Circuit Design

- This week, we continued learning VHDL and were able to create basic sequential circuits and implement them onto the Zedboard.
- We ran into a problem involving binary adder, in which it would not output correct values until 200ns. However, we fixed this by adding a short reset cycle at the start of the testbed.
Scanning Receiver Readings

- Modified the wiserd “timesamplestofile” module to log receiver carrier frequencies
- Wrote a MATLAB function to read in number of samples taken at each carrier frequency
- Adapted MATLAB spectrogram script to plot frequency spectrum based on carrier

```cpp
void TimeSamplesToFile::Proc(std::vector<std::complex<float> > *recv_buffer) {
  if (all_set_) {
    file_mtx_.lock();

    if (current_carrier_ == radio_parameter_map_->at("uhd_rx_freq")) {
      samps_at_carrier_ += recv_buffer->size();
    }
    else {
      freqs_ << current_carrier_ << ' ';
      freqs_ << samps_at_carrier_ << '\n';
      current_carrier_ = radio_parameter_map_->at("uhd_rx_freq");
      samps_at_carrier_ = 0;
    }

    data_.write((char *) &recv_buffer->front(),
                recv_buffer->size() * sizeof(float) * 2);
  }
  file_mtx_.unlock();
}
```
Implementation of MATLAB Code in C++

```matlab
function [ffts,moving_avg,peaks]=spectro(m,c_fr,s_fr,k,o,w,avg)
% m = row matrix of IQ samples
% c_fr = carrier frequency
% s_fr = sampling frequency
% k = size of FFTs
% o = overlap between FFTs (between 0 and 1)
% w = row matrix of size k to be used as a window function
% avg = number of ffts to be averaged together
o = 1-o; N = numel(m);
start = @(j) k*o*j+1; % beginning of each FFT
stop = @(j) start(j)+k-1; % end of each FFT
ffts = [];
i = 0;
fprintf('Generating FFTs ... ');
while stop(i) < N
    s = m(start(i):stop(i));
    s2 = w.*s;
    s2f = fft(s2,k);
    s2f_shift = fftshift(s2f);
    ffts = [ffts;s2f_shift];
i = i+1;
end
fprintf('Done
');
```

```c++
void fft_avg::spectro()
{
    overlap_ = 1-overlap_;
    unsigned int N = iq_samples_.size();
    int index = 0;

    vector<complex<float> > s;
    vector<complex<float> > s2;
    empty_vector_.resize(fft_size_);

    out_ = (fftw_complex*) &empty_vector_.front();
    plan_ = fftw_plan_dft_1d(fft_size_, in_, out_, FFTW_FORWARD, FFTW_ESTIMATE);

    while (stop(index, fft_size_, overlap_) < N)
    {
        for (int i = start(index, fft_size_, overlap_); i <= stop(index, fft_size_, overlap_); i++)
        {
            s.push_back(iq_samples_[i]);
            s2.push_back((window_[i])*(s[i]));
        }
        in_ = (fftw_complex*) &s2.front();
        fftw_execute(plan_);
        fft_data_.push_back(empty_vector_);
        index++;
    }
}
```
Next Week

- Continue learning more advanced topics in VHDL, for example: Arrays and Physical Types. Try to utilize the 7-segment LEDs on ZedBoard.
- Continue implementing MATLAB spectrogram script in C++
  - Plotting FFTs
  - Moving Average Filter