Lab 1: Introduction to LabView

# \\icnas3.cc.ic.ac.uk\py715\Desktop\Comms\fahrenheit_to_celsius_formulas.pngEx1-Data Types in LabView

* Convert each letter of a string into ASCII code.
* Convert each integer of Fahrenheit temperature to Celsius.
* Average value of the temperature is calculated.

Figure 1 Fahrenheit -> Celsius

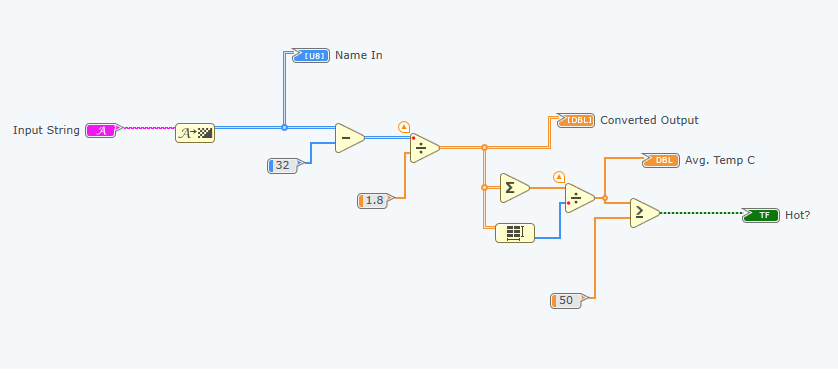


Figure 2 Diagram of the Converter

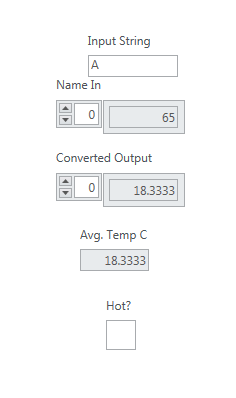
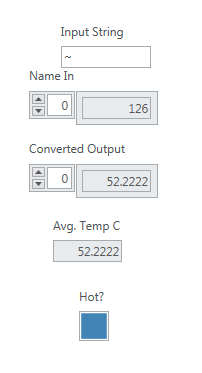
 

Figure 3 Input ~ Figure 4 Input A

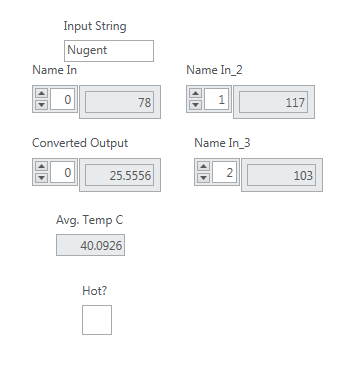
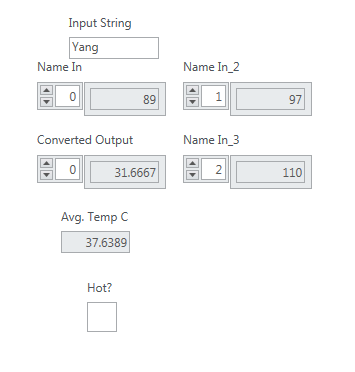


Figure 5 Input Nugent Figure 6 Input Yang

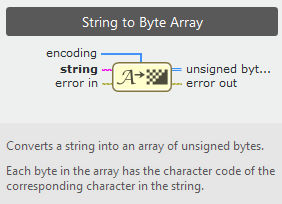


Figure 7 String to Byte Array

# Ex2-Implementation of the Central Limit Theorem (CLT)

CLT: When independent random variables are added, their sum tends toward a normal distribution. (Sufficiently large sample has finite level of variance, mean of the sample is approximately equal to the mean of the whole data.)

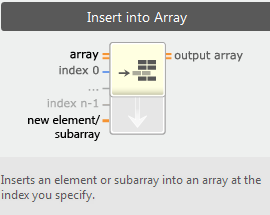


Figure 8 Insert into Array

* Tunnels: specify how the data will be transferred through the frame.

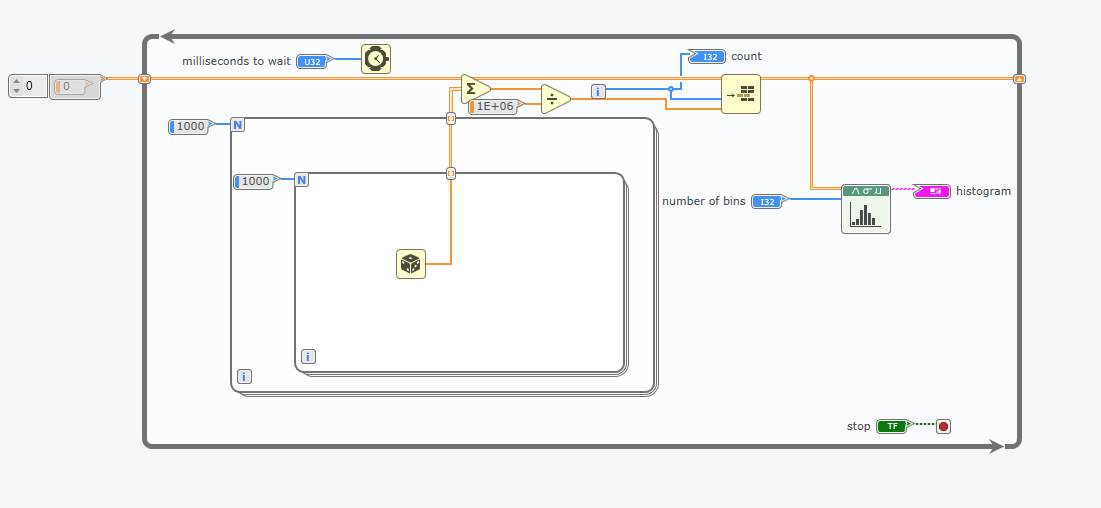


Figure 9 Diagram 1

* Mean-> use divide function to divide the total value of the randomly generated data by the size of the 2D array.
* Shift register is a type of tunnel that carries information from an iteration to the next one. So the tunnel on the left carrier the data from the last iteration to the next.

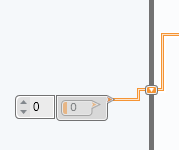


Figure 10 Initialise the array to 0

* Number of bins: the number of interval between the grid.

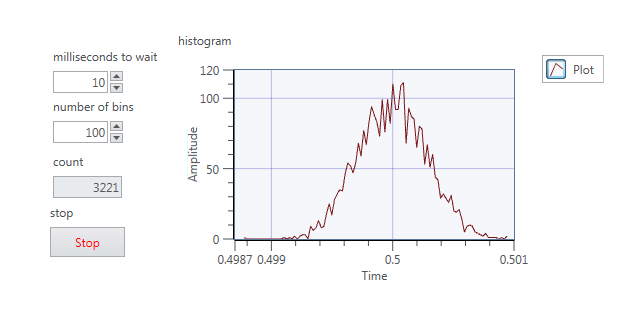


Figure 11 Output

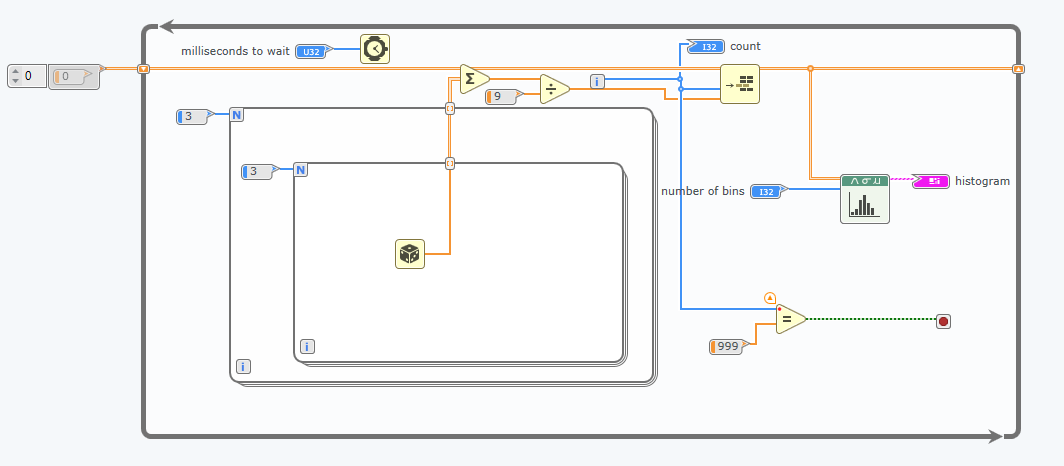


Figure 12 1k iteration

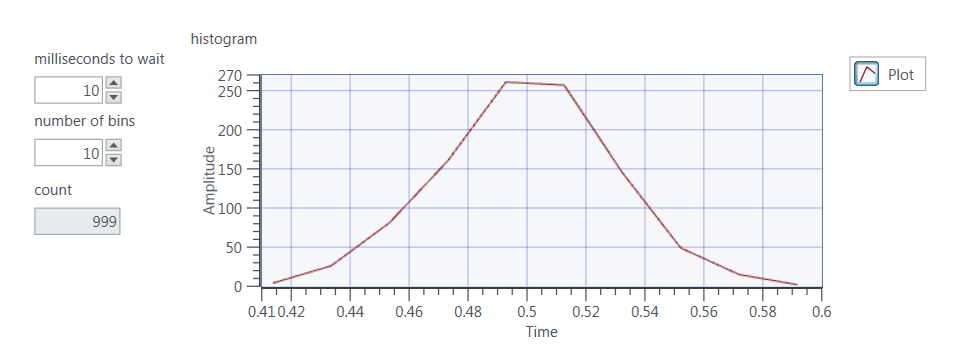


Figure 13 100 iteration

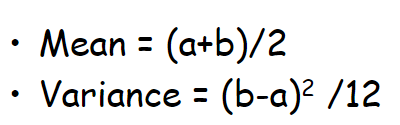


Figure 14 Formula

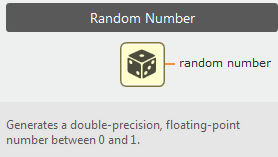
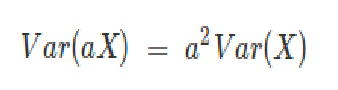


Figure 15 Random Number Generator

Before normalisation, the distribution has mean equal to 0.5, because the range of the number generated is from 0 to 1. Using the formula in Figure 14, we got mean = 0.5.

In order to have standard normal distribution (mean = 0 & variance = 1), we need to shift the distribution graph to left. So we minus 0.5 from the randomly generated sample -> mean now is 0; for variance, we have this formula.

The variance before normalise is 1/12, so if we want to obtain variance = 1, we need to multiply 12.\\icnas3.cc.ic.ac.uk\py715\Desktop\Comms\Var1over12.png \\icnas3.cc.ic.ac.uk\py715\Desktop\Comms\var1.png

Hence, we can multiply sqrt(12) in order to get Var(X) = 1. Outside the loop, we divided by 100, so for the same reason, we need to multiply by 10. Hence \\icnas3.cc.ic.ac.uk\py715\Desktop\Comms\sqrt12 times 10.png.

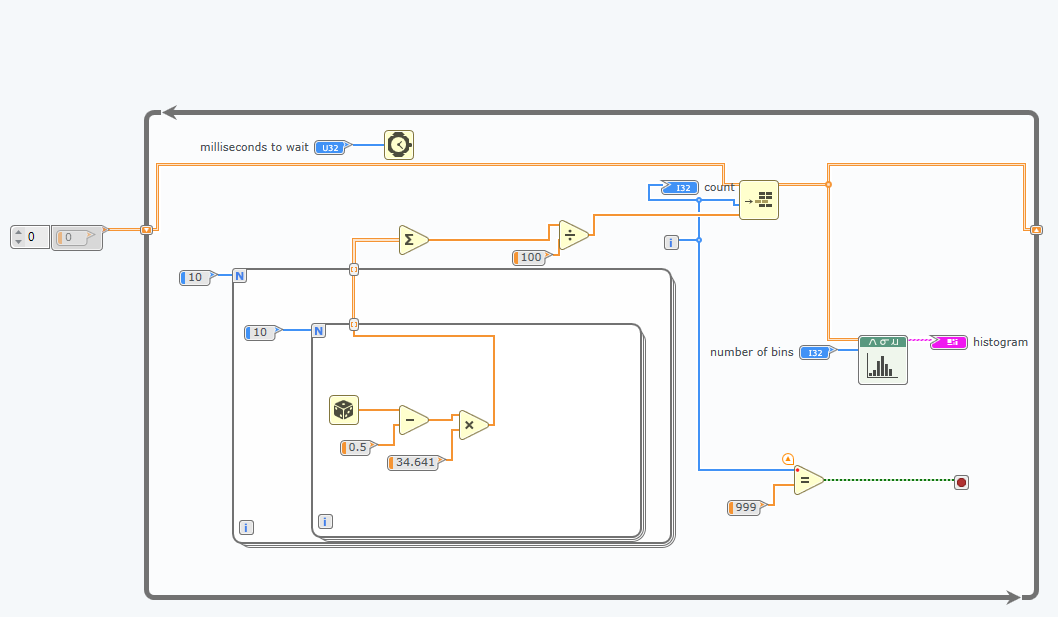


Figure 16 Unit Variance and 0 mean

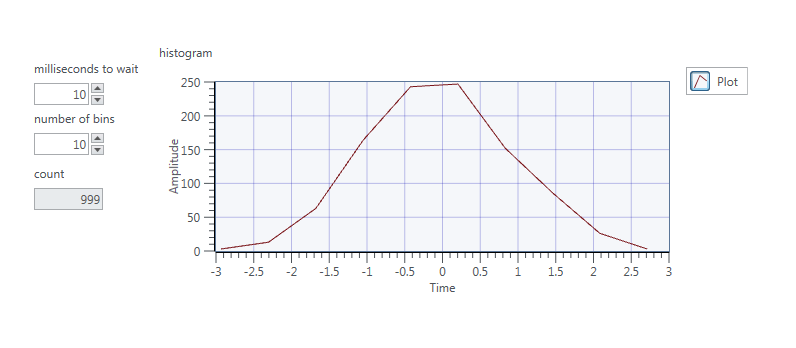


Figure 17 Unit Variance and 0 Mean Output

# Ex3-Waveforms

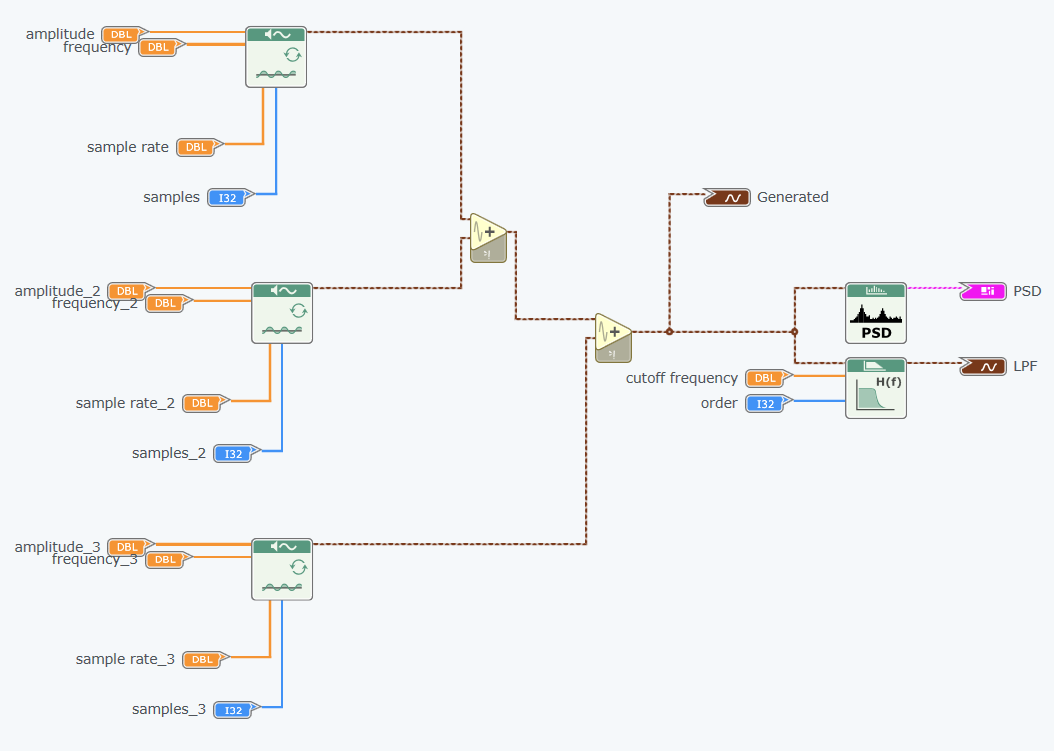


Figure 18 Initial diagram

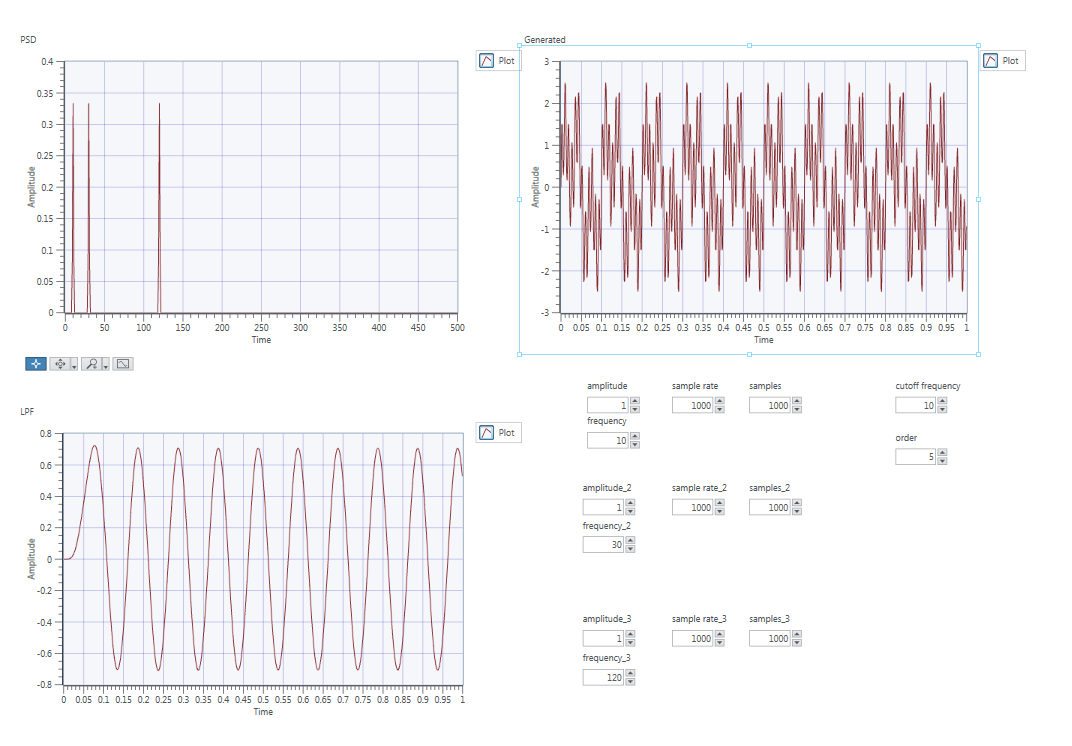


Figure 19 Initial panel

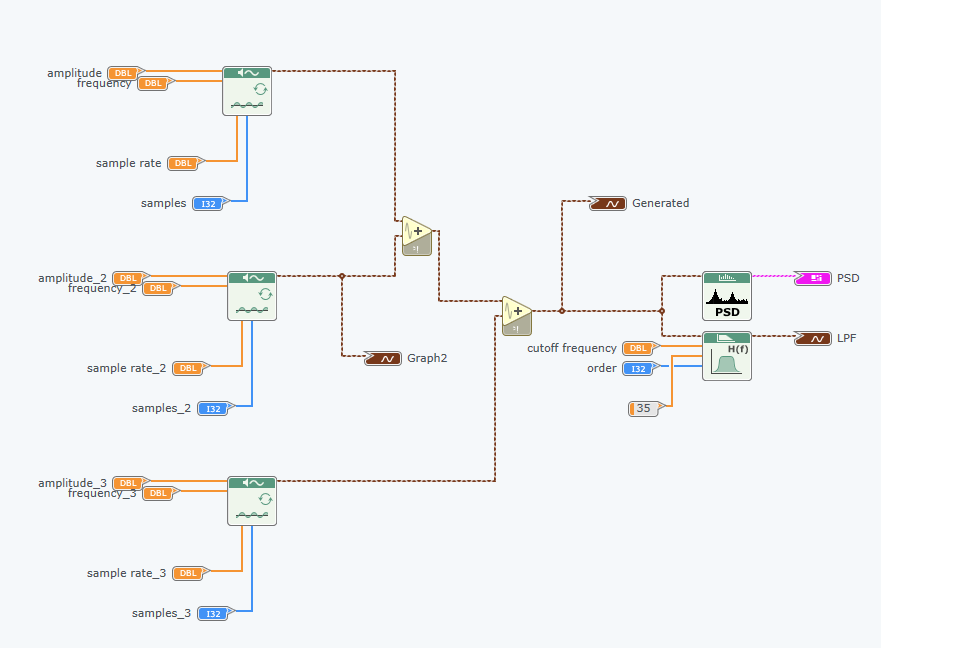
In order to get the sin wave with frequency 30, we adjust the cut-off frequency to around 30, we choose the range from 25(low cut-off) to 35(high cut-off) in this case. 

Figure 20 Frequency 30

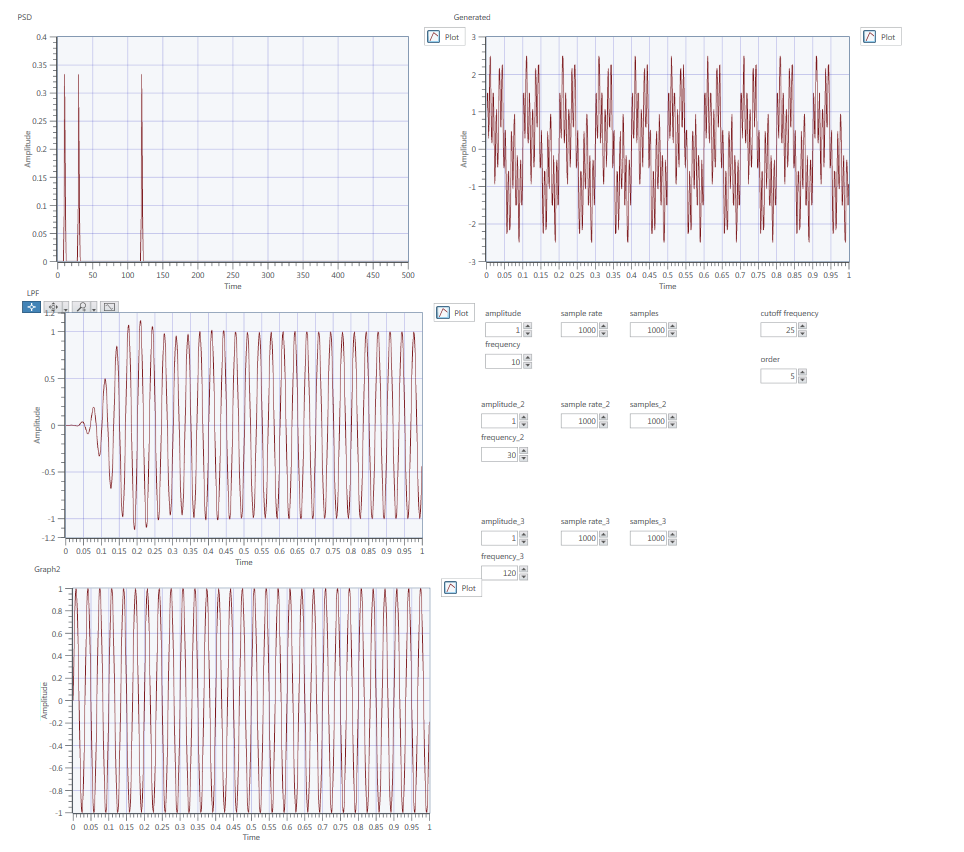
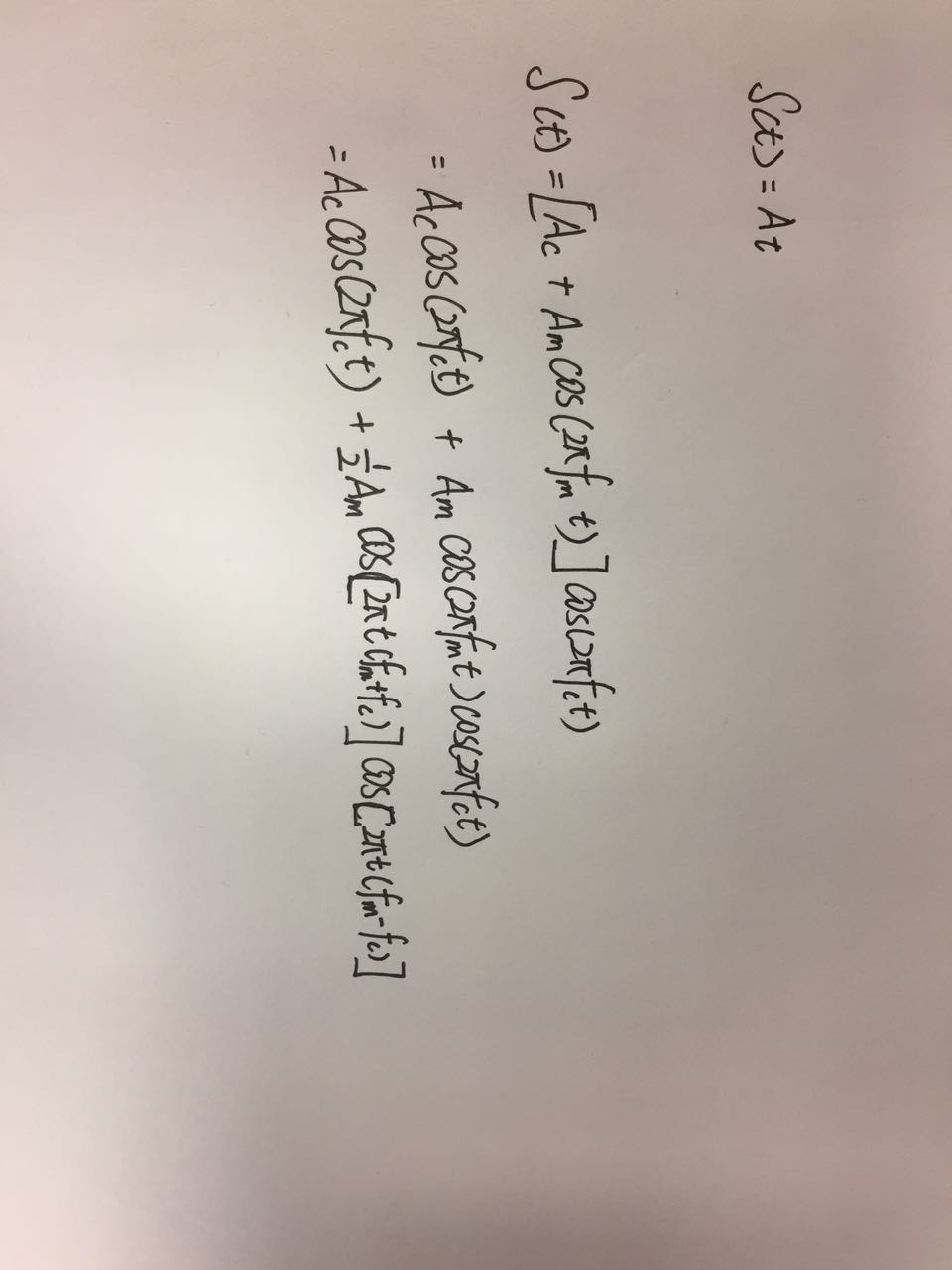


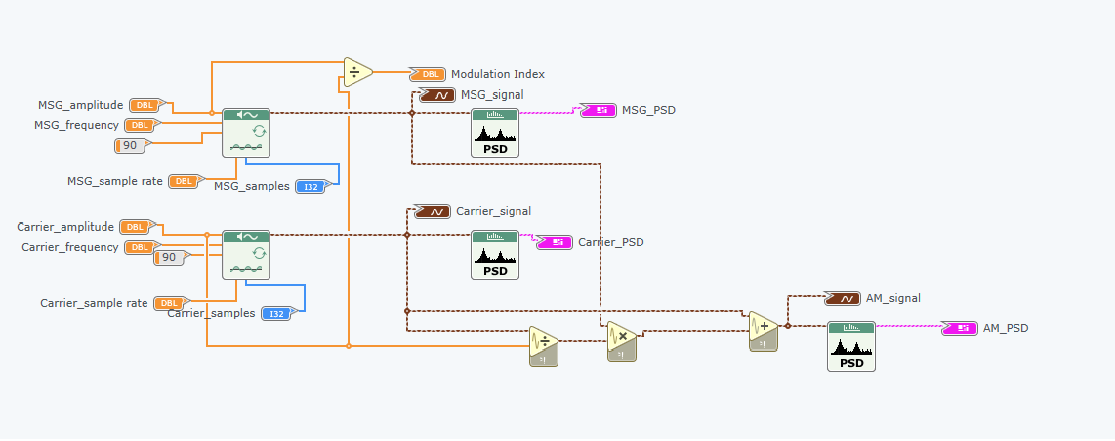
Figure 21 Frequency 30

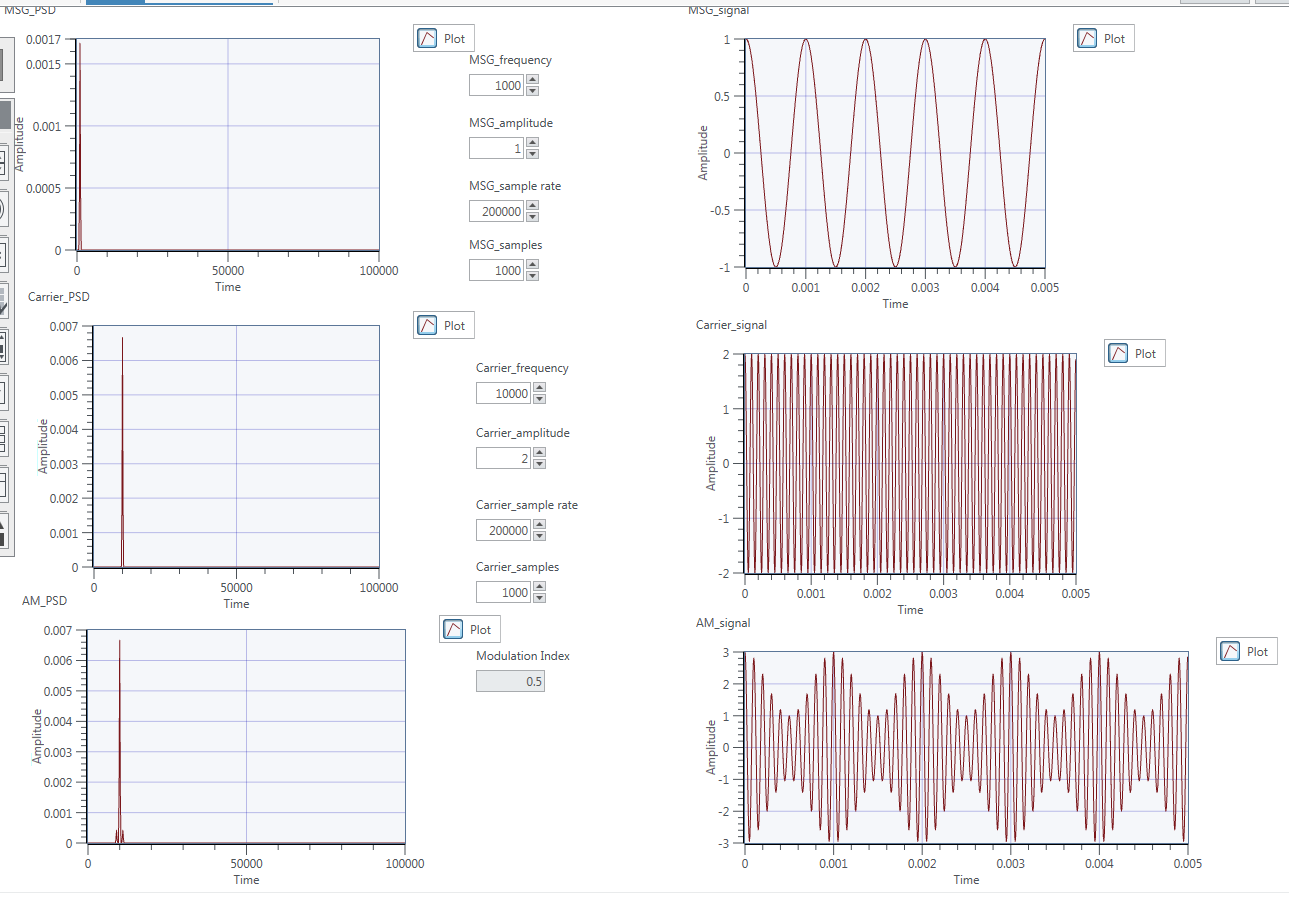
Lab 2: AM Simulation and USRP

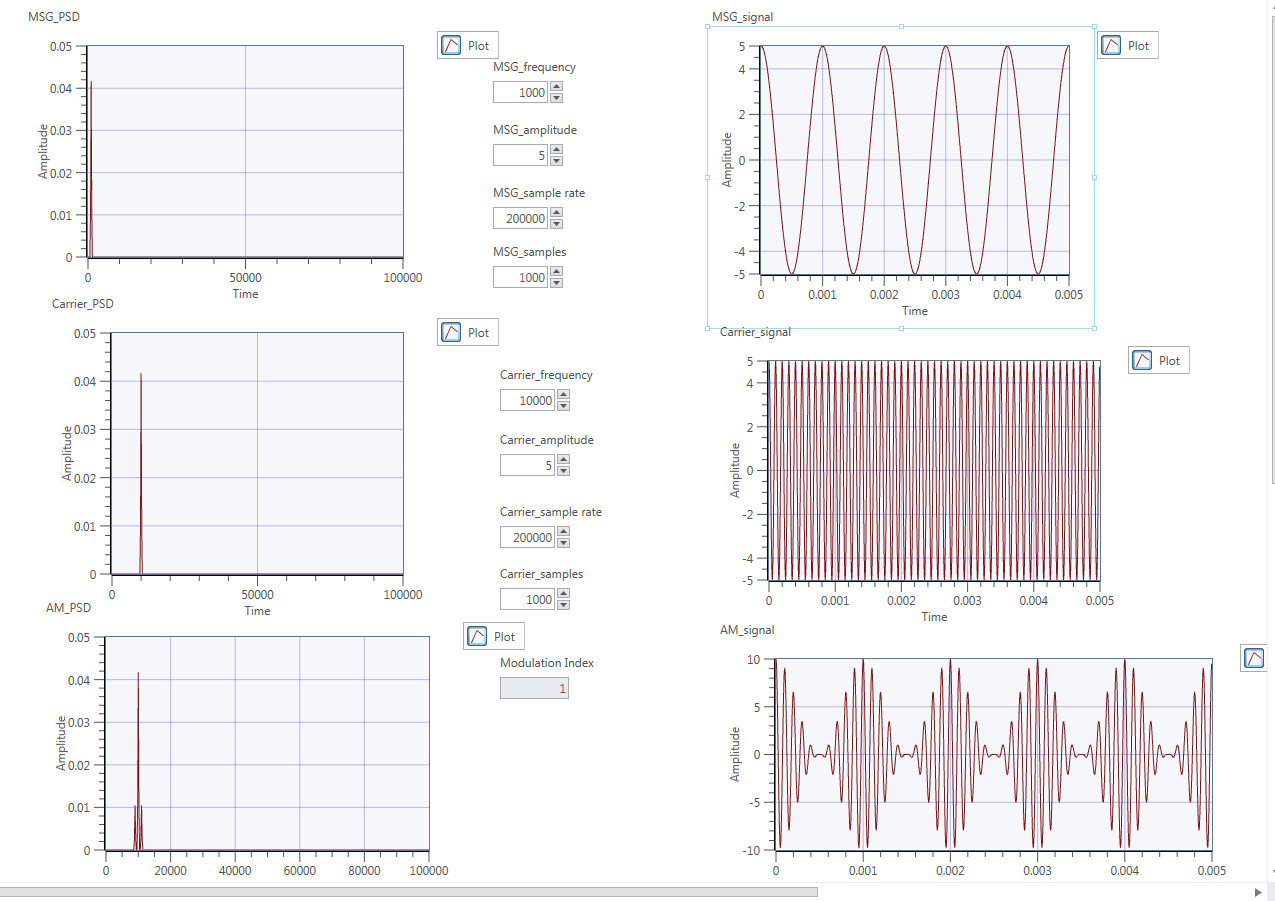
## Ex1-AM Modulator

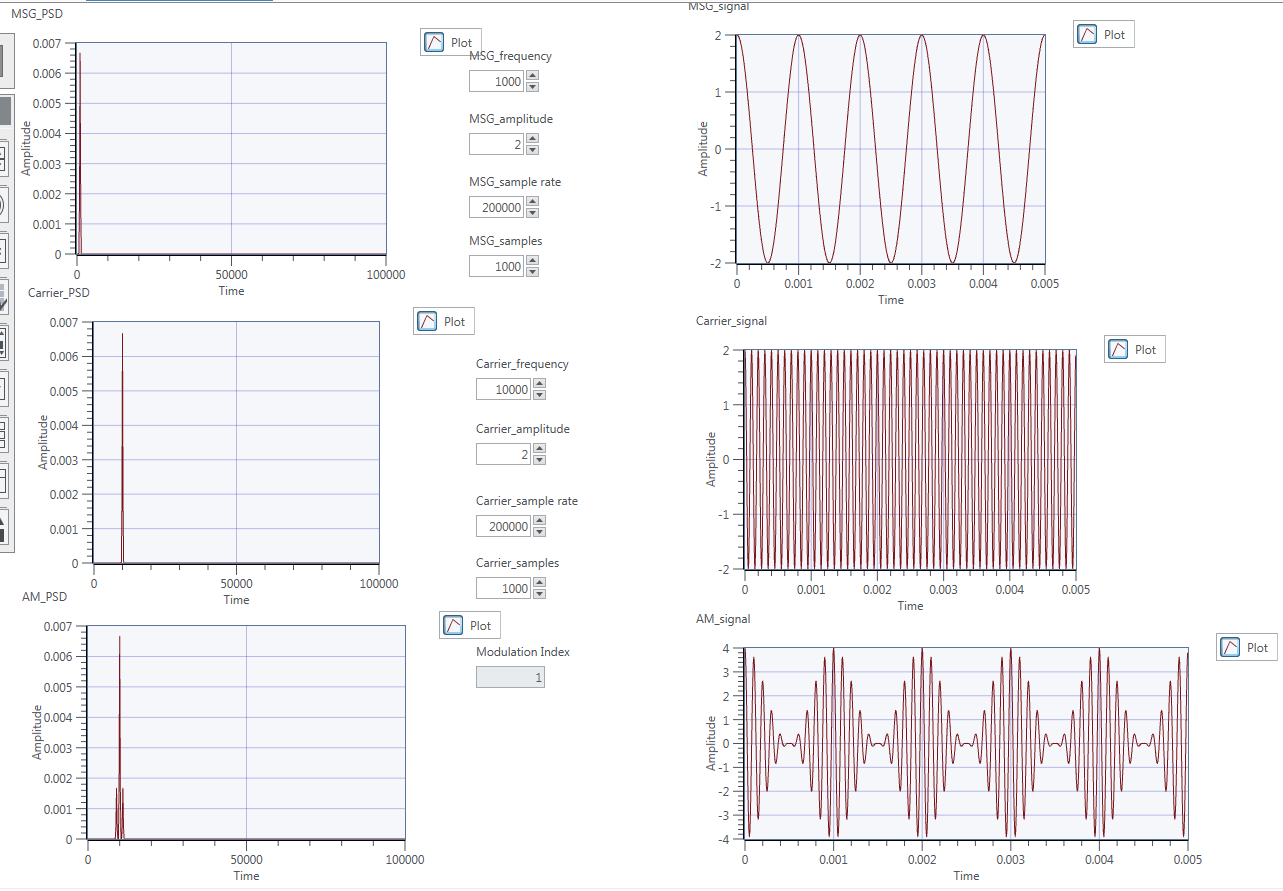
* 𝒔(𝒕)=[𝑨𝒄+𝑨𝒎𝐜𝐨𝐬(𝟐𝝅𝒇𝒎𝒕)]𝐜𝐨𝐬(𝟐𝝅𝒇𝒄𝒕) 
* Modulation index: 𝜇=𝐴𝑚/𝐴𝑐

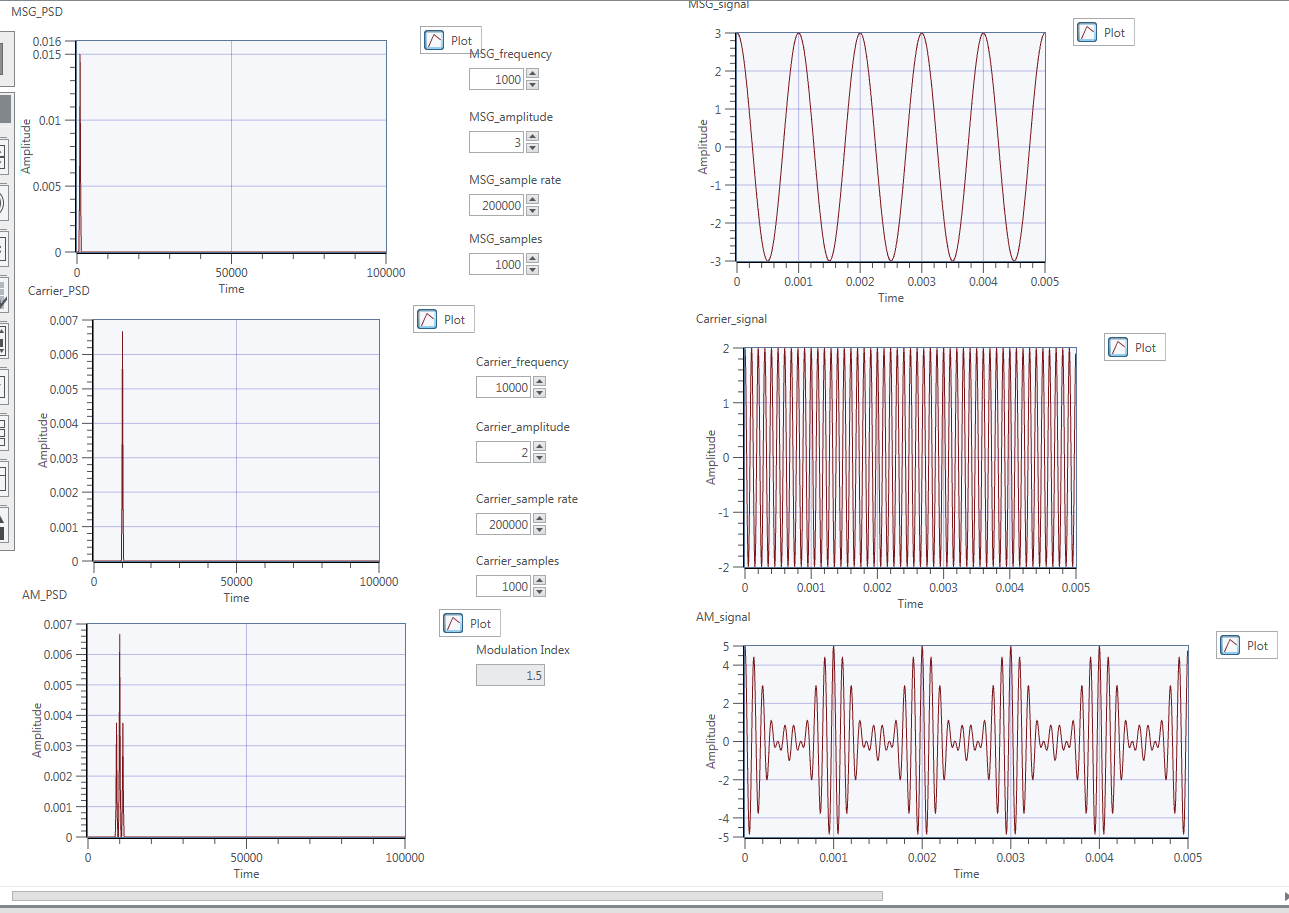
S(t) can also be expressed into the expresion in figure 22. So in the AM.gvi, first we have two sine waveform generator to generate carrier and message signal. We use a divide function to get cos(2𝝅𝒇𝒄𝒕), then a mutiplier function is used to get the combined waveform. Sum the signals (figure 22 euqation) to obtain AM signal.











Above, we change the modulation index from 0.5 to 1.5, we get the conclusion that the smaller the modulation index, the less varies of the unmodulated level.

