MÜHENDİSLİK FAKÜLTESİ
FACULTY OF ENGINEERING
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ BÖLÜMÜ
DEPARTMENT OF ELECTRICAL AND
ELECTRONICS ENGINEERING

ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

DUMLUPINAR BULVARI 06800
ÇANKAYA ANKARA/TURKEY
T: +90 312 210 23 02
F: +90 312 210 23 04
ee@metu.edu.tr
www.eee.metu.edu.tr

EXPERIMENT 2. PROGRAMMING BOTH MYRIO CPU AND FPGA

LABORATORY REPORT

Student 1: Abdullah Canbolat, 2304244
Student 2: Güray Özgür, 2167054
Student 3: Yunus Bilge Kurt, 2232395

In the report, CPU corresponds to floating-point implementation, and FPGA corresponds to fixed-point implementation.

i) Place the screenshot of the final version of **block diagrams** and **front panels** in the space below. Please note that the details should be **clearly visible**. You can use ***Snipping Tool***.

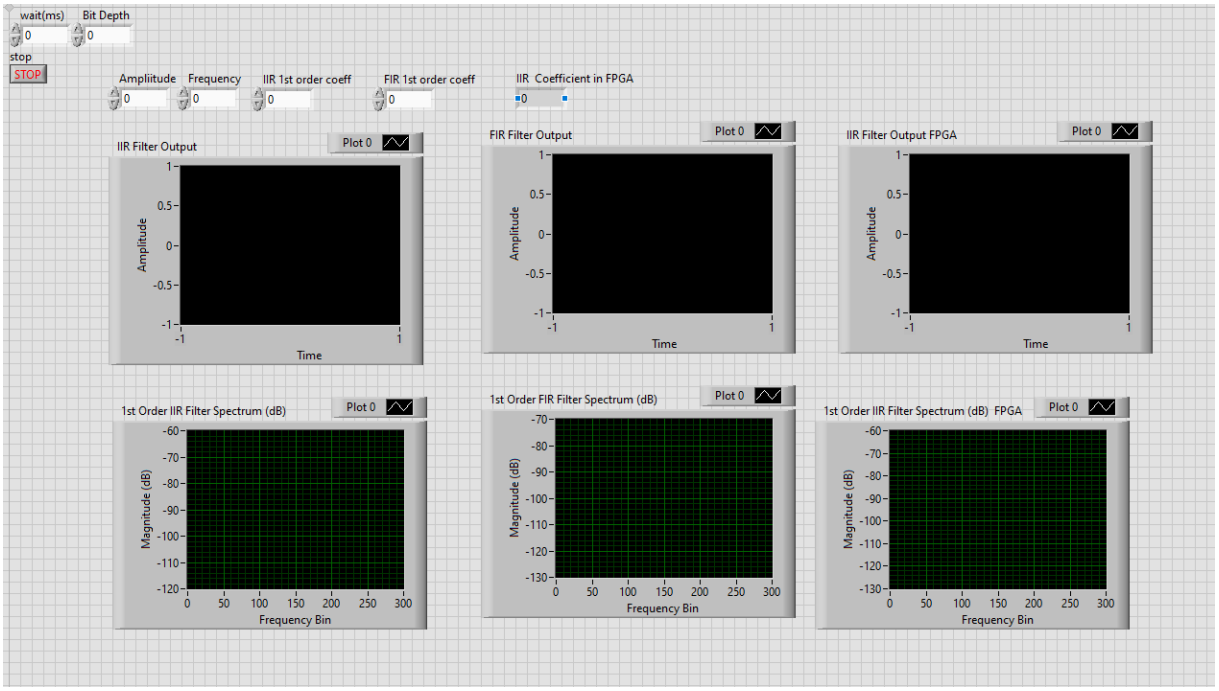

Figure 1: Front panel of the setting for Experiment 2

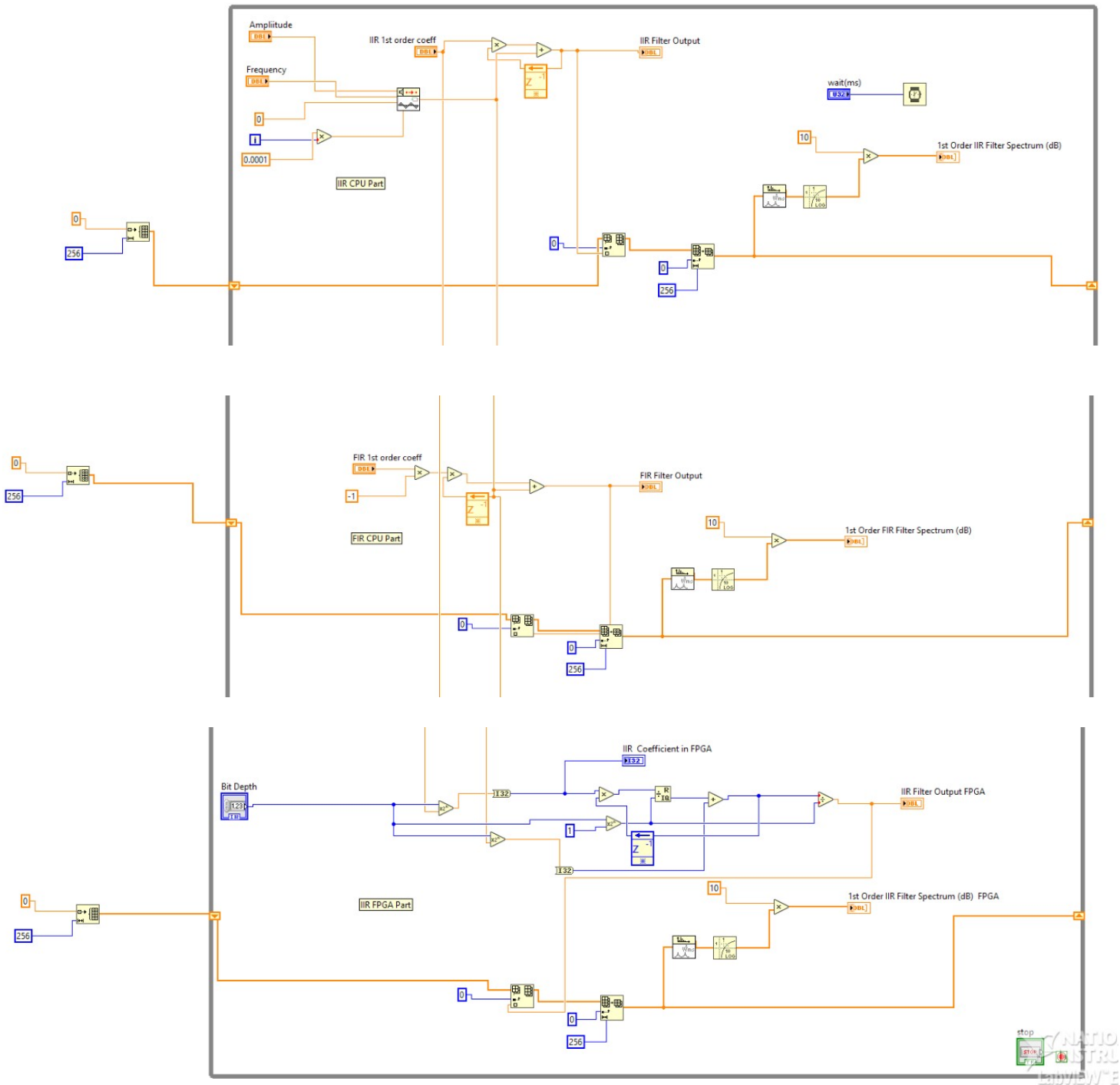

Figure 2: Block diagram of the setting for Experiment 2

ii) Set ***amplitude*** and ***frequency*** of Triangular waveform as **0,001** and ***30***, respectively. Set ***wait(ms)*** as ***10***.

iii) Choose the ***IIR and FIR filter coefficients*** as ***0.5*** and ***Bit Depth*** as ***14***. ***Note*** and ***plot the filtered outputs*** for both ***CPU*** and ***FPGA*** implementations. Comment on the results.

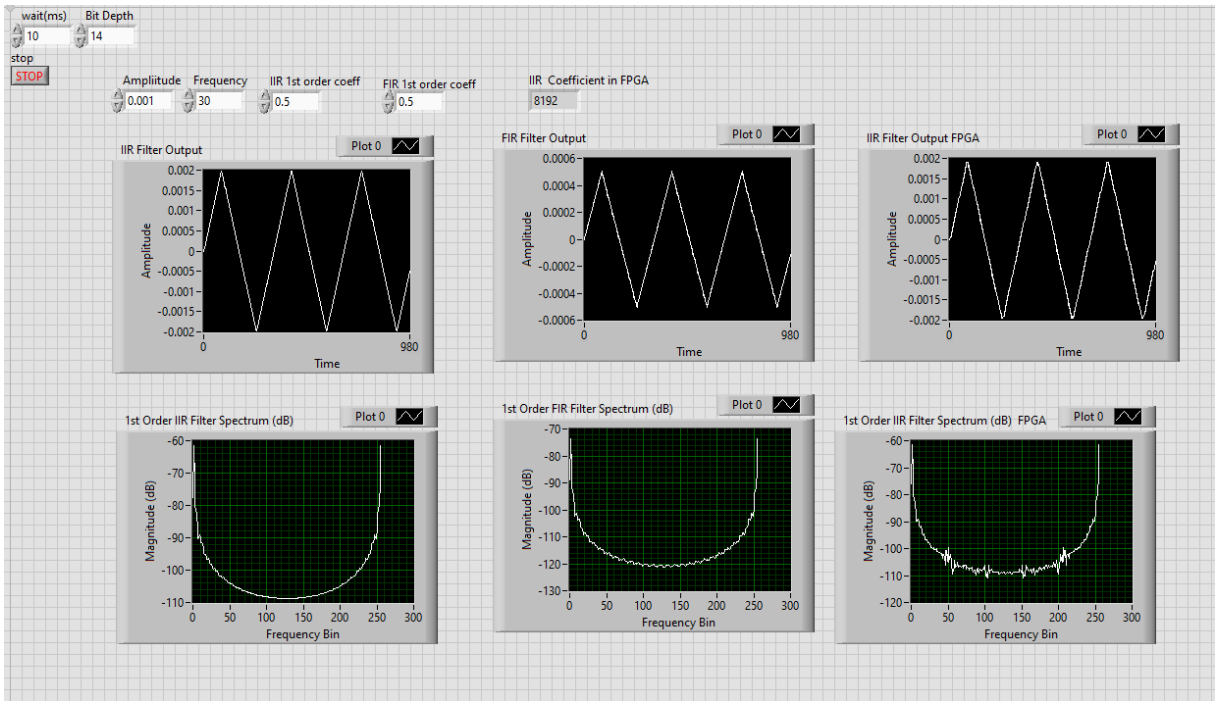

Figure 3: Front panel when IIR and FIR coefficients are 0.5 and bit depth is 14

In this setup, FIR filter is in CPU and IIR is in both FPGA and CPU. CPU uses floating-point representation whereas FPGA uses fixed-point representation. Due to fixed-point representation, errors occur from finite register length, limit cycle and coefficient quantization.

In FPGA spectrum of IIR, we observe some distortions due to fixed-point representation. However, since bit depth is 14, these distortions are small and does not prevent us to see output clearly, which can be seen from Figure 3.

iv) Change ***filter coefficients*** as ***0.99***. ***Note the changes*** in ***myRIO CPU and FPGA*** implementations. ***Plot*** the results. ***Comment*** on the results. What happens to ***FIR filter output***? Does it ***change significantly***?

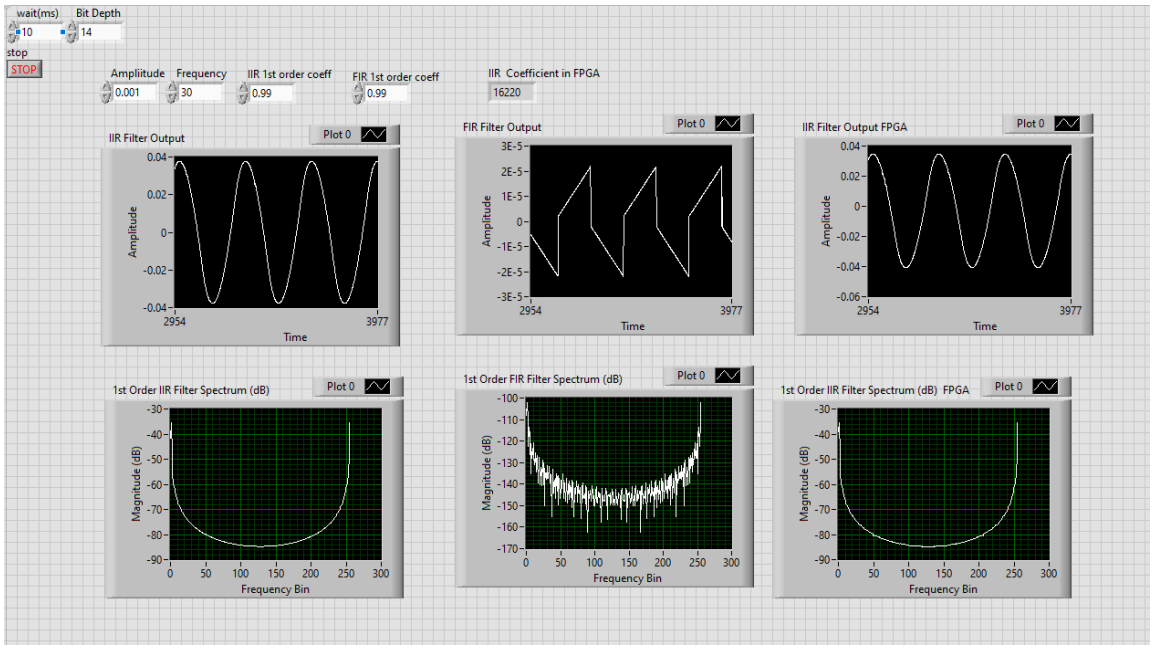

Figure 4: Front panel when IIR and FIR coefficients are 0.99 and bit depth is 14

We have a zero at 0.99 for FIR filter and a pole at 0.99 for IIR filter.
Since the coefficients, zero and pole of FIR and IIR filter respectively, are very close to zero frequency, i.e. 1, FIR filter shows high-pass characteristics, whereas IIR filter shows low-pass characteristics because of its pole at 0.99. For FIR filter, it is observed that it has changed significantly. This can be seen as FIR output has sharp transitions, whereas output of IIR filter is smooth. The reason why we see a sin-like function at the output of IIR filter, high frequency components of triangular wave is eliminated by the filter, which can be seen from Figure 4.

v) ***Decrease Bit Depth*** one by one from ***14*** to ***10***. ***Note the changes*** in the ***FPGA*** output. ***Plot*** the waveforms for bit depth 10. ***Explain*** them.

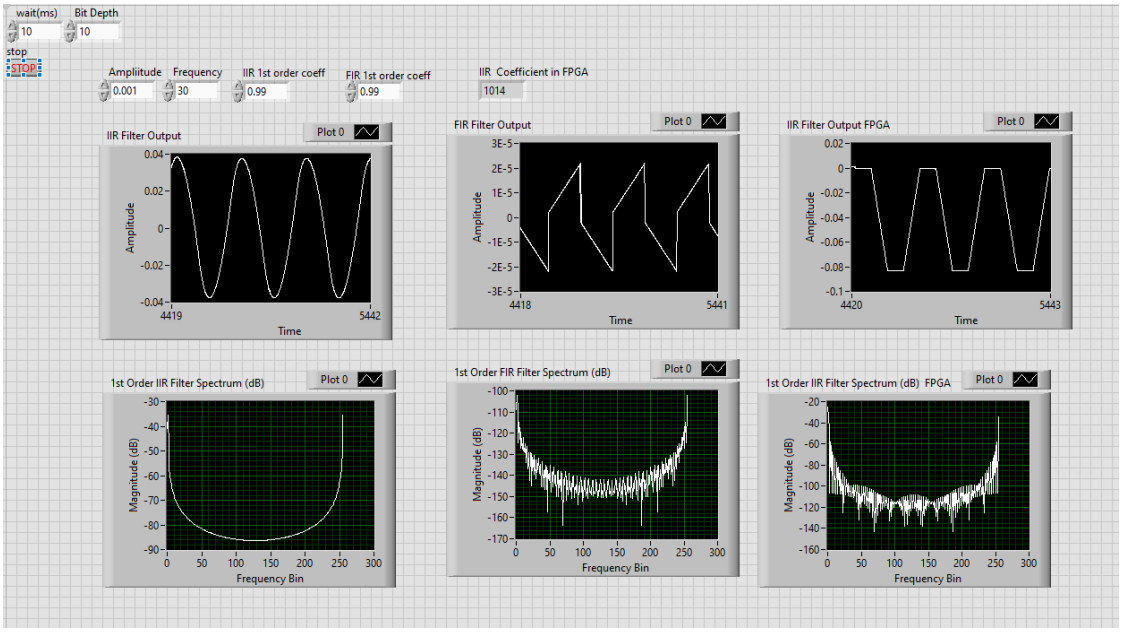

Figure 5: Front panel when IIR and FIR coefficients are 0.5 and bit depth is 10

Since we have decreased the bit depth from 14 to 10, we have observed the error from fixed-point representation from coefficient quantization and input quantization. Coefficient quantization distorts the magnitude and phase characteristics. Since bit depth is not enough to realize the filter due to high errors from quantizations, we observed a saturated output signal as we decrease the bit depth from 14 to 10 gradually, which can be seen from Figure 5.

vi) Now, ***increase Bit Depth*** from ***14*** to ***18*** one by one and ***note the changes***. ***Why*** do we observe such a response for the ***FPGA implementation***, which is ***different than the CPU implementation*** in myRIO? ***Plot*** the waveforms for bit depth 17 and 18.

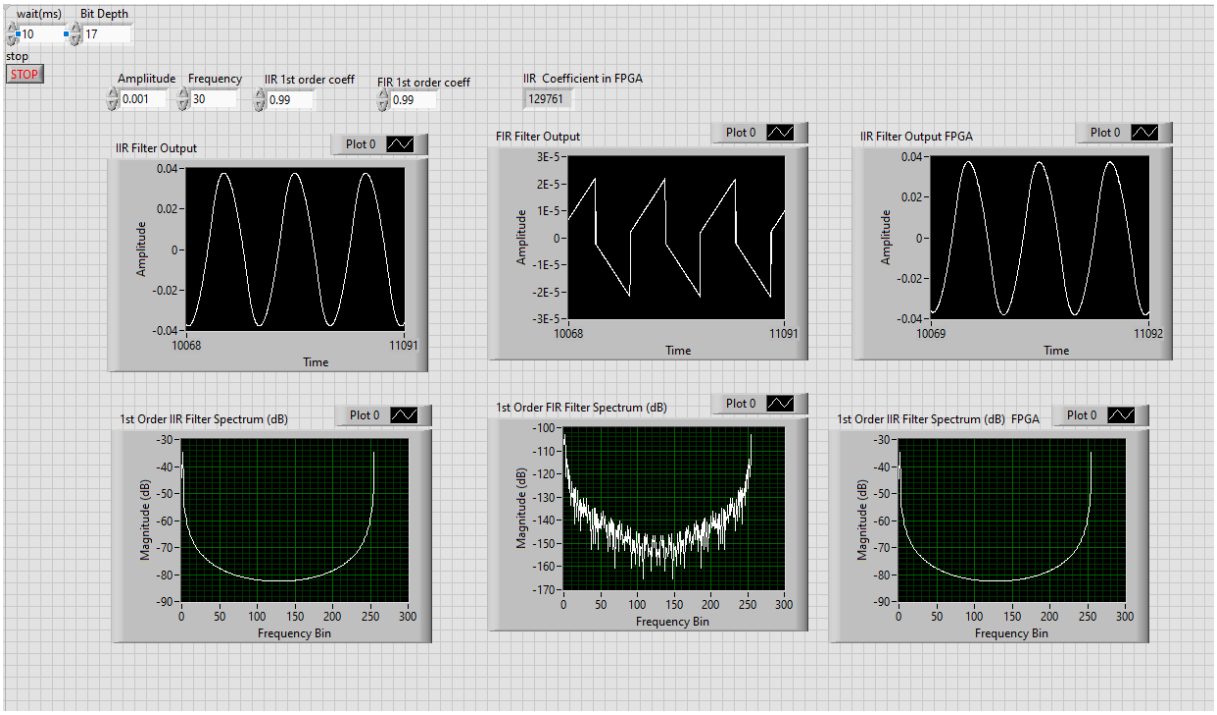

Figure 6: Front panel when IIR and FIR coefficients are 0.5 and bit depth is 17

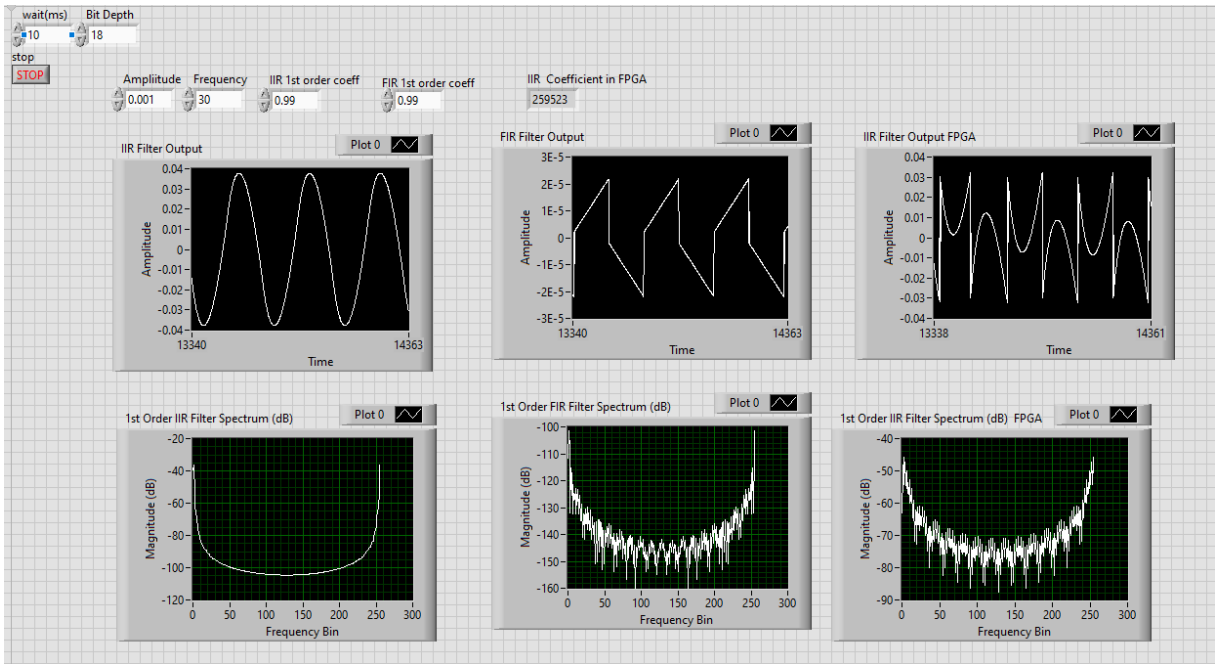

Figure 7: Front panel when IIR and FIR coefficients are 0.5 and bit depth is 18

Errors due to finite register length is observed when we increase the bit depth to 18, since we use signed representation, at most 17 bits can be stored in our 32-bit register. When we increase the bit depth further, our register length is not enough. The output signal deviates from our expectations. One can see that the bit depth is 18, the waveform is interrupted, and has sharp peaks rather than continuing to go on like a sin-like wave, which can be seen from Figures 6 and 7.

The defintion of **Finite register length** is given with its changes in bit depths according to our observartions, which was taken from Experiment 2 manual.

Finite register length: Arithmetic operations are performed using registers with finite length. When two numbers with 18-bit representation are multiplied, the resulting number is a 34-bit number, since it is a signed representation. If the length of the register is 32 bits, then the numerical precision is lost due to quantization of a 34-bit number to 32-bit representation. Therefore, the filter output may deviate significantly from the perfect response. In addition, filter frequency response is also distorted.

vii) Set the bit depth to 14. Change the ***IIR filter coefficient*** to ***1.2***. ***Why*** do we observe such a response for both of the implementations? ***Plot*** the waveforms and ***comment***.
viii) Change the ***FIR filter coefficient*** to ***1.2***. ***Why don't we see a similar change for the FIR filter***?

Now, with the changes in filter coefficients for FIR and IIR to 1.2, we have changed the zero of FIR to 1.2 and the pole of IIR to 1.2. Since the pole of IIR is out of the unit circle, unstable behavior is observed. We do not observe a unstable behavior for FIR, since unit circle is in ROC. Although IIR filter is unstable, we do not observe divergence to infinity due to finite register length in FPGA, but it oscillates wildly, which can be seen from Figure 8.

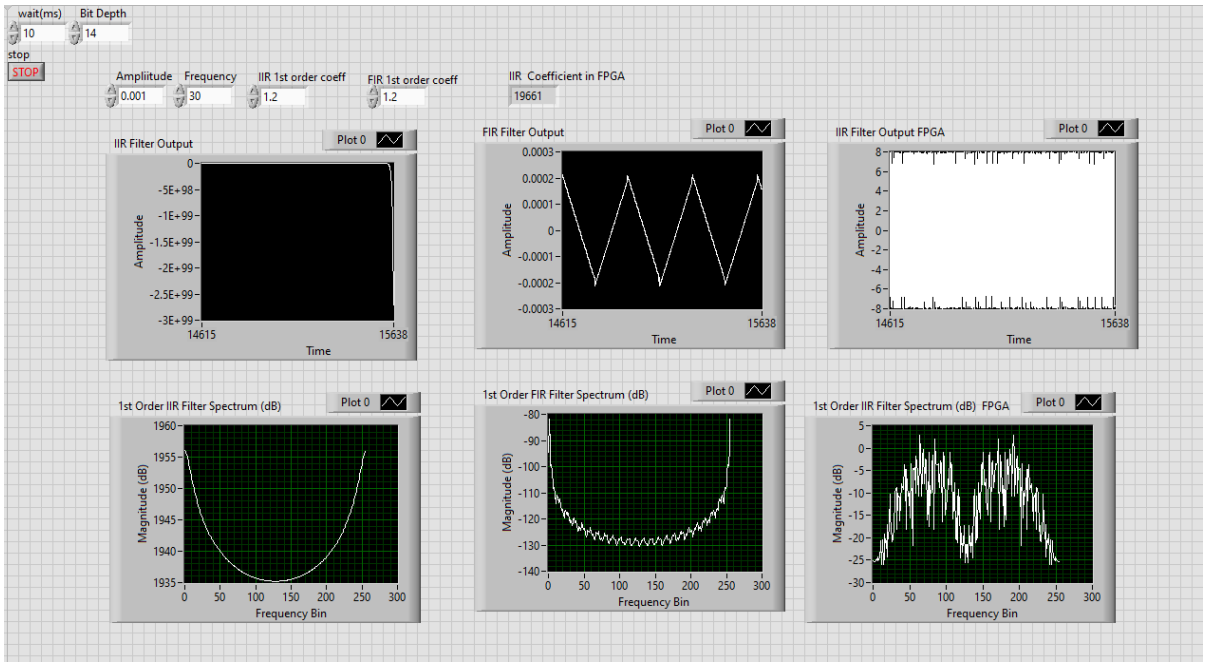

Figure 8: Front panel when IIR and FIR coefficients are 1.2 and bit depth is 14

ix) Choose your second-order FIR and IIR Filter coefficients as 0.5. What do you observe? ***Plot*** the waveforms. ***Compare*** your results with the first-order filter.

When we arrange the coefficients to 0.5, 0.5 for both FIR and IIR, we observe zeros for FIR and poles for IIR at 1, and -0.5. We expect that low-pass characteristics for IIR filter and high-pass characteristics for FIR filter, which can be observed from the front panel in Figure 9. The reason is that IIR filter has a pole at 1 and FIR filter has a zero at 1. For the comparison with the first order, in FIR filter, we see a high-pass characteristics for second order, while seeing low-pass characteristics with the same coefficient for first order. For IIR, we have reached the boundry for stability, if the coefficients were a bit different, we could have experienced unstability.

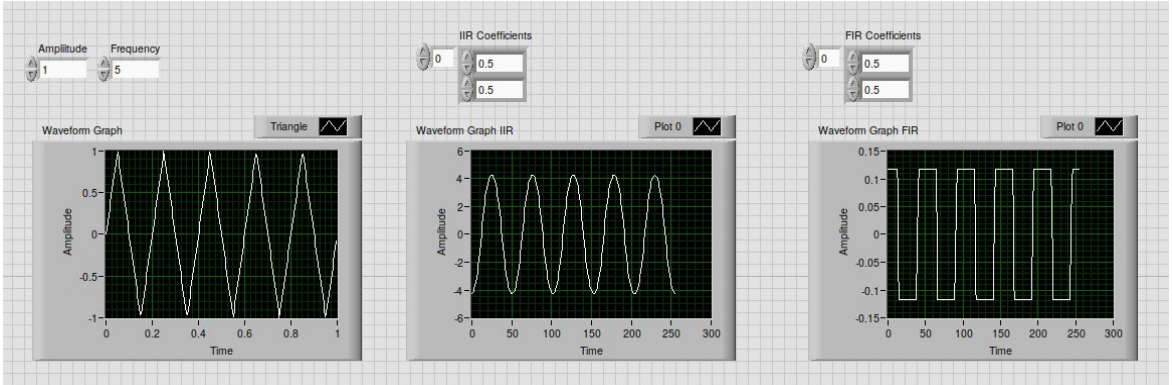

Figure 9: Front panel of the setting for second order FIR and IIR filters

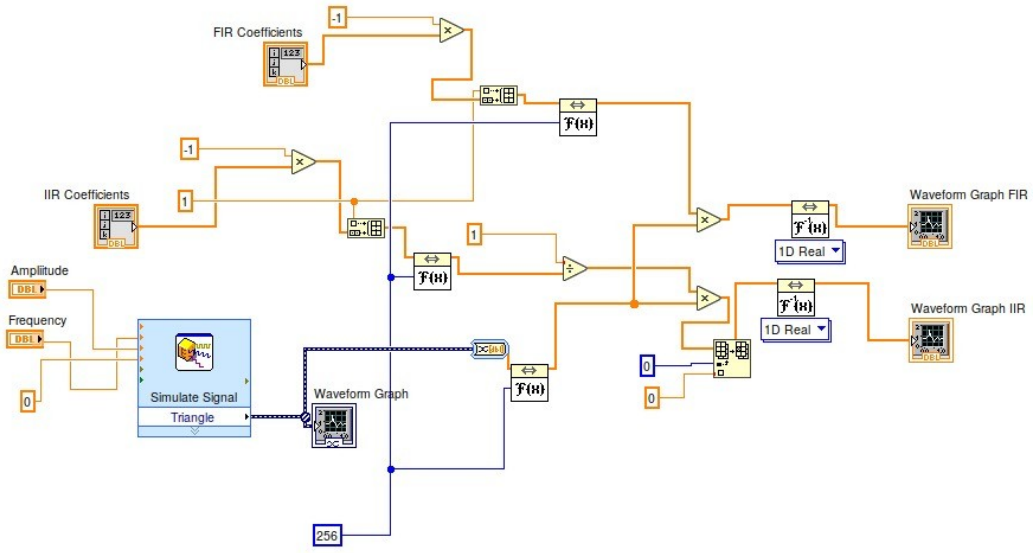

Figure 10: Block diagram of the setting for second order FIR and IIR filters

x) Implement first-order IIR filtering with floating-point and fixed-point in ***MATLAB***. Choose your IIR filter coefficient as 0.5. For input, consider a random signal (rand command) with a length of 256. Choose your bit depth as 8. ***Plot*** the resulting outputs in the ***same figure***. Also, plot the ***error*** between floating-point and fixed-point results. ***Comment*** on them. ***Attach*** your MATLAB codes.

The mean of the error is 0.012 and a variance of 0.0001. We see the plots colliding with each other, which indicates for this implementation, fixed-point representation with a bit depth 8 is good enough, the plots can be seen from Figures 11 and 12.

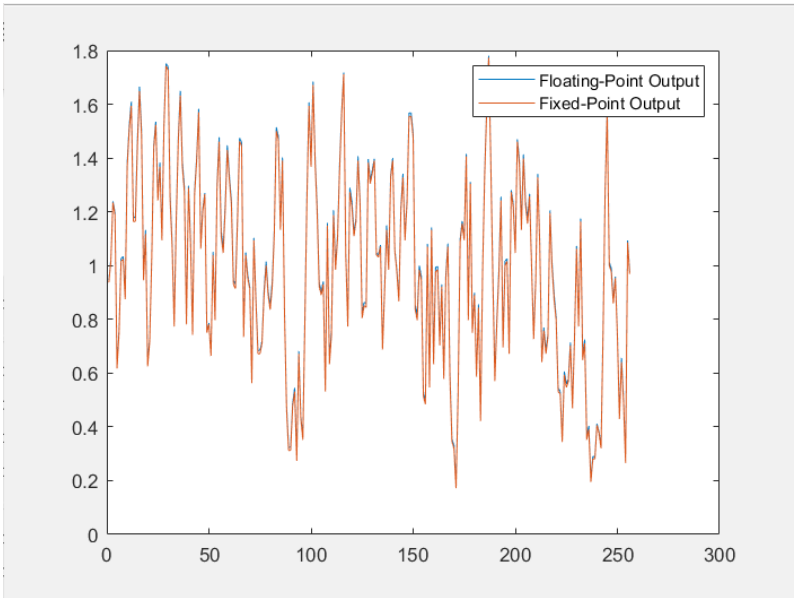

Figure 11: Floating-point and fixed-point implementations of first order IIR filter

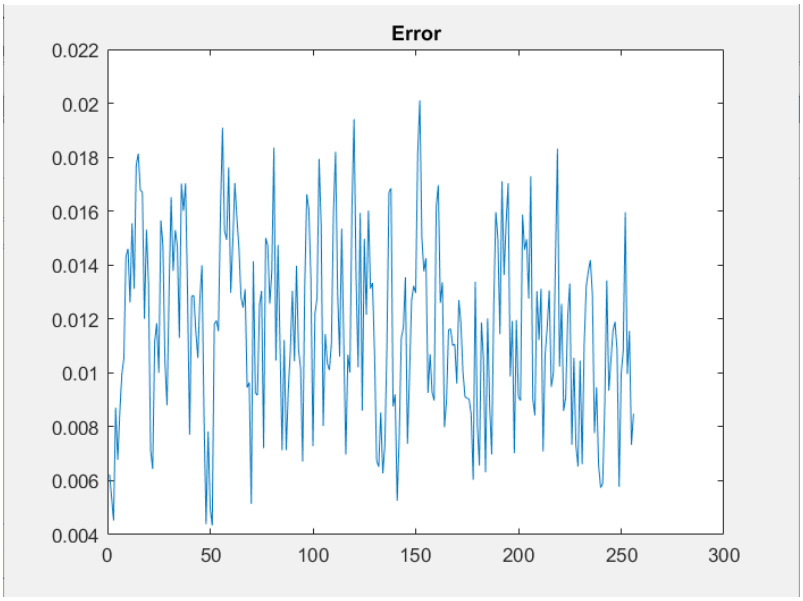

Figure 12: Difference of floating-point and fixed-point implementations of first order IIR filter

```
signal = rand(256,1);

%% IIR Filter Floating Point
y1 = zeros(size(signal));
x = signal;
for k = 1:256
    if k == 1
        y1(k) = x(k);
    else
        y1(k) = 0.5*y1(k-1)+x(k);
    end

end

%% IIR Filter Fixed Point
x2 = floor((2^7)*signal);
y2 = zeros(size(y1));
for k = 1:256
    if k == 1
        y2(k) = x2(k);
    else
        y2(k) =floor( floor(0.5*(2^7))*y2(k-1)*2^(-7))+x2(k);
    end

end

figure
plot(y1)
hold on
plot(y2./(2^7))
legend("Floating-Point Output","Fixed-Point Output")
figure
plot(y1-y2./(2^7))
title("Error")
```