**Michigan Technological University**

**EE 5750**

**Distributed Embedded Control Systems**

**Modeling in Motohawk and Calibration in Mototune**

**Lab Experiment 01**

**Submitted By:** Kirk D’Souza

Lopamundra Baruah(Lab partner)

**Lab Instructor:** Ming Cheng

**Date Performed:** 1/25/2016

**Date Submitted:** 1/29/2016



**Introduction**

All modern and luxury vehicles are mainly compromised of electronics. Monitoring and managing these electronic systems is of prime importance. Electrical Control Modules are the main controlling units for the electrical systems in automobiles. Modern vehicles have multiple ECUs compromising of embedded software with increasing complexity. Some examples are vehicle door control units, engine control units, seat control, speed control etc. Management of these sophisticated electronic systems is an arduous task for the original manufacturers. Understanding how ECUs function goes a long way in meeting that task.

This lab familiarizes us Model-based embedded system design with an introduction to Motohawk and Mototune. Motohawk is necessary for designing and building Control modules using the Simulink software, and Mototune is a real-time calibration software used for checking, and calibrating the results of those models. The Mototune software is connected with the desktop simulator (ECU-565-128) via the CAN and harness

**Systems Input/ Output and Control Logic**

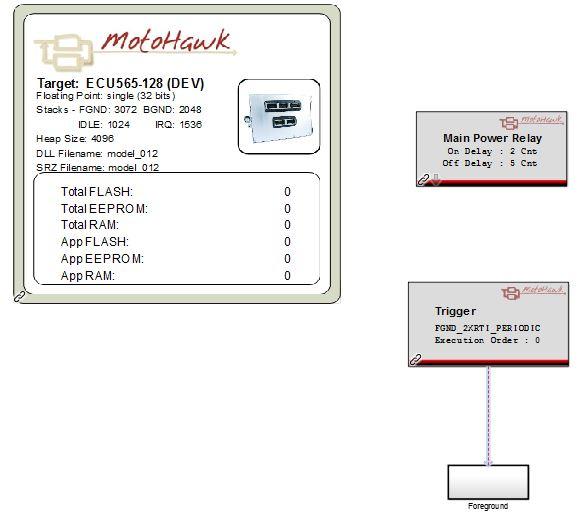
The System inputs are 2\* pi, frequency and a T sec that help formulate the Sime and Cosine functions.

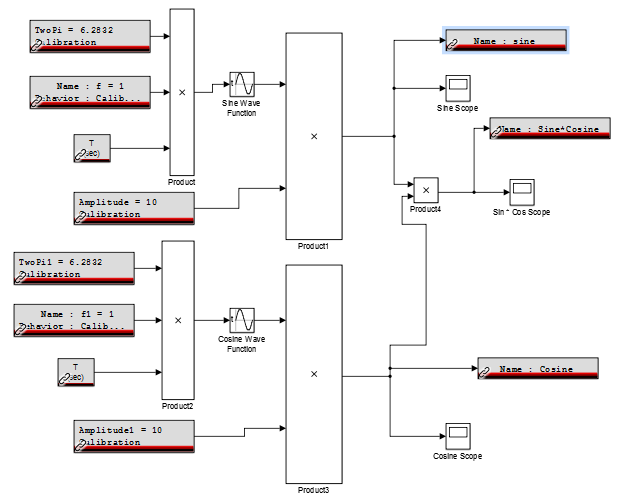
The Amplitude for both those functions is set to a default of 10

Our User defined function, Sin\* Cosine, is derived from the outputs of the Sine and Cosine functions and does not have its own amplitude or frequency settings. Therefore, whatever change is made in the frequency or amplitude of the Sine and Cosine function inputs is also reflected in the Sin\*Cosine function output

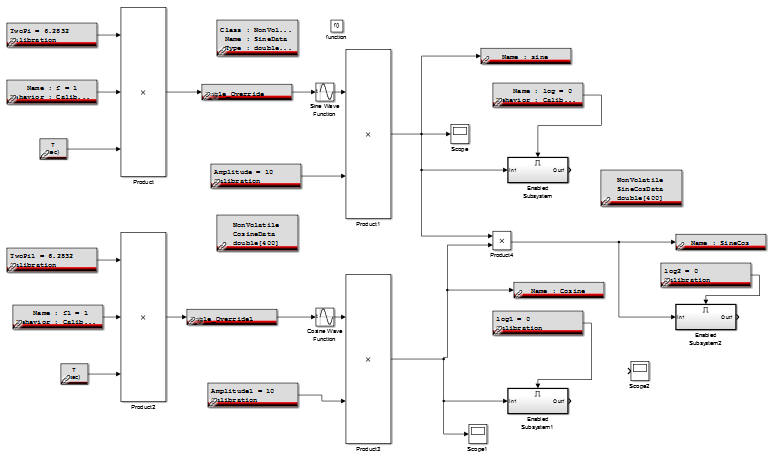
**Model Development**

The main model is as given below:



The intermediate foreground model is as given below:

The final foreground model is as given below:



**Simulation and Calibration Results**

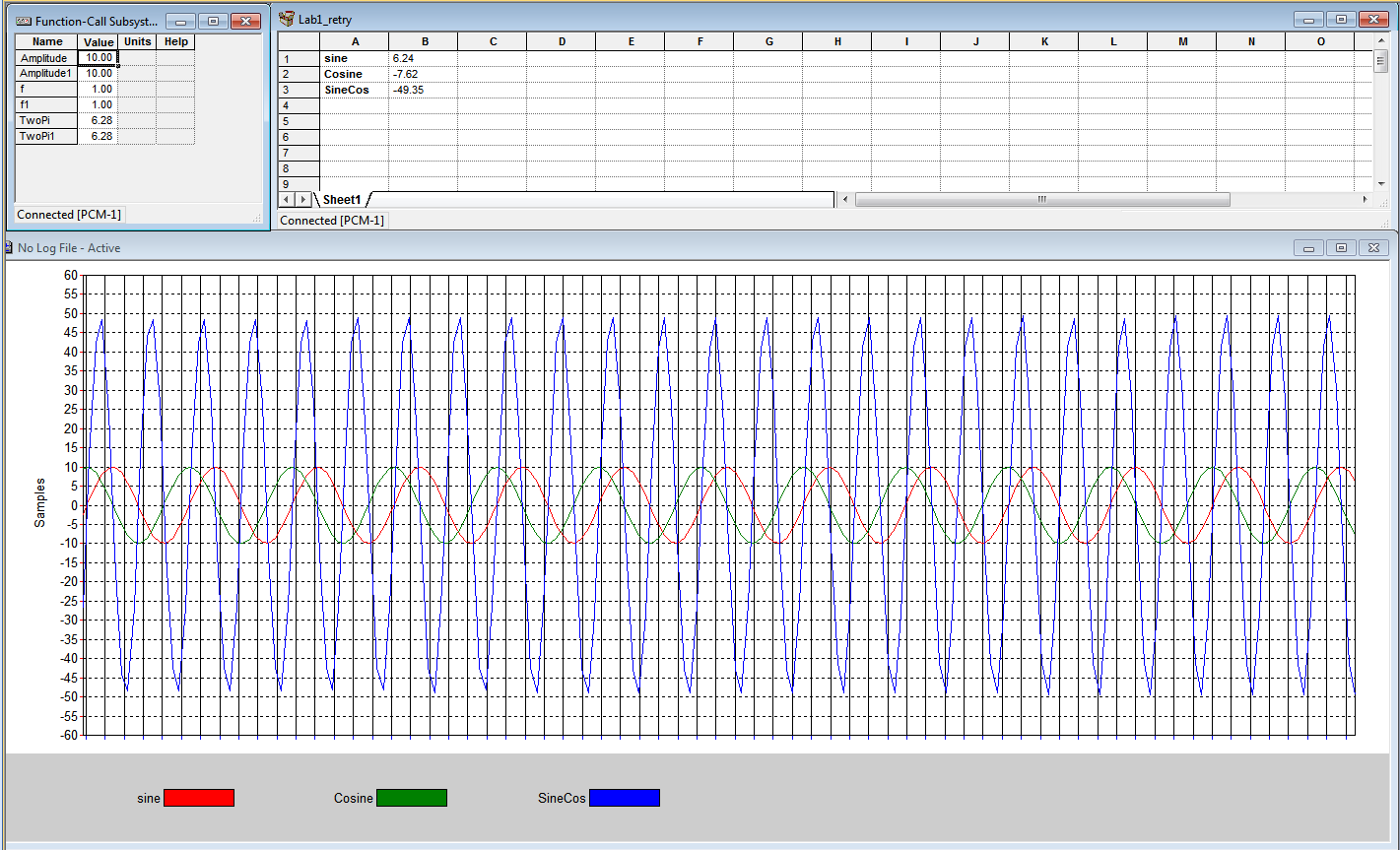
**Results for Sine, Cosine, Sin\*Cos functions with parameters:**

1. Sine Amplitude = 10

Cosine Amplitude = 10

Sine Frequency = 1

Cosine frequency = 1

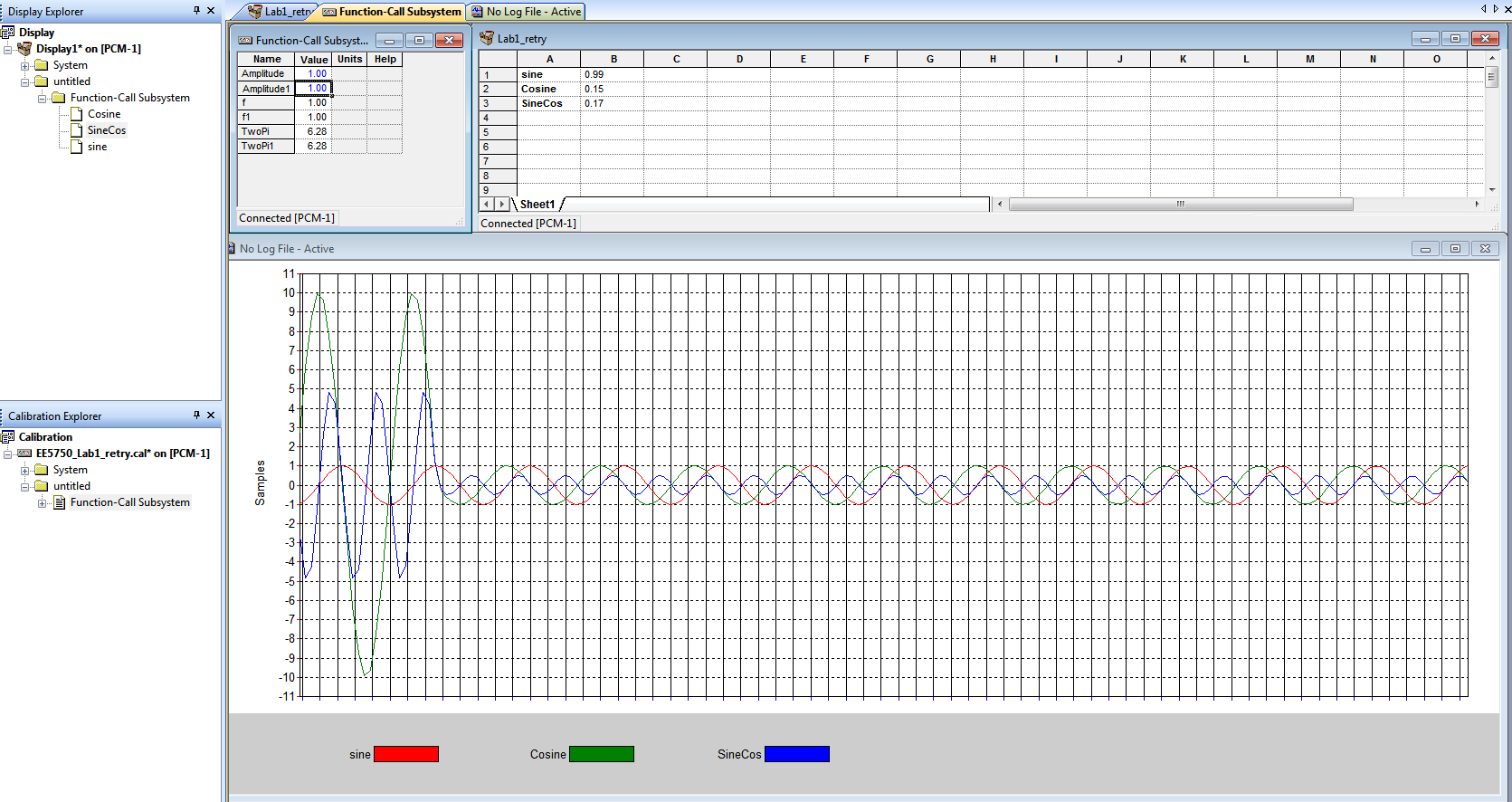


1. Sine Frequency = 1

Cosine Frequency =1

Sine Amplitude = 1

Cosine Amplitude = 1

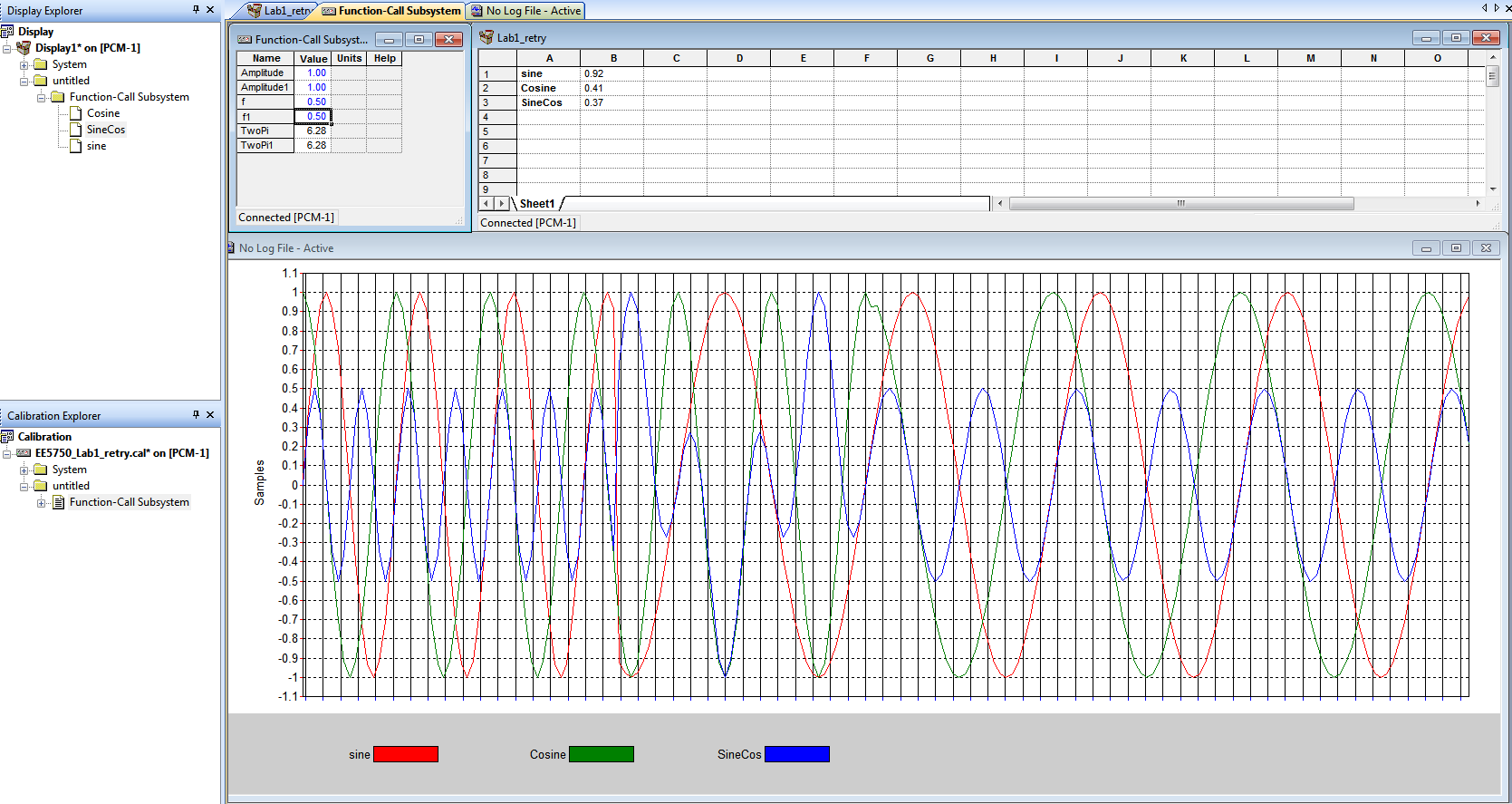


1. Sine Frequency = 0.5

Cosine Frequency = 0.5

Sine Amplitude = 1

Cosine Amplitude = 1

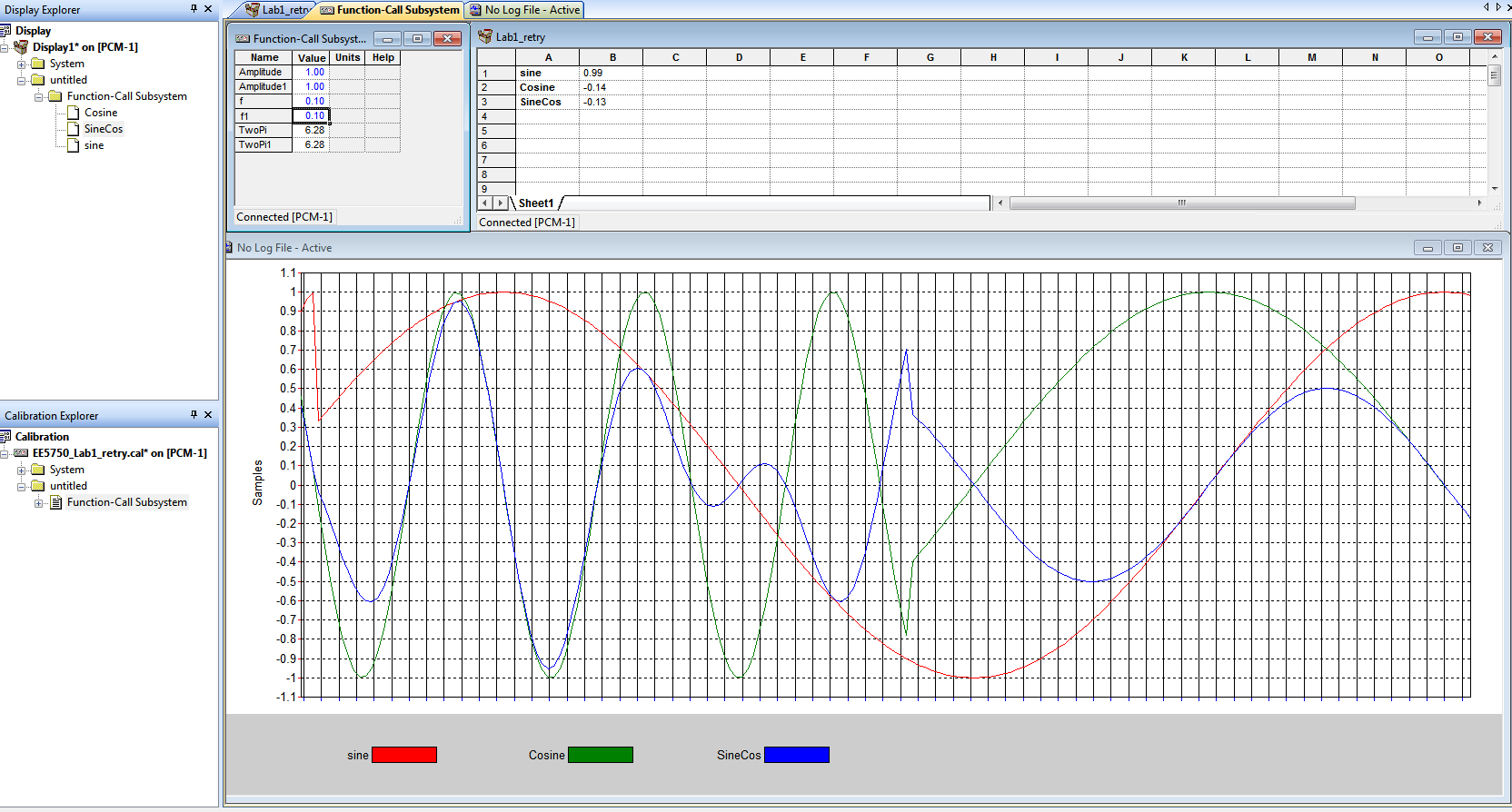


1. Sine Frequency = 0.1

Cosine Frequency = 0.1

Sine Amplitude = 1

Cosine Amplitude = 1

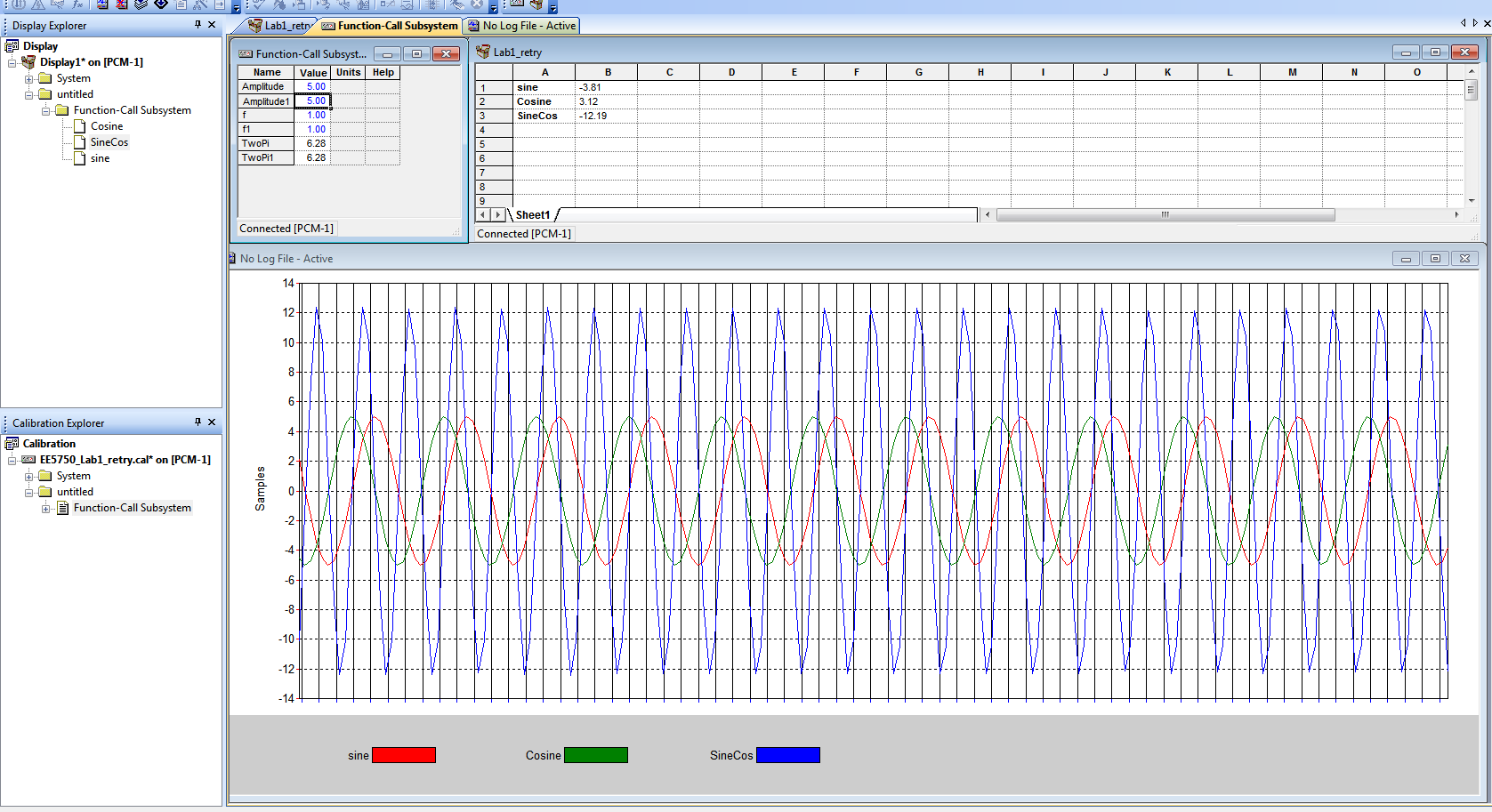


1. Sine Frequency = 1

Cosine Frequency = 1

Sine Amplitude = 5

Cosine Amplitude = 5

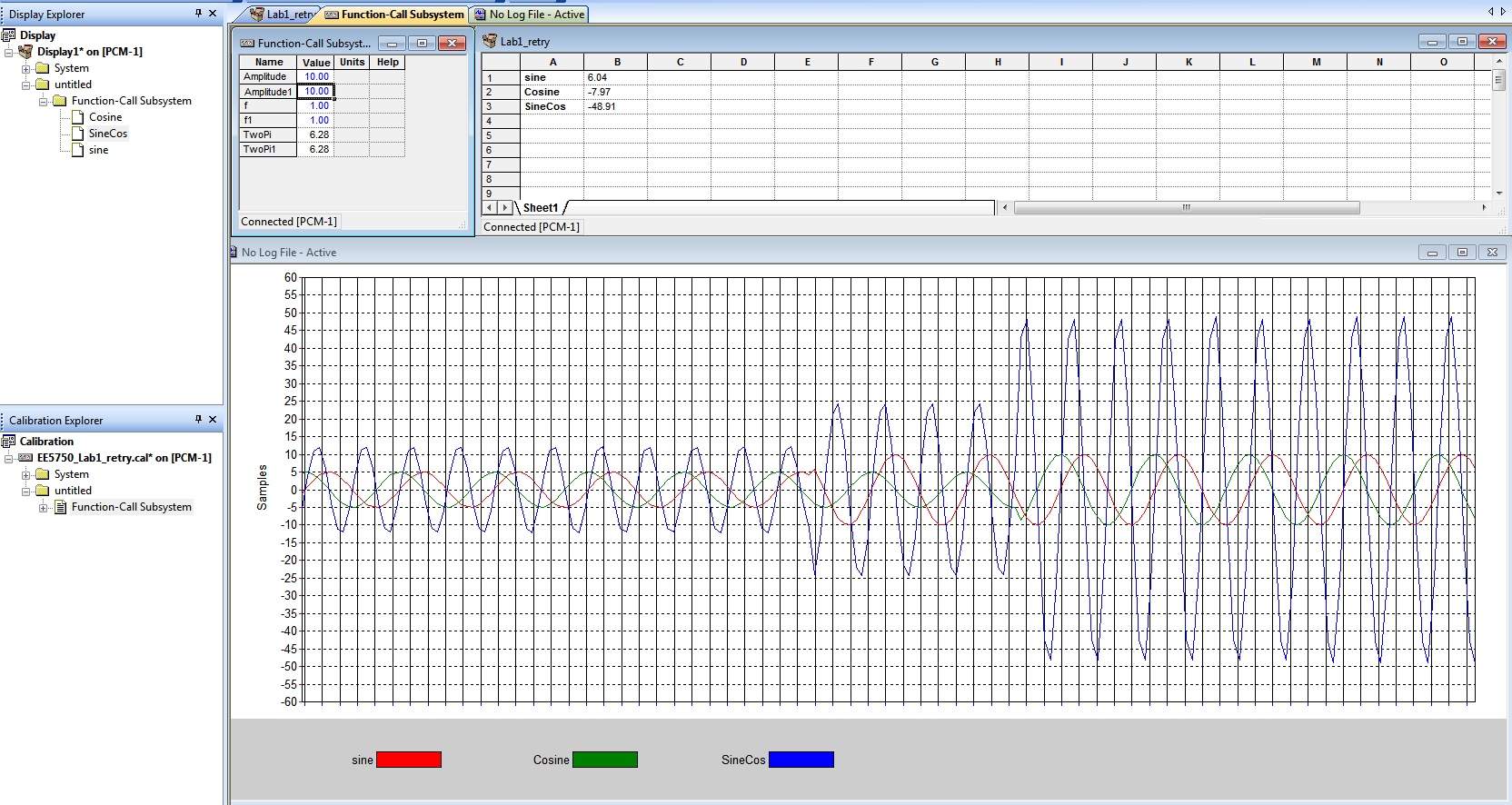


1. Sine Frequency = 1

Cosine Frequency = 1

Sine Amplitude = 10

Cosine Amplitude = 10



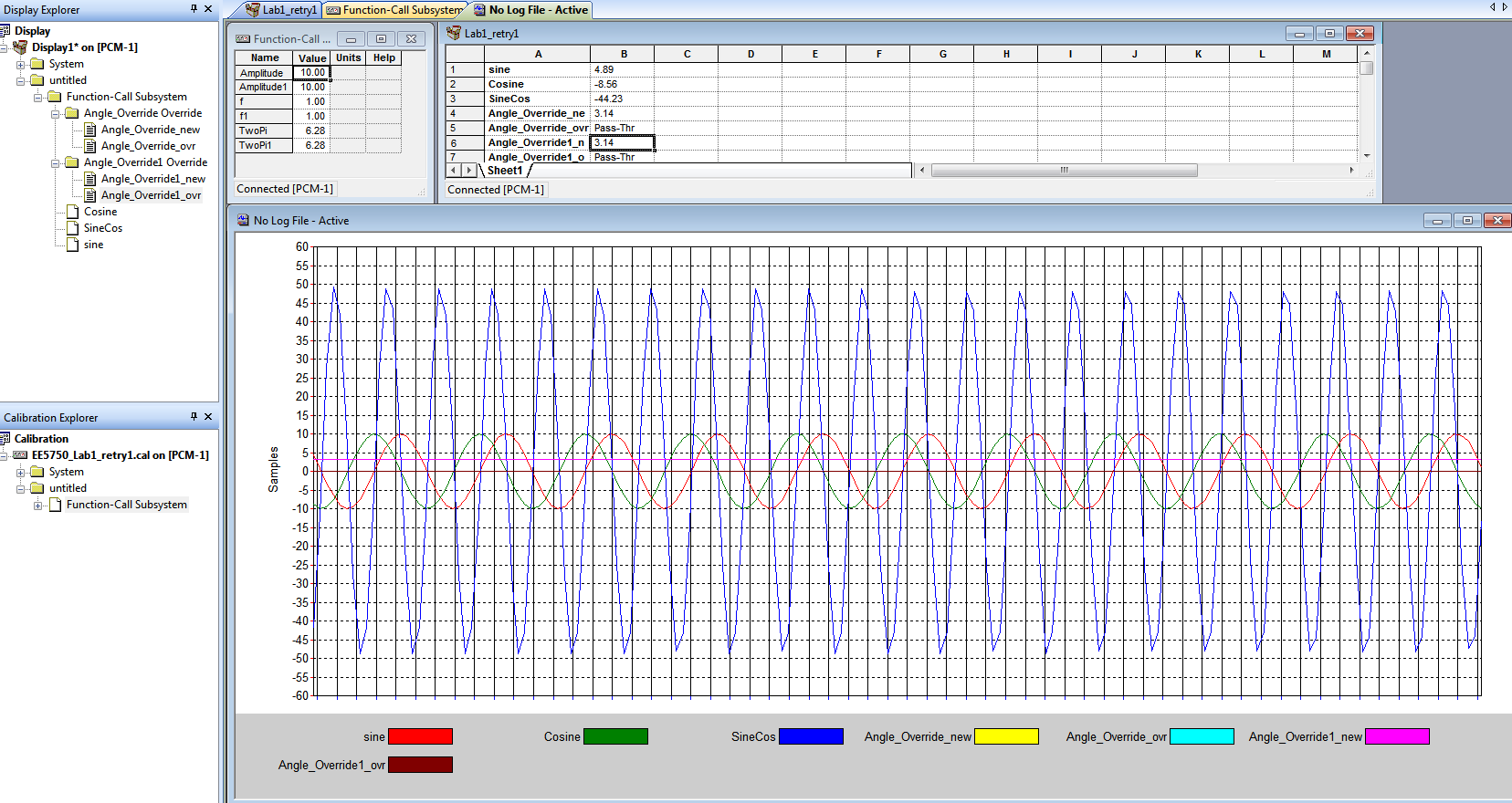
1. Angle Override = PassThrough

Sine Frequency = 1

Cosine Frequency = 1

Sine Amplitude = 10

Cosine Amplitude = 10



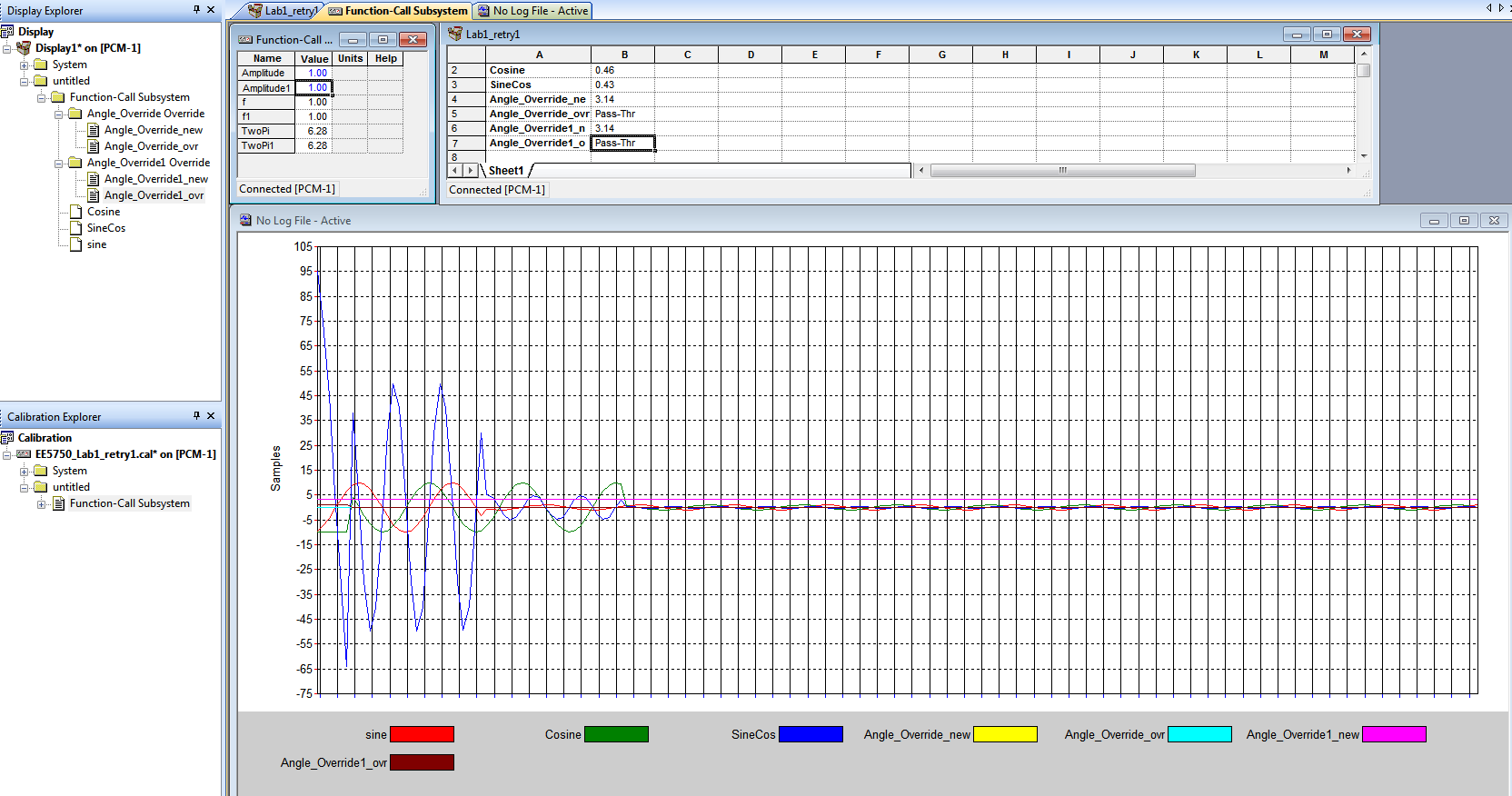
1. Angle Override = Passthrough

Sine Frequency = 1

Cosine Frequency = 1

Sine Amplitude = 1 from 10

Cosine Amplitude = 1 from 10



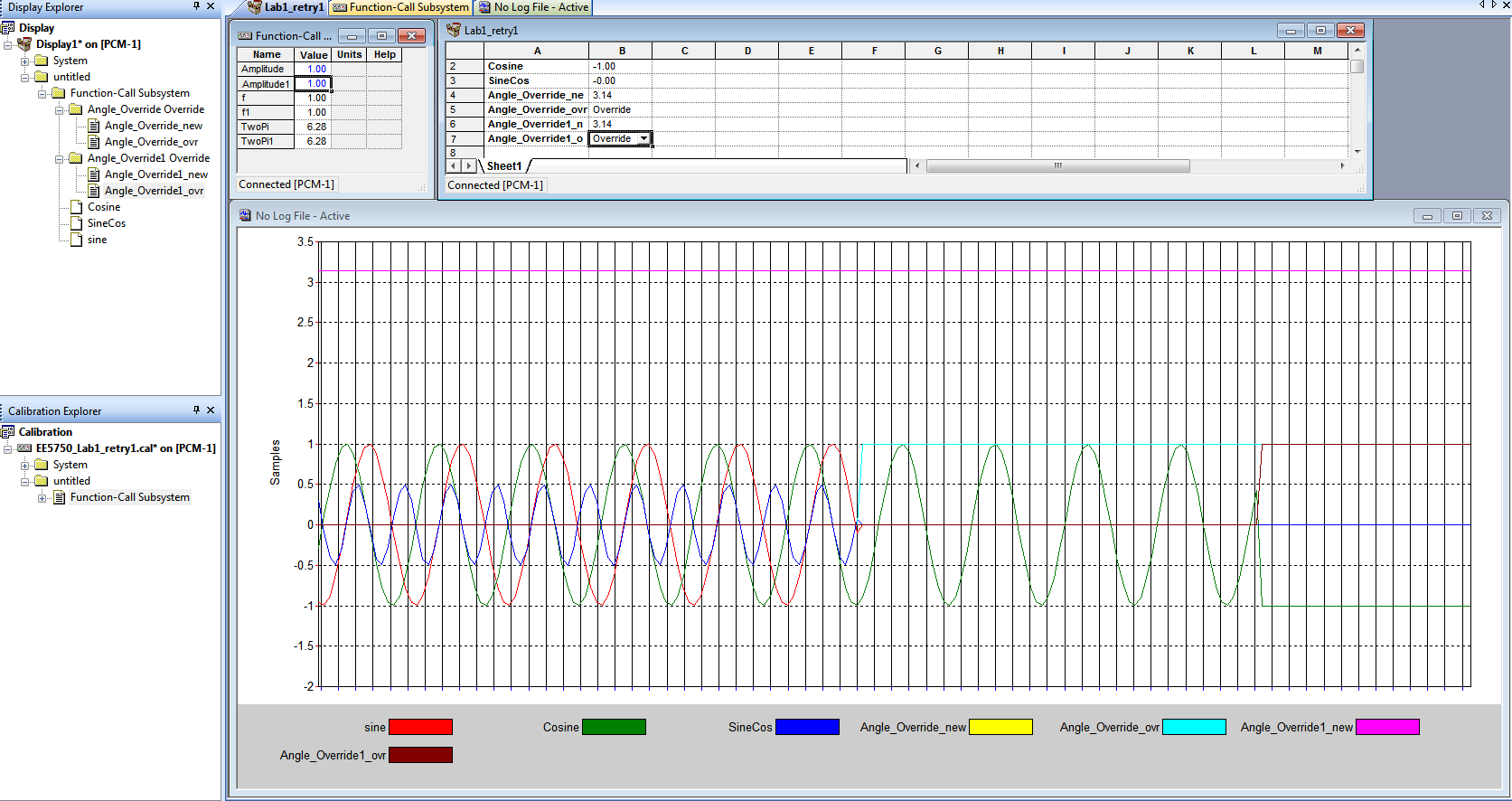
1. Angle Override = Override from Pass-Through

Sine Frequency = 1

Cosine Frequency = 1

Sine Amplitude = 1

Cosine Amplitude = 1



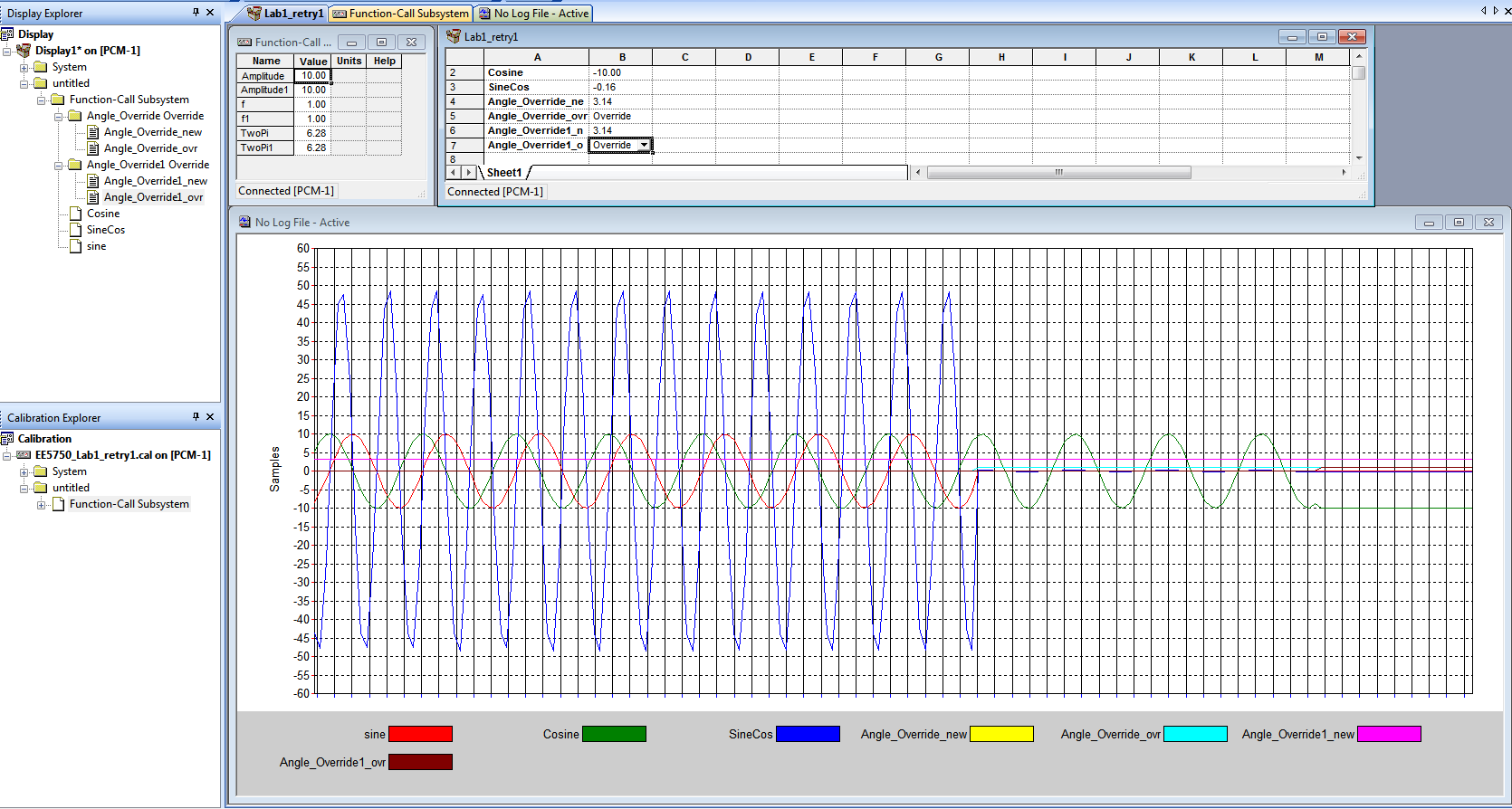
1. Angle Override = Override from Passthrough

Sine Frequency = 1

Cosine Frequency = 1

Sine Amplitude = 10

Cosine Amplitude =10



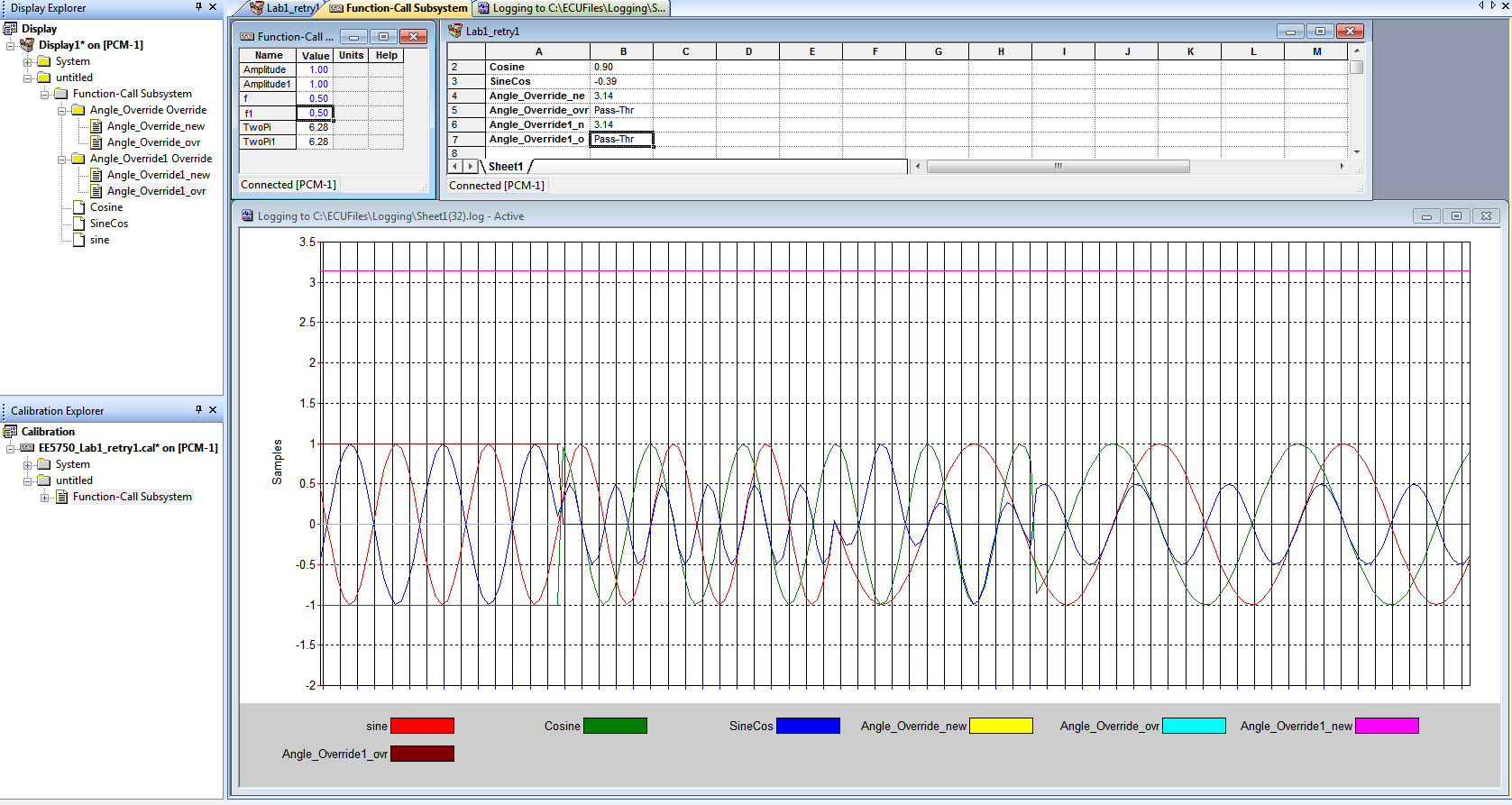
1. Angle Override = Passthrough

Sine Frequency = 1 to 0.5

Cosine Frequency = 1 to 0.5

Sine Amplitude = 1

Cosine Amplitude = 1



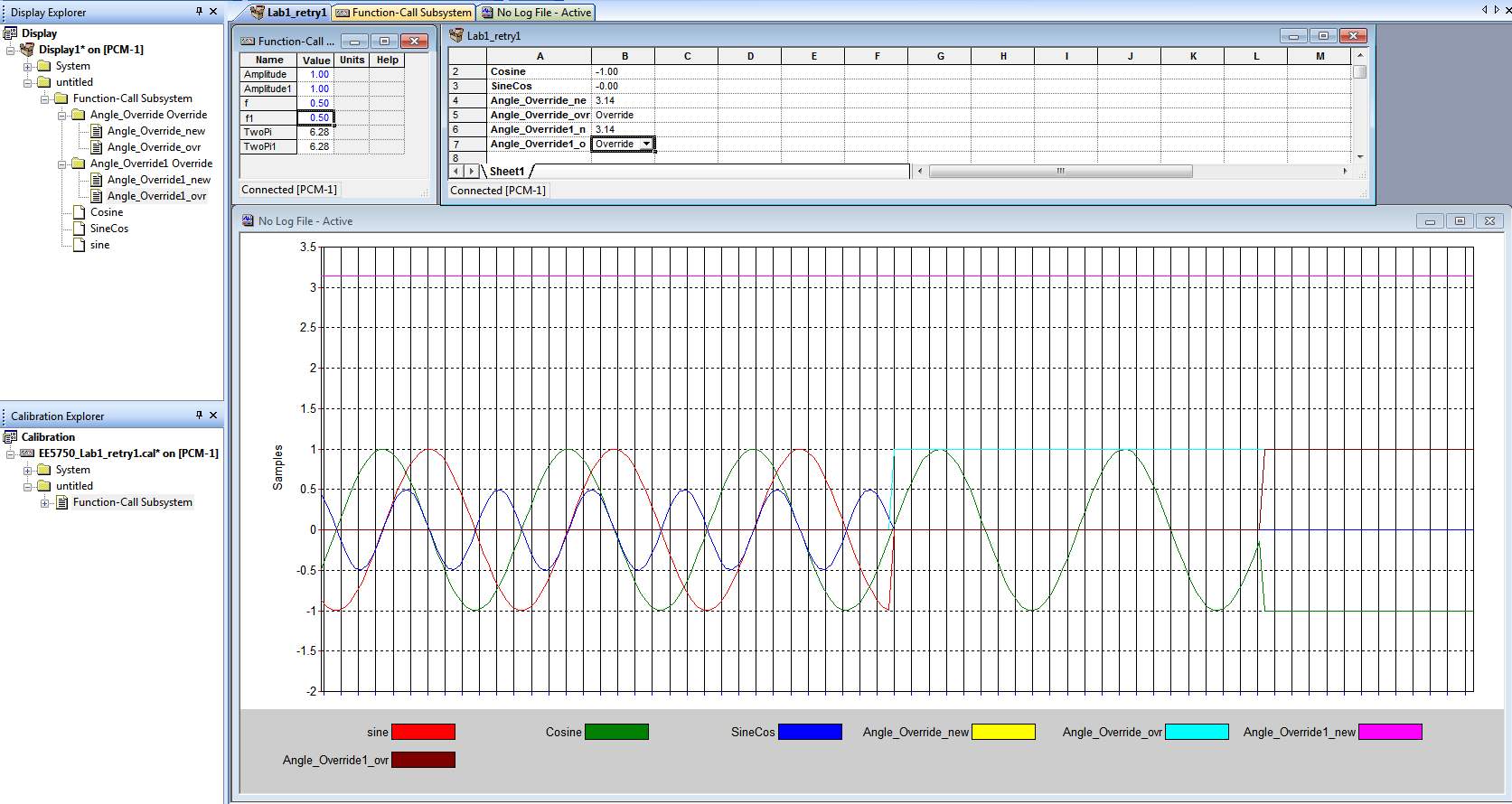
1. Angle Override = Override from Passthrough

Sine Frequency = 0.5

Cosine Frequency = 0.5

Sine Amplitude = 1

Cosine Amplitude = 1



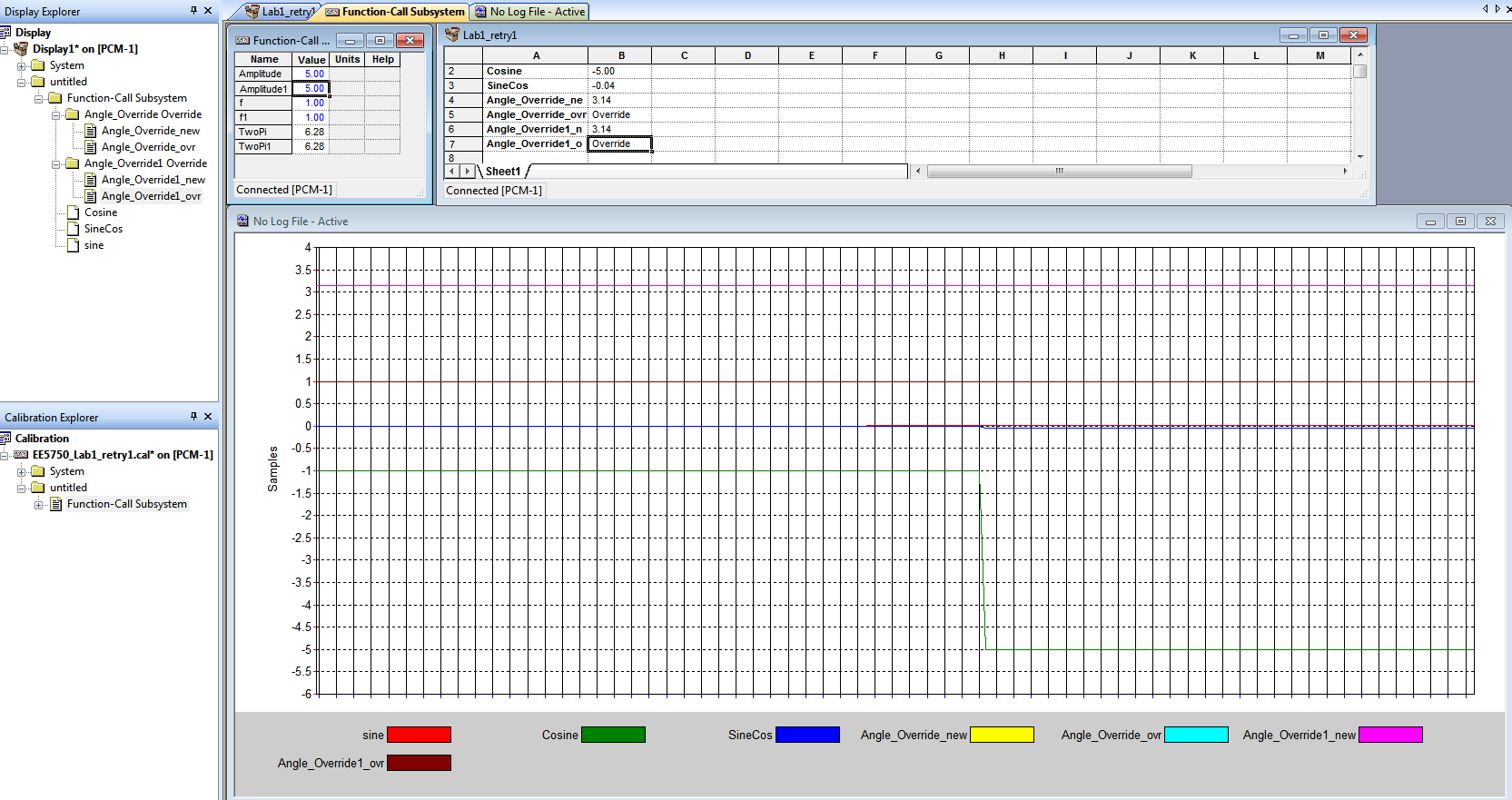
1. Angle Override = Override

Sine Frequency = 1

Cosine Frequency = 1

Sine Amplitude = 1 to 5

Cosine Amplitude 1 to 5



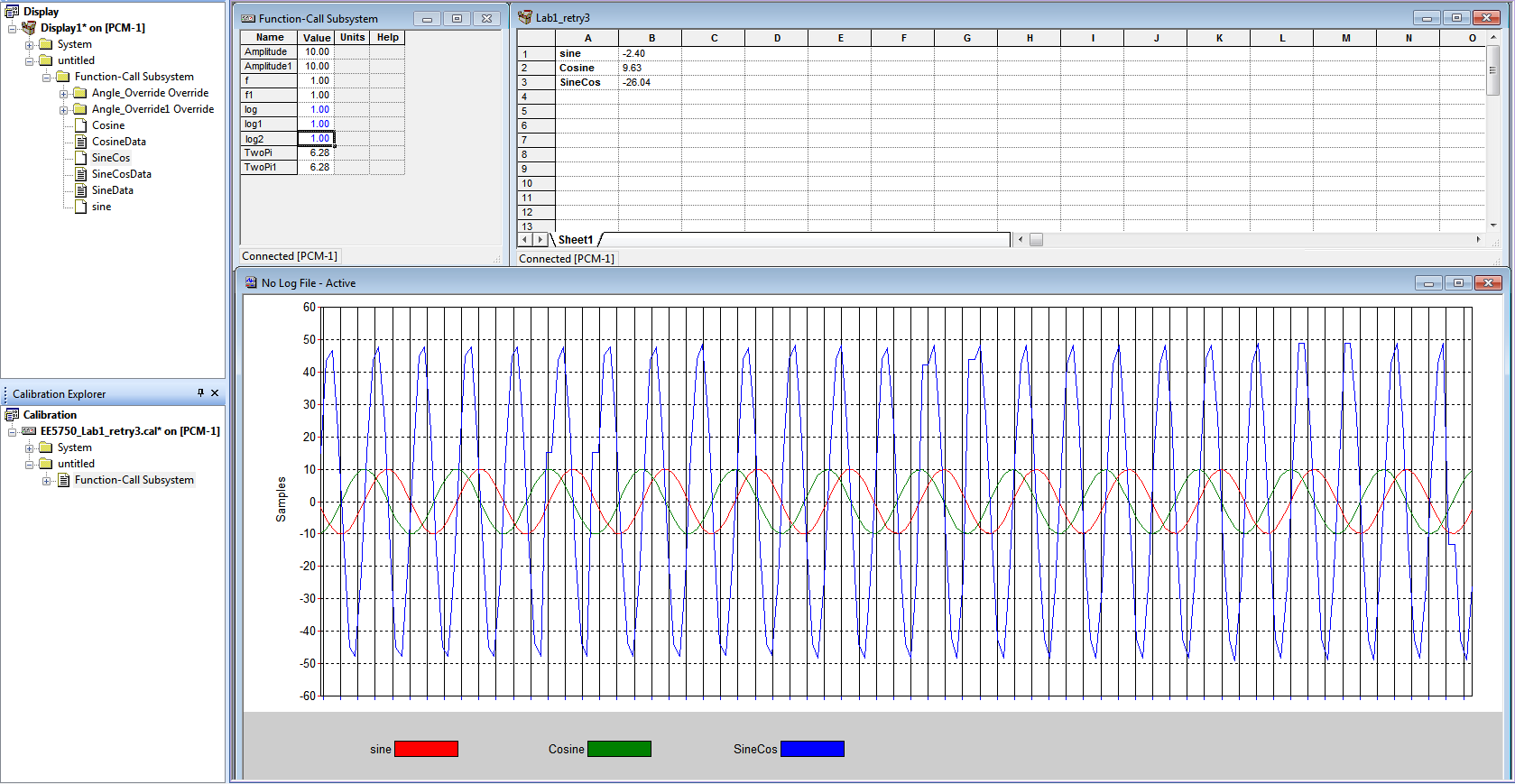
14) Angle Override = Passthrough

Sine Frequency = 1

Cosine Frequency = 1

Sine Amplitude = 10

Cosine Amplitude = 10



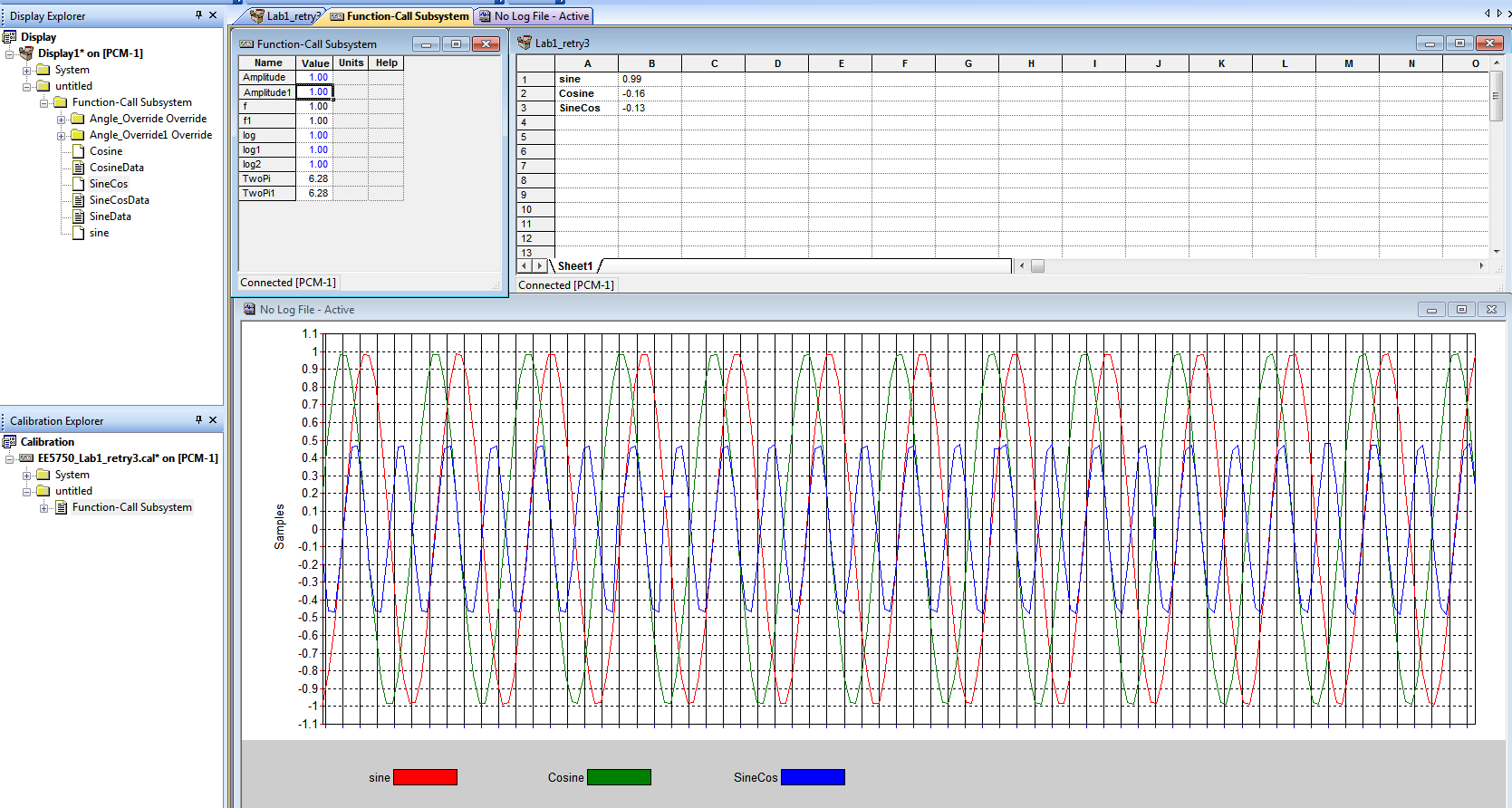
1. Angle Override = Passthrough

Sine Frequency = 1

Cosine Frequency = 1

Sine Amplitude = 1

Cosine Amplitude = 1



The Data Write Block data of data vector length of 400 for Sine frequency = Cosine Frequency = 1, Sine Amplitude = Cosine Amplitude = 1, and Angle\_Override\_ovr = Pass-Through, is given below in the excel table:

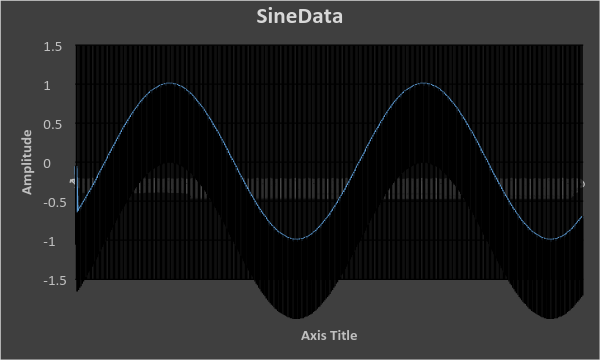


A sample of the above data is given below:

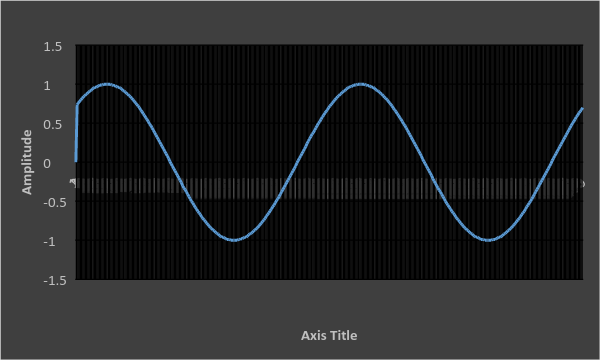
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SineData | -0.05 | -0.65 | -0.63 | -0.6 | -0.58 | -0.55 | -0.52 | -0.5 | -0.47 | -0.44 |
| CosineData | 0 | 0.74 | 0.76 | 0.78 | 0.8 | 0.82 | 0.84 | 0.85 | 0.87 | 0.88 |
| SineCosData | 0.74 | 0.05 | 0.08 | 0.11 | 0.14 | 0.17 | 0.2 | 0.23 | 0.25 | 0.28 |

The graphs for the above data are plotted below:

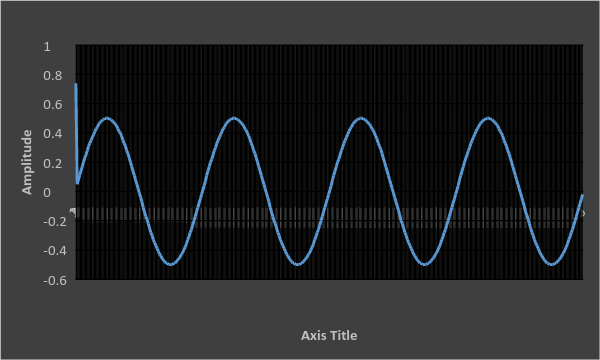
For Sine data:



For Cosine data:



For the User Defined Function (Sine\*Cos) data:



**Validation**

Lab Tasks:

The outputs for Sine wave and Cosine wave with frequencies of 1 and Amplitudes of 1 is given in point 2) in the Results section

The log files for the same are given in sheet1(25)

The outputs for Sine wave and Cosine wave with frequencies of 0.5 and Amplitudes of 1 is given in point 3) in the Results section

The log files for the same are given in sheet1(26)

The outputs for Sine wave and Cosine wave with frequencies of 0.1 and Amplitudes of 1 is given in point 4) in the Results section

The log files for the same are given in sheet1(27)

The outputs for Sine wave and Cosine wave with frequencies of 1 and Amplitudes of 5 is given in point 5) in the Results section

The log files for the same are given in sheet1(28)

The outputs for Sine wave and Cosine wave with frequencies of 1 and Amplitudes of 10 is given in point 6) in the Results section

The log files for the same are given in sheet1(29)

**Discussion**

The above results show that the model designed functions as desired.

The above comparison between the excel plots and the output charts is almost similar especially for the Sine and Cosine waves. The User defined Sin\*Cos function deviates at certain points due to the differences in the data acquisition rate of the simulators.

We can see from the output charts that as the frequency values are lowered, the output curves become smoother.

It is also noticeable from the output chart results that as the amplitude values are raised, the output curves become smoother

Sine and Cosine waves are compared in Pass through and Override in points 8) and 9) in the Results section. Here, the Sine wave remains at the higher amplitude of 1, while the Cosine wave remains at -1

A change of frequency values does affect the Sine and Cosine waves when Angle\_Override\_ovr = Pass-Through as can be seen from point 11) in the Results section

A change of frequency values does not affect the Sine and Cosine waves when Angle\_Override\_ovr = Override as can be seen from point 10) and point 12) in the Results section

A change of Amplitude values does affects the Cosine wave when Angle\_Override\_ovr = Override as can be seen from the Cosine output in point 13) in the Results section

A comparison between the logging sheets i.e. sheet1(34) and sheet1(35), and the output chart results in point 15) of the results section show that the data is plotted accordingly

**Conclusion**

We can conclude that better stability is achieved when the frequency is lowered and the amplitude is raised. It is also conclusive that a change in frequency when the signals are being overridden has no effect on the overridden outputs.

This lab has helped us understand the proper use of Motohawk and Mototune software and how they can be used to design embedded system models and calibrate them according to requirements

**References**

1. <https://en.wikipedia.org/wiki/Electronic_control_unit>
2. <http://www.mathworks.com/help/>
3. <http://www.mathworks.com/help/simulink/>
4. <http://www.woodward.com/mcssoftware.aspx>