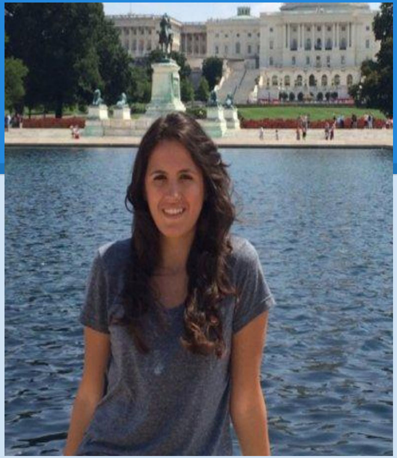
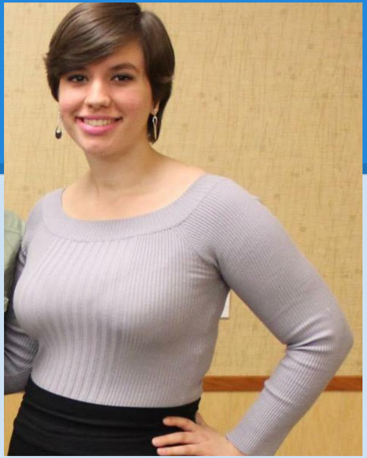
SDR - Spectrum Sensing

by Christina Baaklini, Michael Collins, and Nicole DiLeo





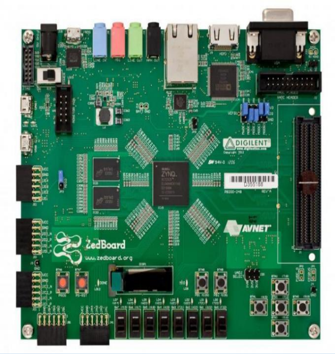
Overview

- **FPGA Programming**
- **Experimentation with ORBIT Grid**
- **Filtering and Peak-Finding in MATLAB**

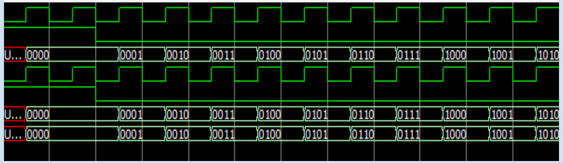
VHDL – VHSIC Hardware Description Language (Very High Speed Integrated Circuit)

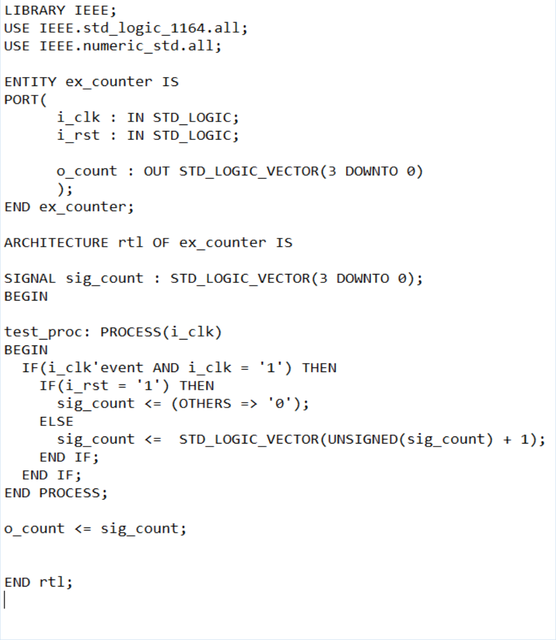
VHDL is a hardware description language that can be used to model a digital system. The digital system can be as simple as a logic gate or as complex as a complete electronic system. VHDL provides 5 design units:

- Entity Declaration
- Architecture Body
- Configuration Declaration
- Package Declaration
- Package Body



ZedBoard (board based on Xilinx) containing FPGA in the center.

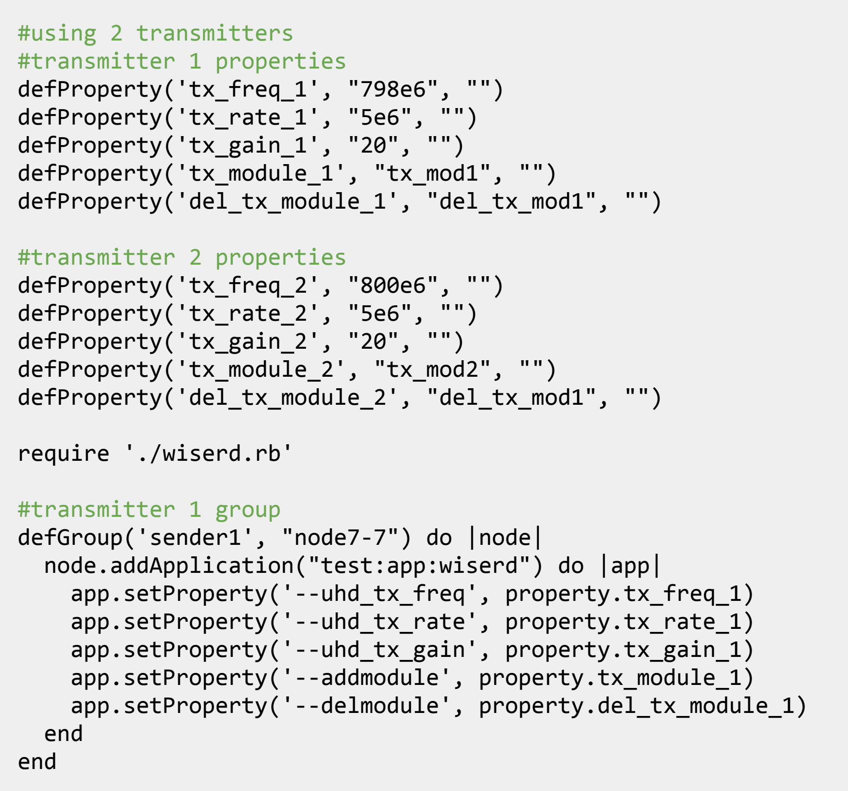

U... 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010
U... 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010
U... 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010



```
LIBRARY IEEE;
USE IEEE.std_logic_1164.all;
USE IEEE.numeric_std.all;

ENTITY ex_counter IS
PORT(
      i_clk : IN STD_LOGIC;
      i_rst : IN STD_LOGIC;

      o_count : OUT STD_LOGIC_VECTOR(3 DOWNTO 0)
      );
END ex_counter;

ARCHITECTURE rtl OF ex_counter IS

SIGNAL sig_count : STD_LOGIC_VECTOR(3 DOWNTO 0);
BEGIN

test_proc: PROCESS(i_clk)
BEGIN
  IF(i_clk'event AND i_clk = '1') THEN
    IF(i_rst = '1') THEN
      sig_count <= (OTHERS => '0');
    ELSE
      sig_count <=  STD_LOGIC_VECTOR(UNSIGNED(sig_count) + 1);
    END IF;
  END IF;
END PROCESS;

o_count <= sig_count;


END rtl;
```

Vivado (Xilinx Tool) and QuestaSim are used as simulators for VHDL.

Experimentation with ORBIT Grid

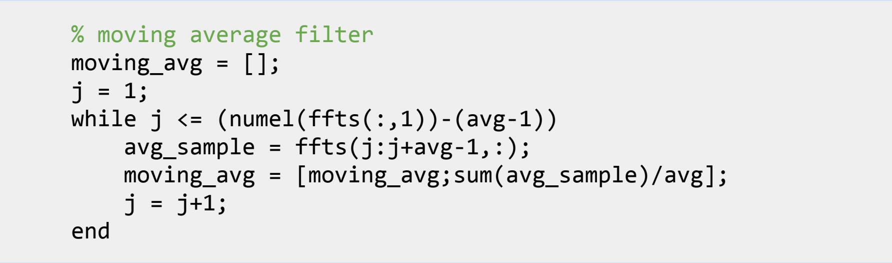


```
#using 2 transmitters
#transmitter 1 properties
defProperty('tx_freq_1', "798e6", "")
defProperty('tx_rate_1', "5e6", "")
defProperty('tx_gain_1', "20", "")
defProperty('tx_module_1', "tx_mod1", "")
defProperty('del_tx_module_1', "del_tx_mod1", "")

#transmitter 2 properties
defProperty('tx_freq_2', "800e6", "")
defProperty('tx_rate_2', "5e6", "")
defProperty('tx_gain_2', "20", "")
defProperty('tx_module_2', "tx_mod2", "")
defProperty('del_tx_module_2', "del_tx_mod1", "")

require './wiserd.rb'

#transmitter 1 group
defGroup('sender1', "node7-7") do |node|
  node.addApplication("test:app:wiserd") do |app|
    app.setProperty('--uhd_tx_freq', property.tx_freq_1)
    app.setProperty('--uhd_tx_rate', property.tx_rate_1)
    app.setProperty('--uhd_tx_gain', property.tx_gain_1)
    app.setProperty('--addmodule', property.tx_module_1)
    app.setProperty('--delmodule', property.del_tx_module_1)
  end
end
```

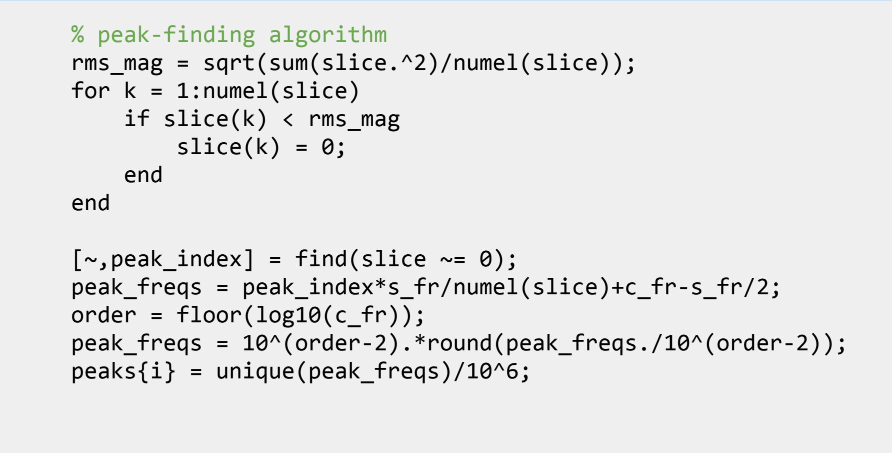
- **Transmitting signals from two radios and receiving at one**
- **Defining multiple groups in OEDL script as opposed to one**
- **fftmovingavgoml module**
- **timesamplestofile module**

Filtering and Peak-Finding in MATLAB

- **Implemented moving average filter to reduce noise**
- **Added algorithm for finding peaks in the frequency spectrum**
- **Can now guess with reasonable accuracy the transmitter frequencies**



```matlab
% moving average filter
moving_avg = [];
j = 1;
while j <= (numel(ffts(:,1))-(avg-1))
    avg_sample = ffts(j:j+avg-1,:);
    moving_avg = [moving_avg;sum(avg_sample)/avg];
    j = j+1;
end
```



```matlab
% peak-finding algorithm
rms_mag = sqrt(sum(slice.^2)/numel(slice));
for k = 1:numel(slice)
    if slice(k) < rms_mag
        slice(k) = 0;
    end
end

[~,peak_index] = find(slice ~= 0);
peak_freqs = peak_index*s_fr/numel(slice)+c_fr-s_fr/2;
order = floor(log10(c_fr));
peak_freqs = 10^(order-2).*round(peak_freqs./10^(order-2));
peaks{i} = unique(peak_freqs)/10^6;
```

Next Week

- **Continue working with Zedboard**
- **Continue adding transmitters and receivers on the Grid (introduce classes/OOP to OEDL scripts)**
- **Examine and experiment with wiserd modules**
- **Start implementing MATLAB code in C/C++ and examine performance increase (long-term goal)**