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 4. DECIMATION, INTERPOLATION AND PHASE-LOCKED LOOP

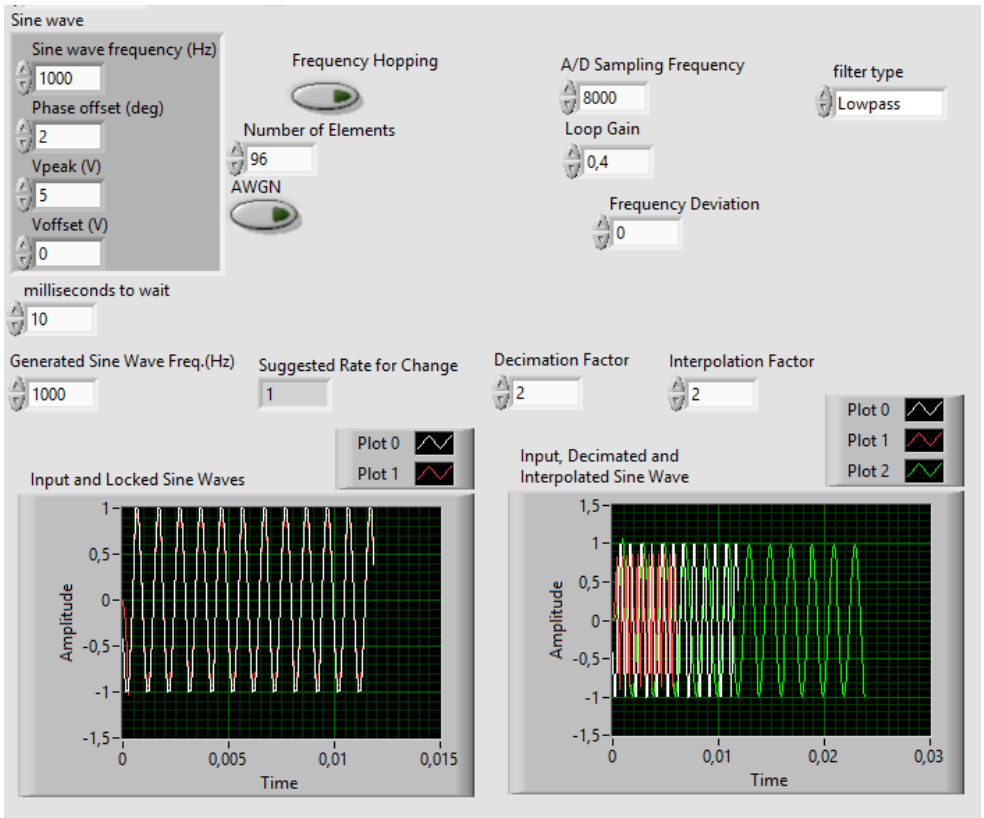
PART 2

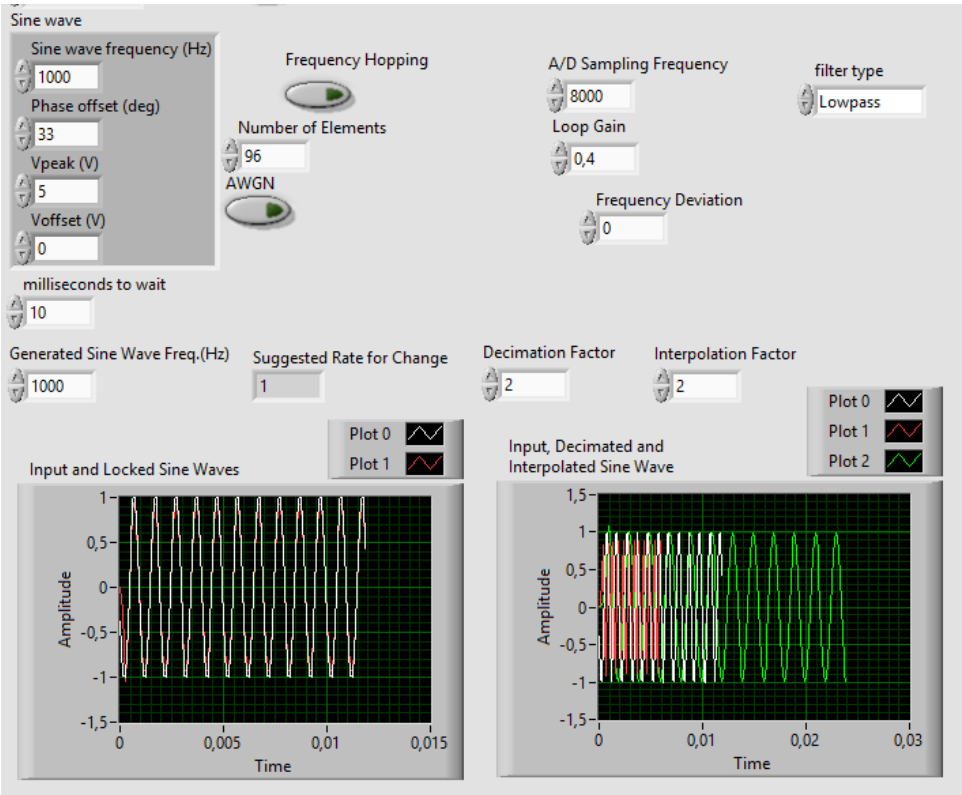
LABORATORY REPORT

Student 1:Süleyman Emre CAN - 2093524
Student 2:Yekta Demirci - 2093607

Task

1) Adjust your program parameters as shown in Fig. 8 and run your program. Observe if the PLL locks onto the input frequency/phase or not. Change the **"Phase offset"** on the upper left part of the front panel to see if the PLL correctly follows the input signal. **Place a screenshot** of the input and locked sine waves.





2) Gradually **increase** the ***input frequency*** **from 1000Hz in 1Hz steps**. Determine the frequency where the PLL is **not locked any longer**. Note the maximum offset frequency which determines the lock-in range. Report this value. You can play with the **loop gain** in order to have the PLL lock. Also you can change the **number of elements read from the FIFO** in order to stabilize the waveform to see if the PLL is in lock.

At 1014 Hz, PLL is not locked any longer.

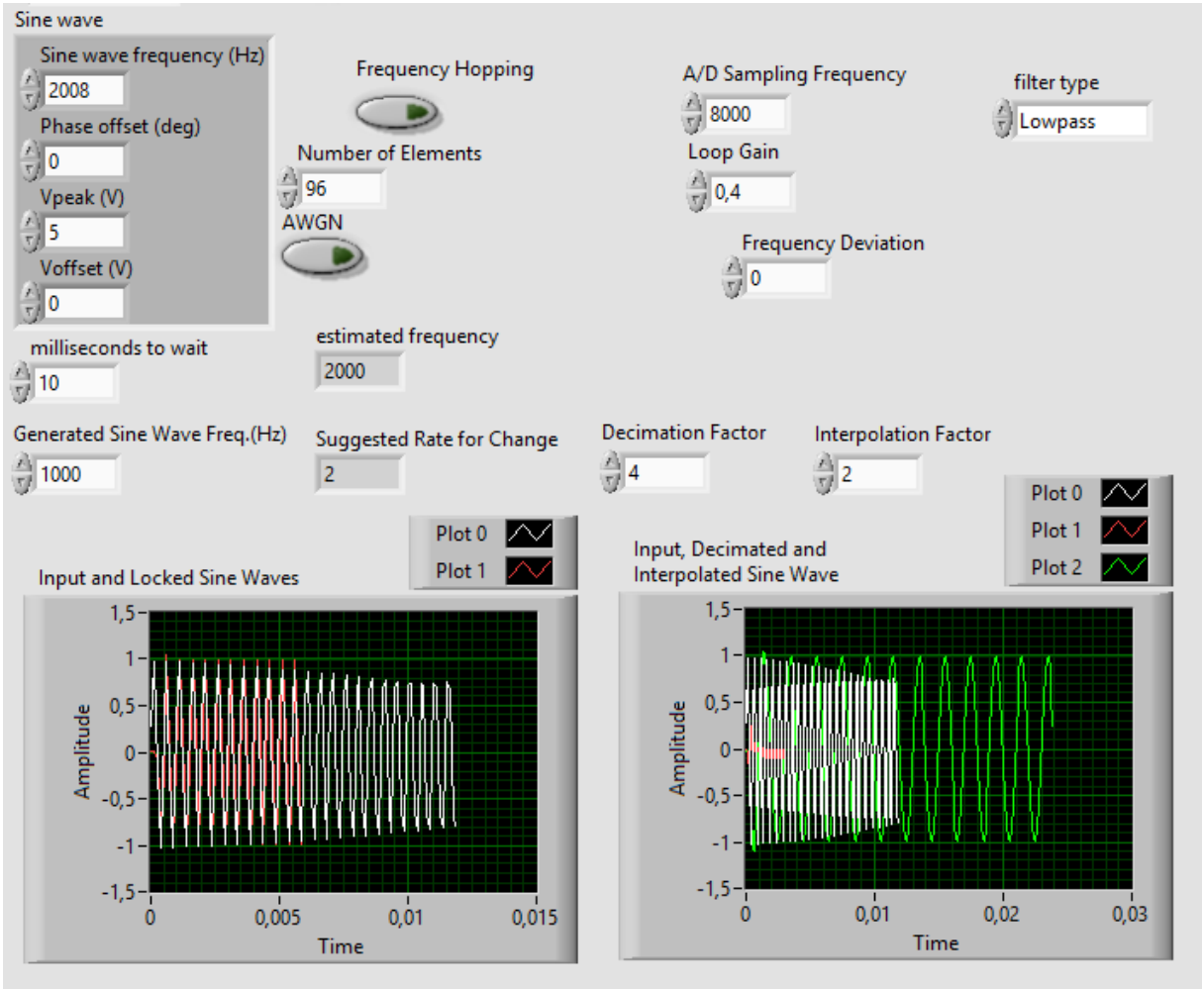
3) Now you need to observe the effect of the loop gain. Choose the **input signal frequency** as **1kHz**. Starting **from 0.05 increase the loop gain to 2 in 0.2 steps**. Determine the value of loop gain where the PLL does not lock on the input signal. **Comment** on the effect of small and large loop gain.

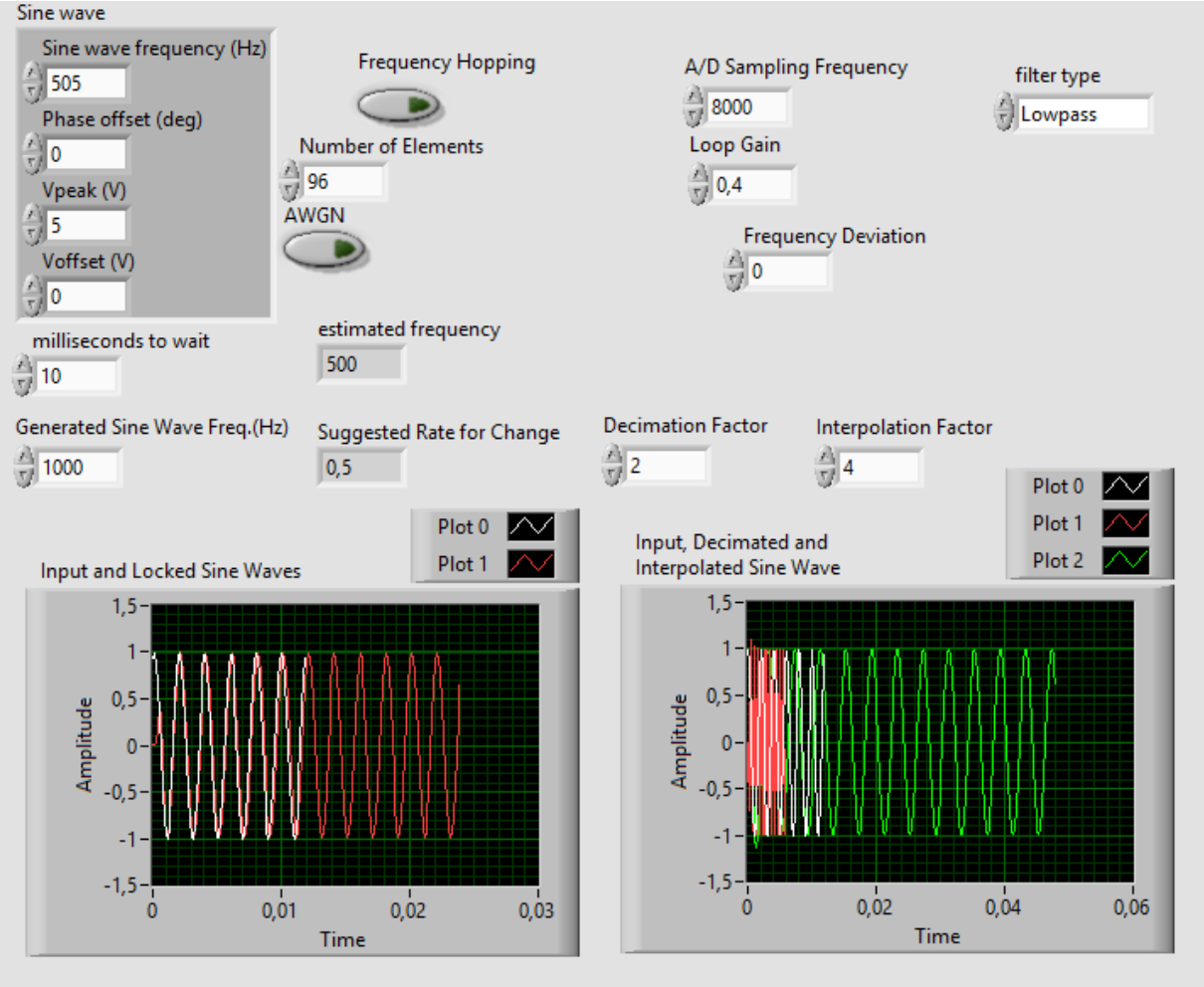
After 0.65, PLL started to not locking on the input.

If the loop gain is low, it locks slowly and it may take a little time to lock. However, it locks at the end. On the other hand, if high loop gain is used, it tries to lock faster however due to high gain it cannot converge and keeps swinging. As a result, it does not lock.

4) Return to the **original settings in Fig. 8**. Now change the **input frequency** to **2008 Hz**. Also adjust the Decimation and Interpolation factors. What happens to the PLL output? What should be the value for the **decimation and interpolation factors** to observe PLL lock? **Repeat the same steps for 505Hz.**

Take a **screenshot** for both cases.





When the interpolation and decimation factors are arranged, it locks the signal.

5) Return to the **original settings in Fig. 8**. **Press the AWGN button** to add white noise to the input signal. Observe the **"Estimated Signal Frequency" while you increase the noise standard deviation.** At which value, estimated frequency starts to change? What can you say about the detection method optimality?

When the standard deviation is around and more than 3.800, it cannot estimate the carrier frequency. It is quite well. Since it is based on finding the index in FFT domain where the highest value occurs, it can estimate carrier frequency very optimally. If it was based on time domain measurements, it could not show such well performance.

6) Return to the **original settings in Fig. 8**. Choose the **frequency deviation as 1Hz**. Apply **frequency hopping mode**. Observe if the PLL locks onto the frequency hopping signal. Now start to **increase the frequency deviation in 1Hz steps**. Determine the value for frequency deviation where the PLL **does not lock onto the signal any longer**. Report this value. **Comment on the differences** between this step and step 2.

Since the noise is put onto the carrier frequency, the maximum value in FFT is still around 1k. Therefore, estimated frequency is 1k as well. However, fc denotes when there is frequency hopping. Therefore, the generated carrier frequency in the PLL does not matched the real one, as a result it cannot locks to the received signal after some frequency deviation which is around 19. In step 2, a similar error happened because of a different reason. In part two due to resolution we could not estimate the exact carrier frequency.