RA8 FFT PoC Test Signal Generation

Requirements:

To use this script, you must have Python 3 installed, as well as the following packages:

1. Numpy
2. Matplotlib
3. Scipy

Summary:

The "signal_generator.py" script is used for the following:

1. Generating signal data for use in the FFT PoC.
2. Generating a representation of the simulated signal data that can be easily copied into the PoC code.
3. Generating low-pass filter coefficients.
4. Running a simulation of the FFT and filtering.
5. Reading in data that is generated from the PoC.
6. Plotting both the simulated data and the real data from the PoC.

By default, the script generates 1024 sine wave samples at a sample rate of 20 kHz. The parameters for this generation are defined at the top of the script as "num_samples" and "sample_rate". The output wave is a combination of two waves, a 1 kHz component, and a 5 kHz component.

The low-pass filter coefficients are created by the "create_lpf_coefficients" function. This function takes two inputs: the number of filter taps, and the cutoff ratio.

The sine wave samples and the filter coefficients can be easily printed and copied into the PoC with the "print_c_array" function. You can uncomment the usage of this function at the end of the script to output them to the terminal.

The simulation and real data from the PoC are plotted using the "matplotlib" library. Two figures are being created. Each figure contains the same four plots. Figure 1 displays the simulated data and figure 2 displays the real data. The four plots are the input signal plot, the FFT of the input signal, the filtered signal, and the FFT of the filtered signal.

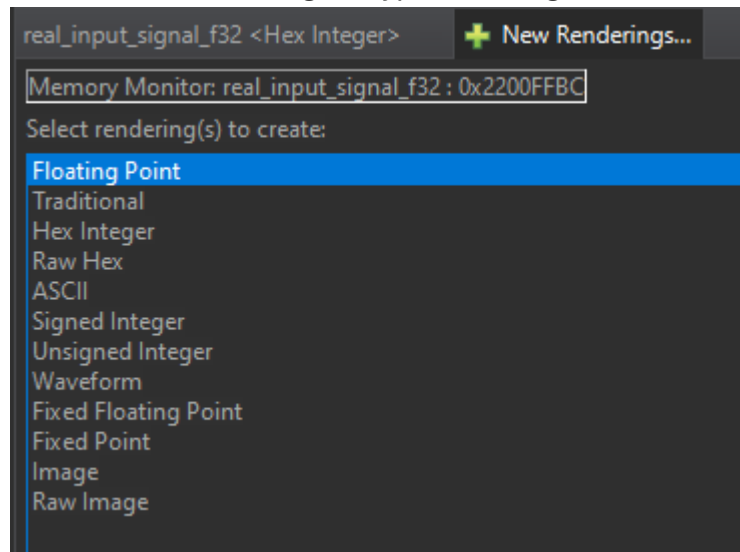**Usage Instructions:**

Gathering the input data:

1. Build and flash the PoC project in e2 studio. Set a breakpoint at the end of the main loop.

```
154
155            fft_cycle = ts_fft > ts_0? ts_fft - ts_0 : ~(ts_0 - ts_fft);
156            filter_cycle = ts_filter > ts_fft? ts_filter - ts_fft : ~(ts_fft - ts_filter);
157            fft2_cycle = ts_fft2 > ts_filter? ts_fft2 - ts_filter : ~(ts_filter - ts_fft2);
158            total_cycle = ts_fft2 > ts_0? ts_fft2 - ts_0 : ~(ts_0 - ts_fft2);
159
160            APP_PRINT("FFT input signal clock cycle: %d\n", fft_cycle);
161            APP_PRINT("Filter input signal clock cycle: %d\n", filter_cycle);
162            APP_PRINT("FFT filtered signal clock cycle: %d\n", fft2_cycle);
163            APP_PRINT("Total clock cycle: %d\n", total_cycle);
164
```

2. Open the "Memory" tab and add monitors for the buffers.
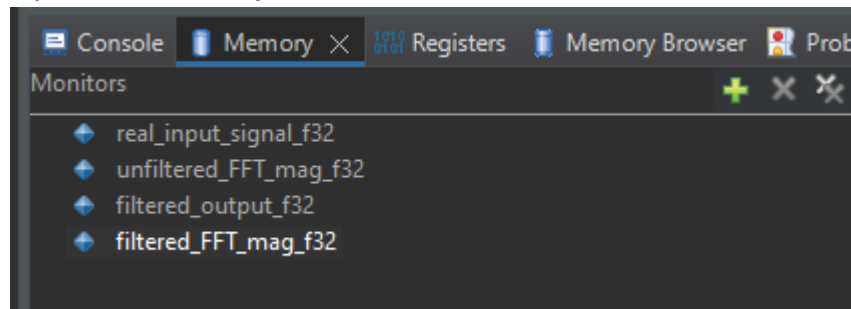
3. Add a "New Rendering" of type "Floating Point"

4. Select and copy the data that you wish to plot. In this case, we are copying 1024 samples.

5. Paste the data into a text editor such as Notepad++.

6. Select and delete the first column of memory addresses using "Shift + Alt" to select.

1	0x000000002200FFBC	-8.054578E-1	-8.054578E-1	-8.127106E-1
2	0x000000002200FFFC	-9.255311E-1	-9.319780E-1	-9.400366E-1
3	0x000000002201003C	-1.021429E0	-1.026264E0	-1.031099E0
4	0x000000002201007C	-1.089121E0	-1.092344E0	-1.096374E0
5	0x00000000220100BC	-1.134249E0	-1.135861E0	-1.139084E0
6	0x00000000220100FC	-1.164872E0	-1.165678E0	-1.168095E0
7	0x000000002201013C	-1.182601E0	-1.183407E0	-1.185018E0
8	0x000000002201017C	-1.198718E0	-1.199524E0	-1.200330E0
9	0x00000000220101BC	-1.209194E0	-1.208388E0	-1.208388E0
10	0x00000000220101FC	-1.212418E0	-1.211612E0	-1.212418E0
11	0x000000002201023C	-1.215641E0	-1.215641E0	-1.215641E0
12	0x000000002201027C	-1.218059E0	-1.218059E0	-1.218059E0
13	0x00000000220102BC	-1.218864E0	-1.218864E0	-1.218864E0
14	0x00000000220102FC	-1.219670E0	-1.220476E0	-1.219670E0
15	0x000000002201033C	-1.220476E0	-1.220476E0	-1.220476E0
16	0x000000002201037C	-1.221282E0	-1.221282E0	-1.221282E0
17	0x00000000220103BC	-1.222088E0	-1.221282E0	-1.222088E0
18	0x00000000220103FC	-1.222088E0	-1.221282E0	-1.222894E0
19	0x000000002201043C	-1.222894E0	-1.222088E0	-1.222894E0
20	0x000000002201047C	-1.222894E0	-1.222894E0	-1.222894E0
21	0x00000000220104BC	-1.222894E0	-1.223700E0	-1.222894E0
22	0x00000000220104FC	-1.223700E0	-1.224505E0	-1.224505E0
23	0x000000002201053C	-1.226117E0	-1.226117E0	-1.226923E0
24	0x000000002201057C	-1.230146E0	-1.230146E0	-1.230146E0

7. Repeat steps 3 through 6 for each of the memory buffers.
8. Save the files with whatever names you find appropriate. You will need to input these file names in the first section of the script.

```python
def main():
    ################################################################
    # Define input file paths here

    real_signal_path ='real_signal_out.txt'
    real_signal_data = read_data(real_signal_path)


    fft_input_path = 'unfilt_fft_mag_out.txt'
    fft_input_data = read_data(fft_input_path)


    filter_input_path = 'filter_out.txt'
    filter_input_data = read_data(filter_input_path)


    fft_filter_input_path = 'filt_fft_mag_out.txt'
    fft_filter_input_data = read_data(fft_filter_input_path)


    ################################################################
```

9. Now you can run the script with "python.exe signal_generator.py".