Creating a Communications and Remote Control Environment (Labview 2)

Sejin Jeon
Sogang University Physics Department
Student ID 20231262

(11th Week Post-Experiment Lab Report)

I. THE ARBITRARY FUNCTION GENERATOR & DIGITAL OSCILLOSCOPE

A. Example 1 (Data & Analysis)

For the first example, the following sub-VI's were used to make a code that inputs frequency, waveform, and amplitude as parameters and creates a waveform that is displayed on the function generator.

1. "Initialize.vi"
2. "Enable output.vi"
3. "Configure Standard Waveform.vi"
4. "Close.vi

As seen in the images below, the VISA resource name is first registered into the code, and afterwards, the library/drive that is separately downloaded initialises the whole process, where the necessary outputs are connected to the configure standard waveform VI. This allows all the details of the graph to be inputted, in this case the amplitude, frequency, waveform function, and channel correctly registered through the blocks of the code. Next, the enable output VI turns the actual formed waveform on, and if there is no error, the last two codes are ignored. If there is an error, the close VI detects it, and sends a appropriate response through the error out block.

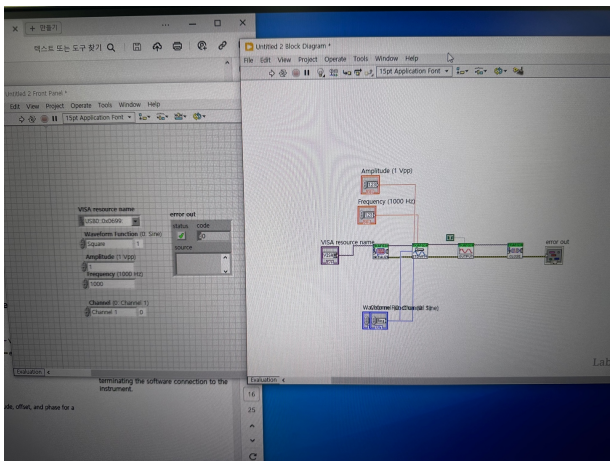

FIG. 1

The images above show the block diagram and implemented function generator screen.

FIG. 2

B. Example 2 (Data & Analysis)

In the second example, the example above was extended, where the code was modified such that the code would also input frequency and amplitude. The implementation looks like what follows.

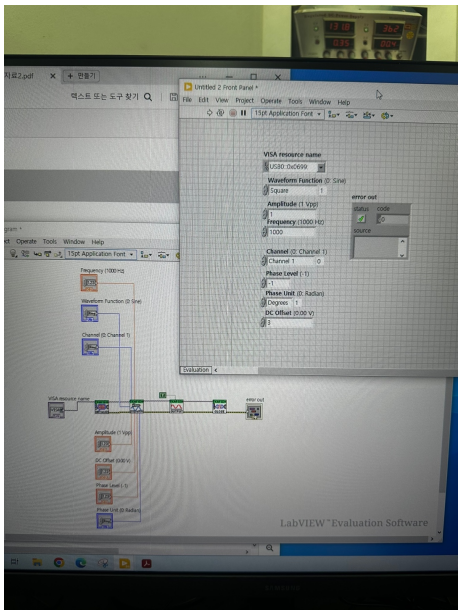

FIG. 3

As an extension, first, the frequency sweep was initiated

FIG. 4

through the function generator, with the start frequency, stop frequency, sweep time, and amplitude as parameters. This was next created as code through labview, and the implementation looks like the following.

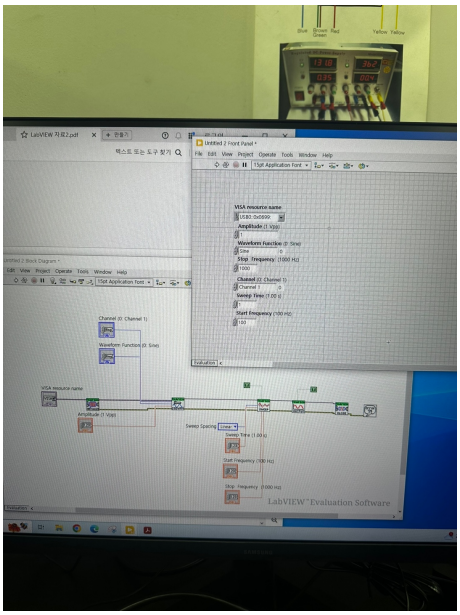

FIG. 5

The code follows the form of the first function up to the enable output VI, where, this time, the frequency sweep option is connected and turned on, with additional parameters that help regulate this process.

FIG. 6

C. Example 3 (Data & Analysis)

In the third example, the amplitude modulation function was used through the function generator. This made a certain envelope for the function generator to create a waveform in. The same technique was then made through Labview code, as you can see below.

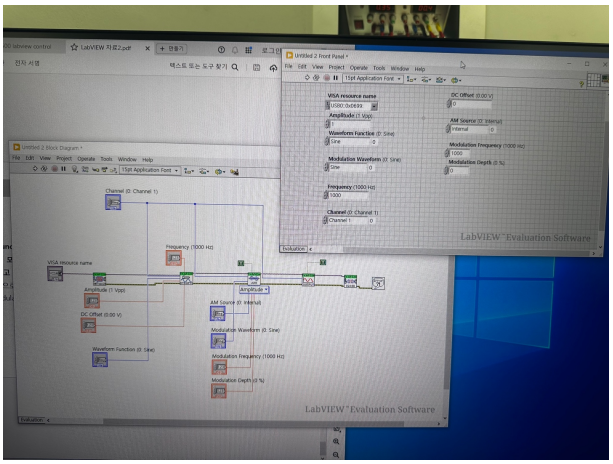

FIG. 7

In the code above, instead of the function sweep option turned on, the amplitude modulation option is turned on, where the output enabling block is replaced with the amplitude block.

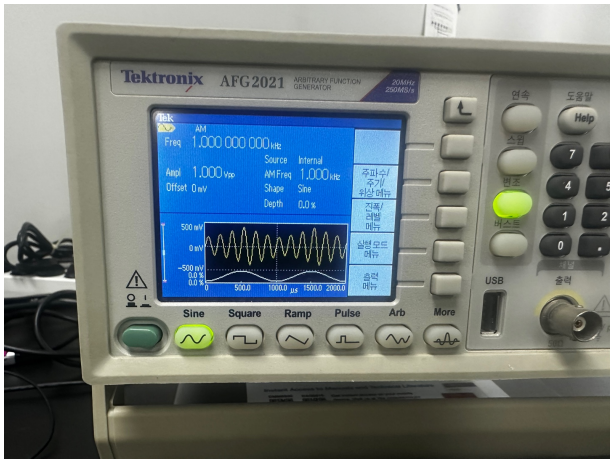

FIG. 8

II. DMM6500 & LABVIEW CONTROL

A. Simple Example 1 (Data & Analysis)

In the first simple example in the DMM6500 practice, we created a live DC voltage measuring code, which looked like the following.

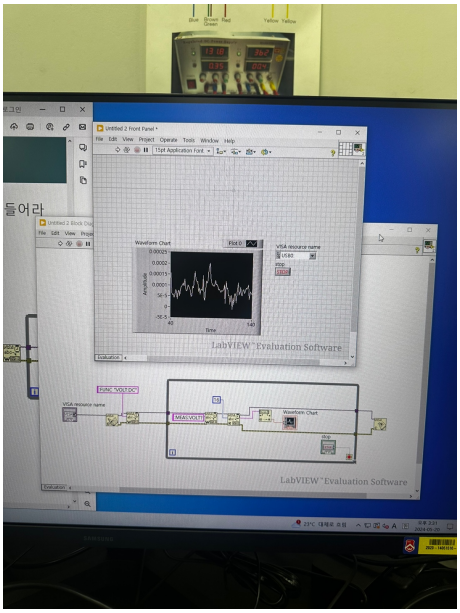

FIG. 9

The code is a while argument in its entirety, where a waveform chart is created through appropriate blocks.

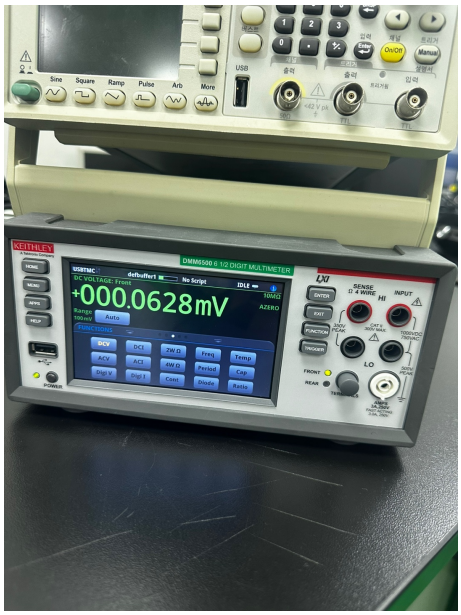

FIG. 10

B. Simple Example 2 (Data & Analysis)

In the second simple example made through LabVIEW, a program was made such that it would measure the current flowing through a certain material that would have a voltage initiated to it. Using a 1000 Ω resistor, we tested whether the VI plot would give a proper value of the resistance, which it certainly did observable through the graph seen below.

FIG. 11

FIG. 12

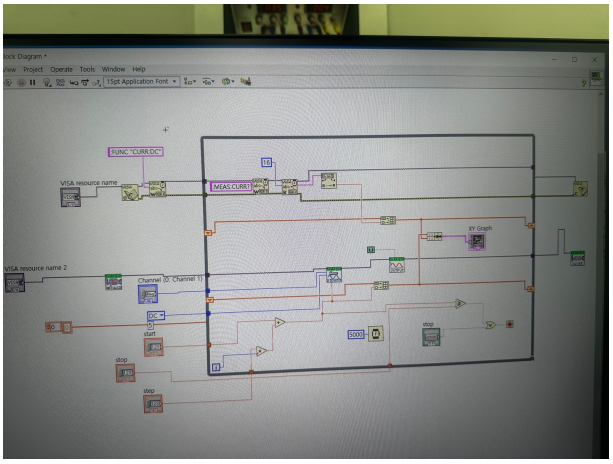

FIG. 14

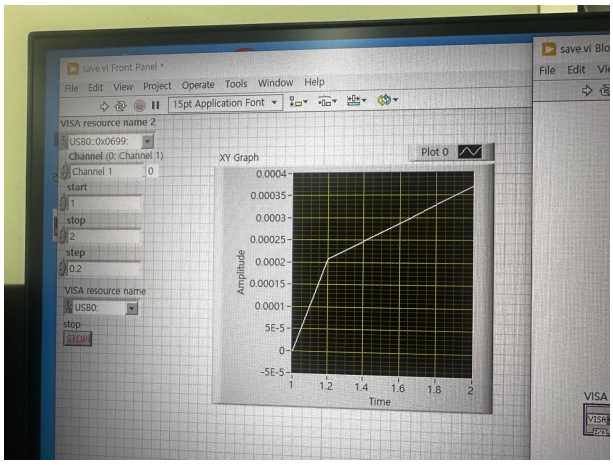

FIG. 13

In the code, the input voltage is first sent to the scan from string function, and directly sent into a block that collects certain data points and makes it into an array. The other set of data points are taken from the time flowed in the system. The other axes that the graph takes is precisely the current, were the DMM6500 sends an appropriate measured current of the two ends into the system for it to match it up with the voltage. Together, it constantly (live) creates a graph for the measuring the voltage per current that the DMM6500 measures.

III. DISCUSSION

Throughout the experiment there were a few errors that needed to be evaluated and there were also quite a few improvements that could be made. They are listed as paragraphs below.

Synchronization issues between the DMM6500 and AFG1022. If the data acquisition timing between the digital multimeter (DMM6500) and the function generator (AFG1022) is not properly synchronized, the measured current and applied voltage might not correspond to each other at the exact same time instances. This misalignment can result in an inaccurate current versus voltage graph, as the data points plotted would not reflect the true relationship between the current and the voltage. A way of mitigating this certain type of error is to properly time-stamp the operation. This may produce a more accurate plot.

The following is a reason to why the graph might not have been perfectly linear, only exhibiting proper characteristics in the latter part.

Presence of a non-ohmic component in the circuit. If the circuit includes elements that do not adhere to Ohm's law, such as diodes or transistors, it is well know that the current versus voltage relationship might not be linear. These non-ohmic components exhibit a nonlinear current response to the applied voltage, which could result in an initially non-linear graph. To ensure a linear IV characteristic which we required, the circuit should be checked to confirm that only ohmic components, such as function generators with deliberately eliminated impedance, are present and that all connections are secure and free from faults/open ends.

[1] The Sogang University Physics Department. Experimental physics 1 manual. *"LabVIEW 2"*.

[2] The Sogang University Physics Department. Experimental physics 1 manual. *"DMM6500 & LabVIEW"*.