**DECLARATION**

We **Ajay Kumar Raghav*, B.Tech-CS 1st year, 2215000139, Aman Kumar, B.Tech-CS 1st year, 2215000182, Krishanveer Singh, B.Tech-CS 1st year, 2215000927, Kush Pratap Singh, B.Tech-CS 1st year, 2215000970, Piyush Gautam, B.Tech-CS 1st year, 2215001227*** hereby declare that the work presented in this project report entitled **Data Acquisition and Analysis using Lab VIEW** is an authentic record of our work carried out under the supervision of Dr. Mohd. Zuhaib.

Ajay Kumar Raghav*, 2215000139* Aman Kumar*, 2215000182*

*Sign. Sign.*

Krishanveer Singh*, 2215000927* Kush Pratap Singh*, 2215000970*

*Sign. Sign.*

Piyush Gautam, 2215001227

*Sign.*

**CERTIFICATE**

This is to certify that the above statement made by the students is correct to the best of my knowledge and belief.

Date: 15/01/2023

Place: Mathura

Name and Signature with Affiliation of Supervisor



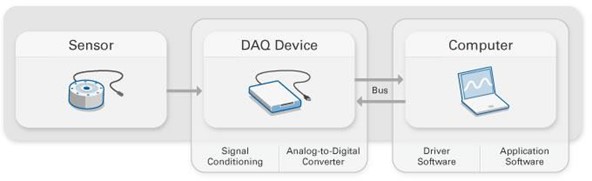
**Contents**

|  |  |
| --- | --- |
| **Certificate & Declaration** | **1-2** |
| **Table of Contents** | **3** |
| 1. **Introduction** | **4-5** |
| 1. **Project Description and Work done** | **6-24** |
| 1. **Geotagged Images** | **25** |
| **4. References** | **26** |

**Introduction**

* 1. ***Data Acquisition***

**Data acquisition (DAQ)** is the process of measuring an electrical or physical phenomenon such as voltage, current, temperature, pressure, or sound with a computer. A DAQ system consists of sensors, DAQ measurement hardware, and a computer with programmable software.



### 1.2 *DAQ Device*

****A data acquisition device or DAQ is a device that converts a physical signal into a digital signal that a computer can read. Data acquisition devices can also include signal conditioning that can filter, linearize, amplify, or attenuate the signal from the sensor. DAQ devices can connect to a computer via USB, Ethernet, or Wi-Fi.

* 1. **Data Analysis-**

**Data analysis** is a process of inspecting, cleansing, transforming, and modelling data to discover useful information, informing conclusions, and support decision-making.

* 1. ***LabVIEW-***

It stands for **Laboratory Virtual Instrumentation Engineering Workbench.** LabVIEW is a graphical programming environment engineers use to develop automated research, validation, and production test systems.

**Virtual Instrumentation**

* 1. ***Introduction***

For many years electronic instruments have been easily identified products. Although they ranged in size and functionality, they all tended to be box-shaped objects with a control panel and a display. Stand-alone electronic instruments are very powerful, expensive and designed to perform one or more specific tasks defined by the vendor. However, the user generally cannot extend or customize them. The knobs and buttons on the instrument, the built-in circuitry, and the functions available to the user, all of these are specific to the nature of the instrument. In addition, special technology and costly components must be developed to build these instruments, making them very expensive and hard to adapt.

Widespread adoption of the PC over the past twenty years has given rise to a new way for scientists and engineers to measure and automate the world around them. One major development resulting from the ubiquity of the PC is the concept of virtual instrumentation. A virtual instrument consists of an industry-standard computer or workstation equipped with of the-shelf application software, cost-effective hardware such as plug-in boards, and driver software, which together perform the functions of traditional instruments. Today virtual instrumentation is coming of age, with engineers and scientists using virtual instruments in literally hundreds of thousands of applications around the globe, resulting in faster application development, higher quality products and lower costs. Virtual instruments represent a fundamental shift from traditional hardware-centred instrumentation systems towards software centred systems that exploit the computing power, productivity, display and connectivity capabilities of popular desktop computers and workstations. Although PC and integrated circuit technologies experienced significant advances in the past two decades, it is the software that makes possible building virtual instruments on this foundation. Engineers and scientists are no longer limited by traditional fixed-function instruments. Now they can build measurement and automation systems that suit exactly their specific needs.

Virtual Instrumentation combines mainstream commercial technologies such as PC, with flexible software and a wide variety of measurement and control hardware. Engineers use virtual instrumentation to bring the power of flexible software and PC technology to test, control and design applications making accurate analog to digital measurements. Engineers and scientists can create user-defined systems that meet their exact application need. Industries with automated processes, such as chemical or manufacturing plants use virtual instrumentation with the goal of improving system productivity, reliability, safety, optimization and stability. Virtual Instrumentation is computer software that a user would employ to develop a computerized test and measurement system for controlling from a computer desktop, an external device on instrument-like panels on a computer screen. It extends to computerized systems for controlling processes based on data collected and processed by a computerized instrumentation system. The front panel control function of the existing system is duplicated through the computer interface. The application ranges from simple laboratory experiments to large automated application.

Virtual Instrumentation as shown in figure below uses highly productive software, modular I/O and commercial platforms. National Instruments LabVIEW, a premier virtual instrumentation graphical development environment, uses symbolic or graphical representations to speed up development. The software symbolically represents functions. Consolidating functions within rapidly graphical blocks further speeds up development.

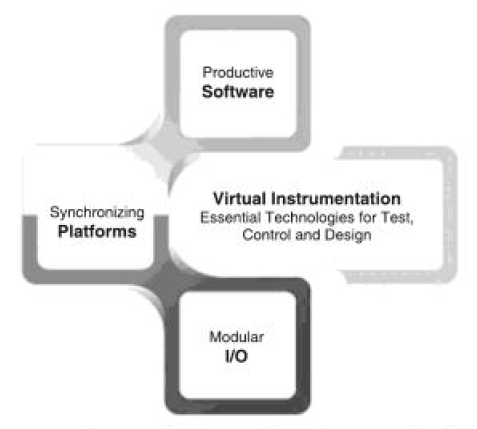


Figure 2.1 Virtual Instrumentation combines Productive software, modular I/O and scalable platforms.

**2.2 *Virtual instrument and Traditional instrument***

Traditional instruments and software- based virtual instruments largely share the same architectural components but radically different philosophies as shown in Figure 2.2. Conventional instruments as compared to a virtual instrumentation can be very large and cumbersome. They also require a lot of power and also have excessive amounts of features that are rarely, if ever used. Most conventional instruments do not have any computational power as compared to virtual instrument. Since the virtual instrument is part of a personal computer configuration, the personal computer’s computational as well as controlling capability can be applied into a test configuration. Virtual instruments are compatible with traditional instruments almost without exception. Virtual instrumentation software typically provides libraries for interfacing with common ordinary instrument buses such as GPIB, serial or Ethernet.

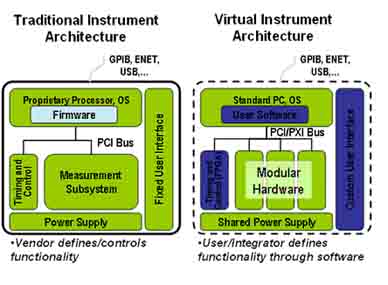


Figure 2.2 Traditional instruments (left) and software based virtual instruments (right).

Except for the specialized components and circuitry, found in traditional instruments, the general architecture of stand – alone instruments is very similar to that of a PC based virtual instrument. Both require one or more microprocessor, communication ports (for example, serial and GPIB), and display capabilities, as well as data acquisition modules. What makes one different from the other is their flexibility and the fact that we can modify and adapt the instrument to our particular needs. A traditional instrument might contain an integrated circuit to perform a particular set of data processing functions; in a virtual instrument, these functions would be performed by a software running on a PC processor. We can extend the set of functions easily, limited only by the power of the software used. By employing virtual instrumentation solutions, we can lower capital cost, system development costs, and system maintenance costs, while improving time to market and the quality of our own products.

**2.3 *Virtual Instruments***

LabVIEW programs are called **Virtual** **Instruments**, or VIs, because their appearance and operation imitate physical instruments, such as oscilloscopes and multimeters. LabVIEW contains a comprehensive set of tools for acquiring, analyzing, displaying, and storing data, as well as tools to help us to troubleshoot our code.

We can use LabVIEW to communicate with hardware such as data acquisition, vision and motion control devices, GPIB, PXI, VXI, RS-232 and RS-485 instruments.

In LabVIEW we build a user interface or front panel with controls and indicators. Controls are knobs, push buttons, dials, and other input devices. Indicators are graphs, LEDs, and other displays. After we built the user interface, we add code using VI’s and structures to control the front panel objects.

The block diagram contains this code. It should be kept in mind that:

**Controls** equal **inputs**, and **indicators** equal **output**.

Each VI contains three main parts:

Front panel: It helps the user to interact with the VI.

Block Diagram: The code that controls the program.

Icon/connector: Means of connecting a VI to the VI’s.

***Front Panel of virtual instrument***

When we create a new VI or selected an existing VI, the Front Panel and the Block Diagram for that specific VI will appear.

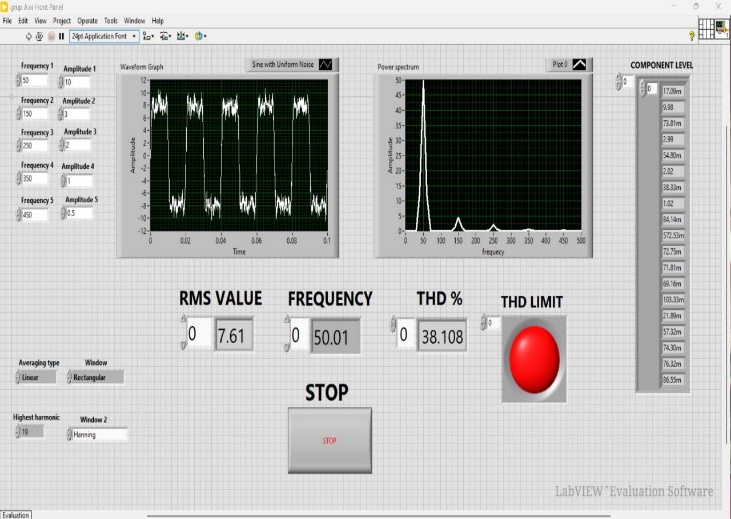
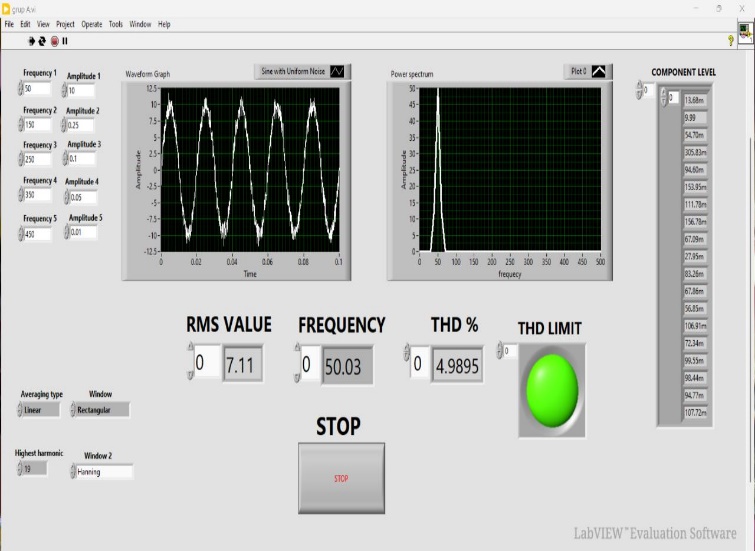
 

Figure 2.3: LabVIEW GUI Window

In LabVIEW, we build a user interface, or front panel, with controls and indicators. Controls are knobs, push buttons, dials, and other input devices. Indicators are graphs, LEDs, and other displays. We build the front Panel with controls and indicators, which are the interactive input and output terminals of the VI, respectively. Controls are knobs, push buttons, dials, and other input devices. Indicators are graphs, LEDs, and other displays. Control simulate instrument input devices and supply data to the block diagram of the VI. Indicators simulate instrument output devices and display data, the block diagram acquires or generates.

**2.4 *Block Diagram***

After building the user interface, we add codes using VI’s and structures to control the front panel objects. The block diagram contains this code. In some ways, the block diagram resembles a flowchart.

After building the front panel, we add code using graphical representations of functions to control the front panel objects. The block diagram contains this graphical source code.

Front panel objects appear as terminals, on the block diagram. Block diagram objects include terminals, sub VIs, functions, constants, structures, and wires, which transfer data among other block diagram objects.

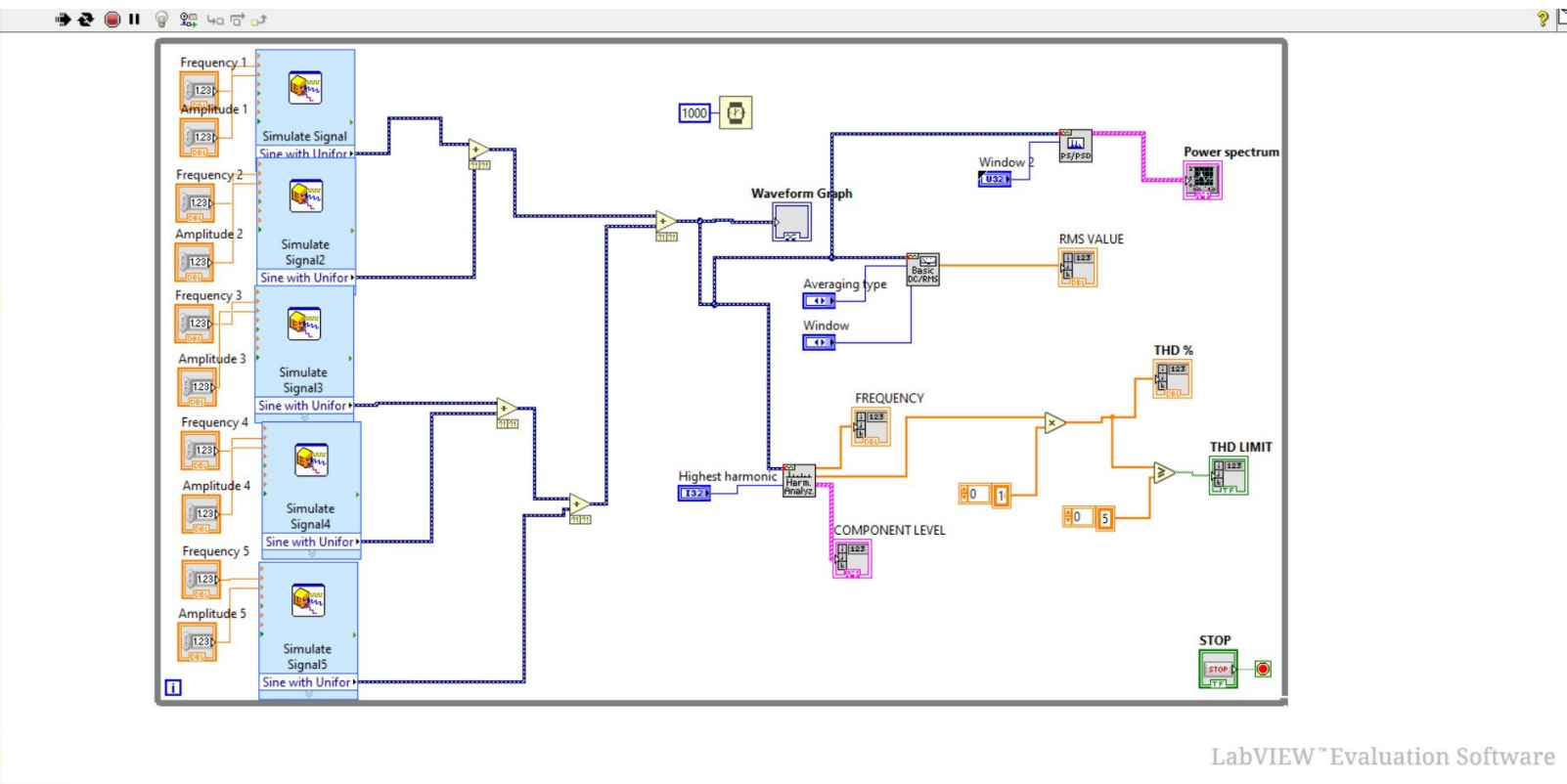


Figure 2.4: LabVIEW Block Diagram

**2.5** ***Controls Palette***

The Controls and Functions palettes contain sub palettes of objects that we can use to create a VI. When we click a sub palette icon, the entire palette changes to the sub palette we selected. To use an object on the palettes, click the object and place it on the front panel or block diagram. The Controls palette is available only on the front panel. The Controls palette contains the controls and indicators that we can use to build the front panel.

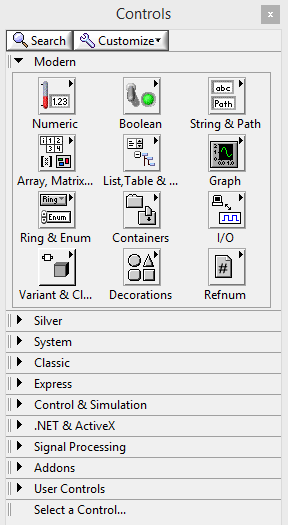


Figure 2.5: Control Palette

The most used Sub Palettes are the Numeric Sub Palette, the Boolean Sub Palette and the String & Path Sub Palette.

**2.6 Numeric Sub Palette**

“Numerical Control” And “Numerical Indicator” are the most used objects in the numeric sub palette.

**Boolean Sub Palette**

This palette has lots of different buttons you may use. OK, Cancel and Stop buttons are useful.

Numeric and Boolean Sub Palette are shown in figure 2.6.

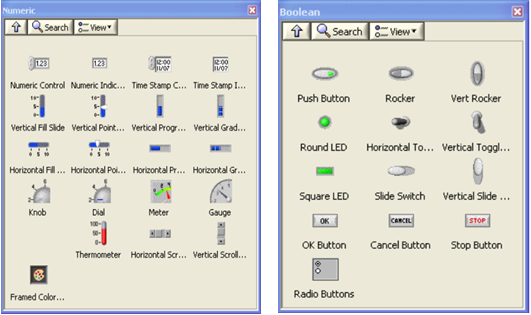


Figure 2.6 Numeric and Boolean Sub Palette

**2.7 Function Palette**

The Functions palette is available only on the block diagram. The Functions palette contains the VI’s and functions we use to build the block diagram.

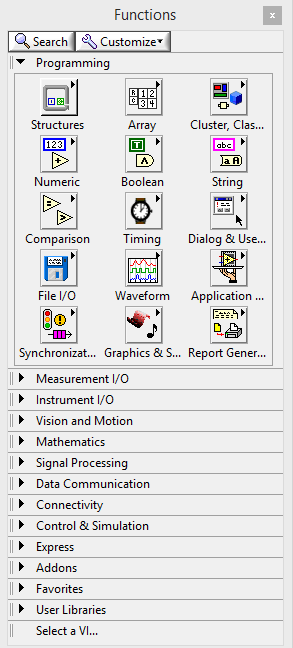


Figure 2.7 Function Palette

**2.8 Tools Palette**

We can create, modify, and debug VI’s using the tools located on the floating Tools palette. The Tools palette is available on both the front panel and the block diagram. A tool is a special operating mode of the mouse cursor. The cursor corresponds to the icon of the tool selected in the Tools palette. Use the tools to operate and modify front panel and block diagram objects.

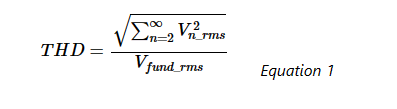


Figure 2.8 Tool palette

**Total Harmonic Distortion**

***3.1 Introduction***

THD is defined as the ratio of the equivalent root mean square (RMS) voltage of all the harmonic frequencies (from the 2nd harmonic on) over the RMS voltage of the fundamental frequency (the fundamental frequency is the main frequency of the signal, i.e., the frequency that you would identify if examining the signal with an oscilloscope). Equation 1 shows the mathematical definition of THD (note that voltage is used in this equation, but current could be used instead):

****

**Where**

* Vn\_rms is the RMS voltage of the nth harmonic
* Vfund\_rms  is the RMS voltage of the fundamental frequency

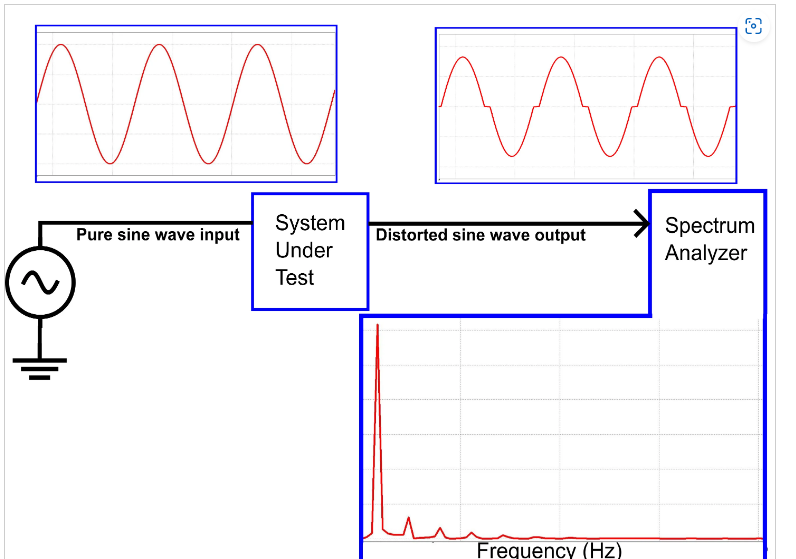
### 3.2 *Measuring Total Harmonic Distortion*

Calculating theoretical THD can be a good exercise, but it can be a lot of work, and in practice, you aren’t going to get an ideal signal (e.g., a perfect square wave) anyway. The outcome of these calculations can only give an approximation for the THD that you might get for a given signal type. In practice, THD must be measured to obtain the RMS value of the fundamental frequency and all of the harmonics. This measurement can be done in a couple of ways.

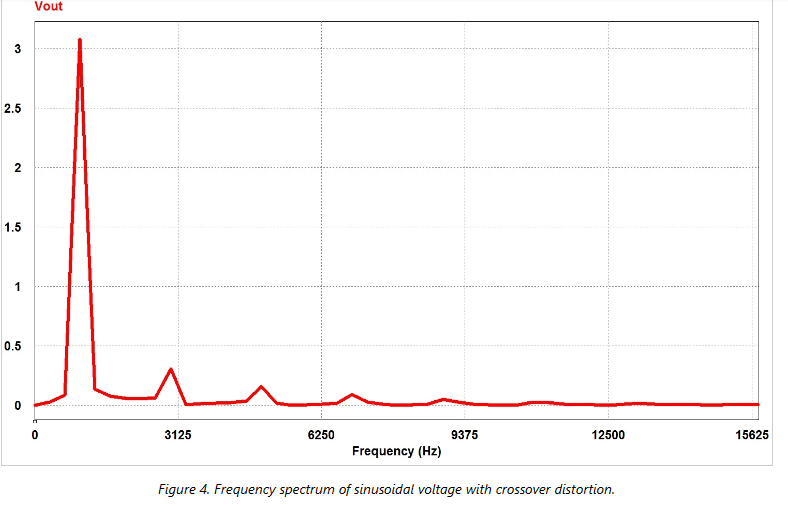
The method for measuring THD is to measure the amplitude of the fundamental frequency and each harmonic and then use those measurements to calculate THD using Equation 1. This measurement can easily be done using a spectrum analyzer or a THD analyzer which will execute Equation 1 automatically. An alternative measurement technique is to capture voltage or current data and then perform a Fourier transform on the data collected.

### 3.3 *Example THD Measurement*

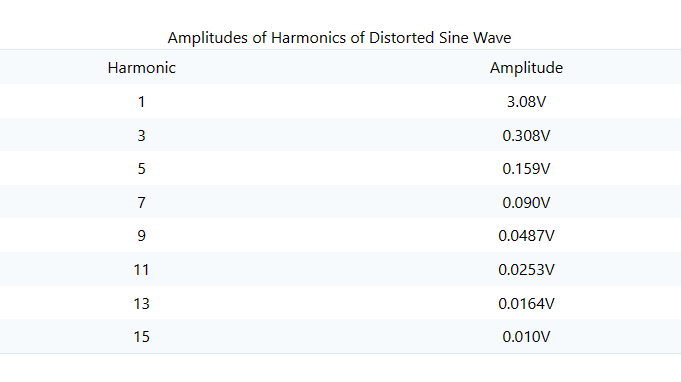
The example block diagram in figure 3 shows a 1 kHz sine wave passing through an amplifier to create a new 1kHz sine wave that has some crossover distortion. This new wave is fed in to a spectrum analyzer which gives a graphical display of the amplitude of a number of the harmonics.



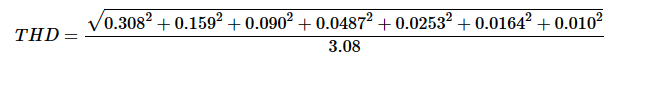
Zooming in on the frequency spectrum of the distorted sine wave output, we can see the amplitudes at several of the harmonic frequencies:



From this frequency spectrum, I manually measured the amplitude of each of the harmonic frequencies and recorded the data in the table below:



The amplitudes of even-numbered harmonics and harmonics above the 15th are nearly 0, so I didn’t include them in my calculation.

The measured amplitudes are plugged in to the THD equation:

This calculation gives a THD of 0.118 or 11.8%.

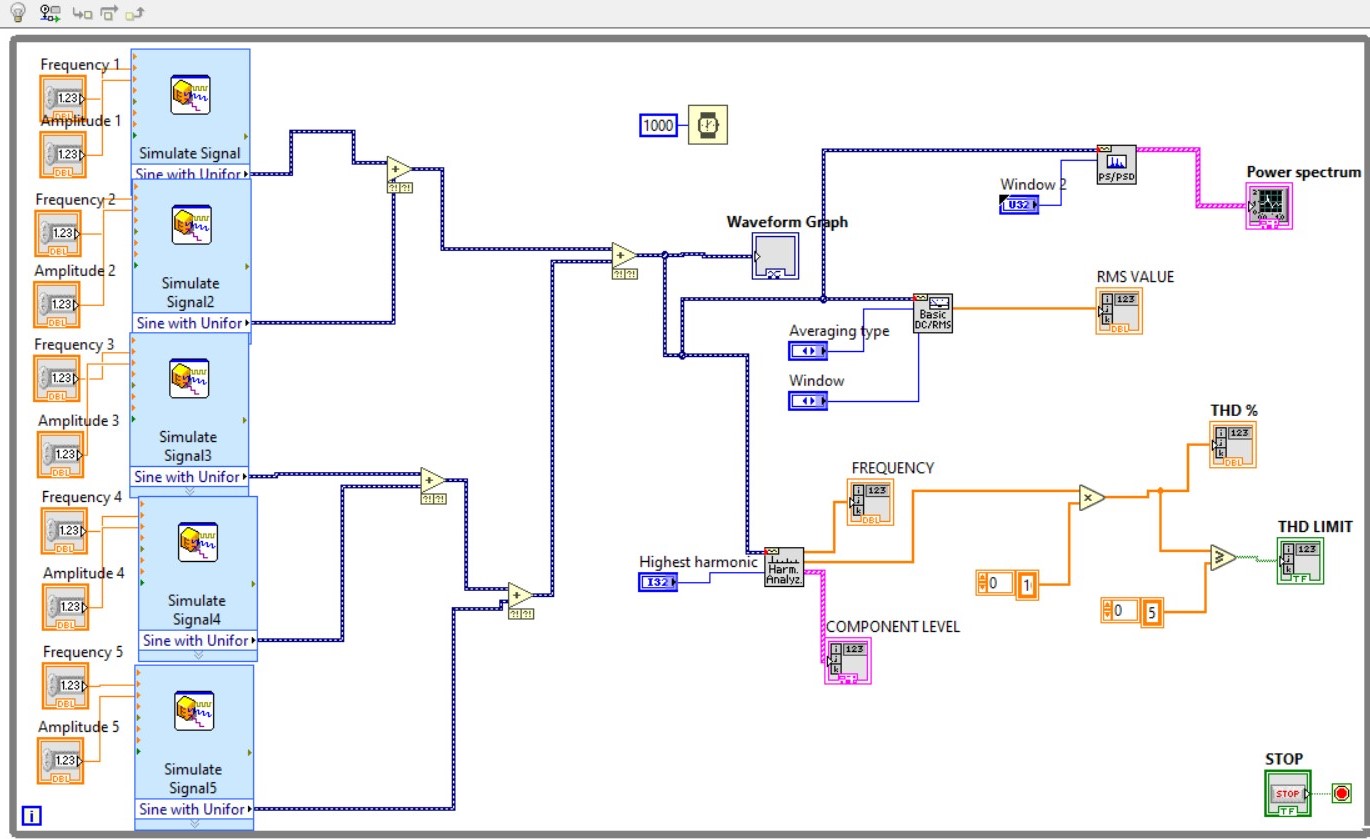
Of course a THD analyzer would automate the process of calculating THD from the amplitudes of the harmonics. Using a THD analyzer for this signal gives a value of 11.9%, which confirms the accuracy of the manual method that I just went through.

### 3.4 *Importance of THD in Systems*

THD is important in several types of systems, including power systems, where a low THD means higher [power factor](https://eepower.com/textbook/vol-i-foundations-power-design/chapter-3-power-ac-systems/power-triangle-and-power-factor), lower peak currents, and higher efficiency; audio systems, where low THD means that the audio signal is a more faithful reproduction of the original recording; and communication systems, where low THD means less interference with other devices and higher transmit power for the signal of interest.

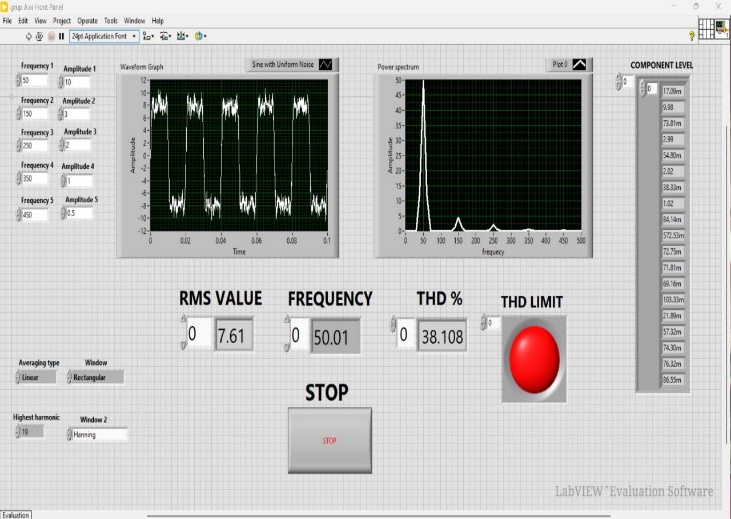
***Work done***

**4.1 *Block Diagram* *Circuit-***

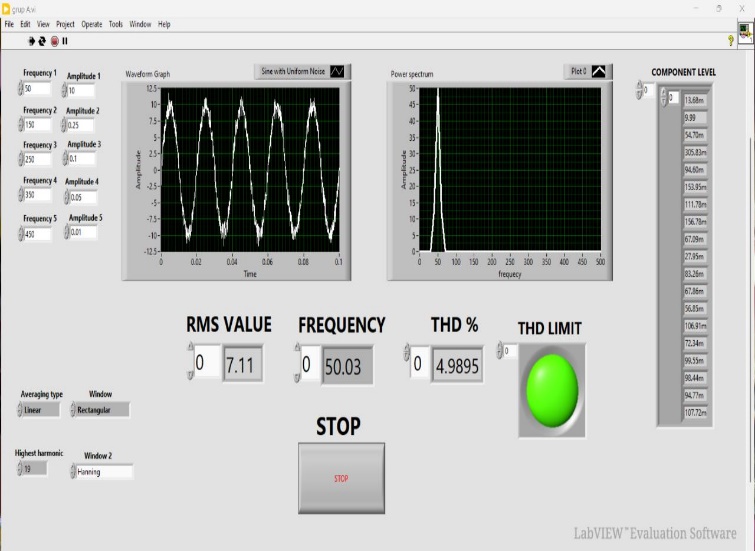
******

**4.2 *Front panel (Running window)-***

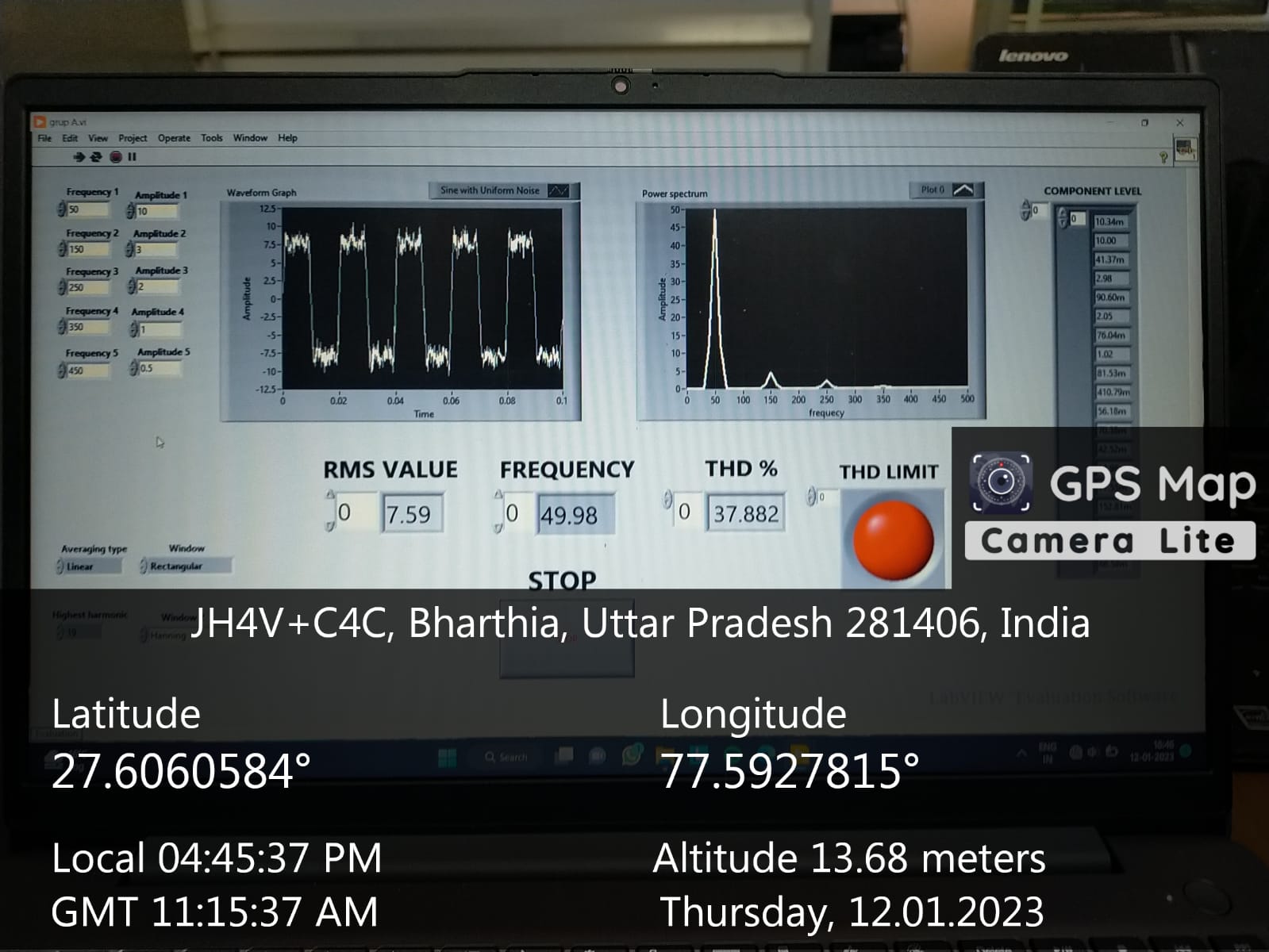
***If THD is greater than 5***

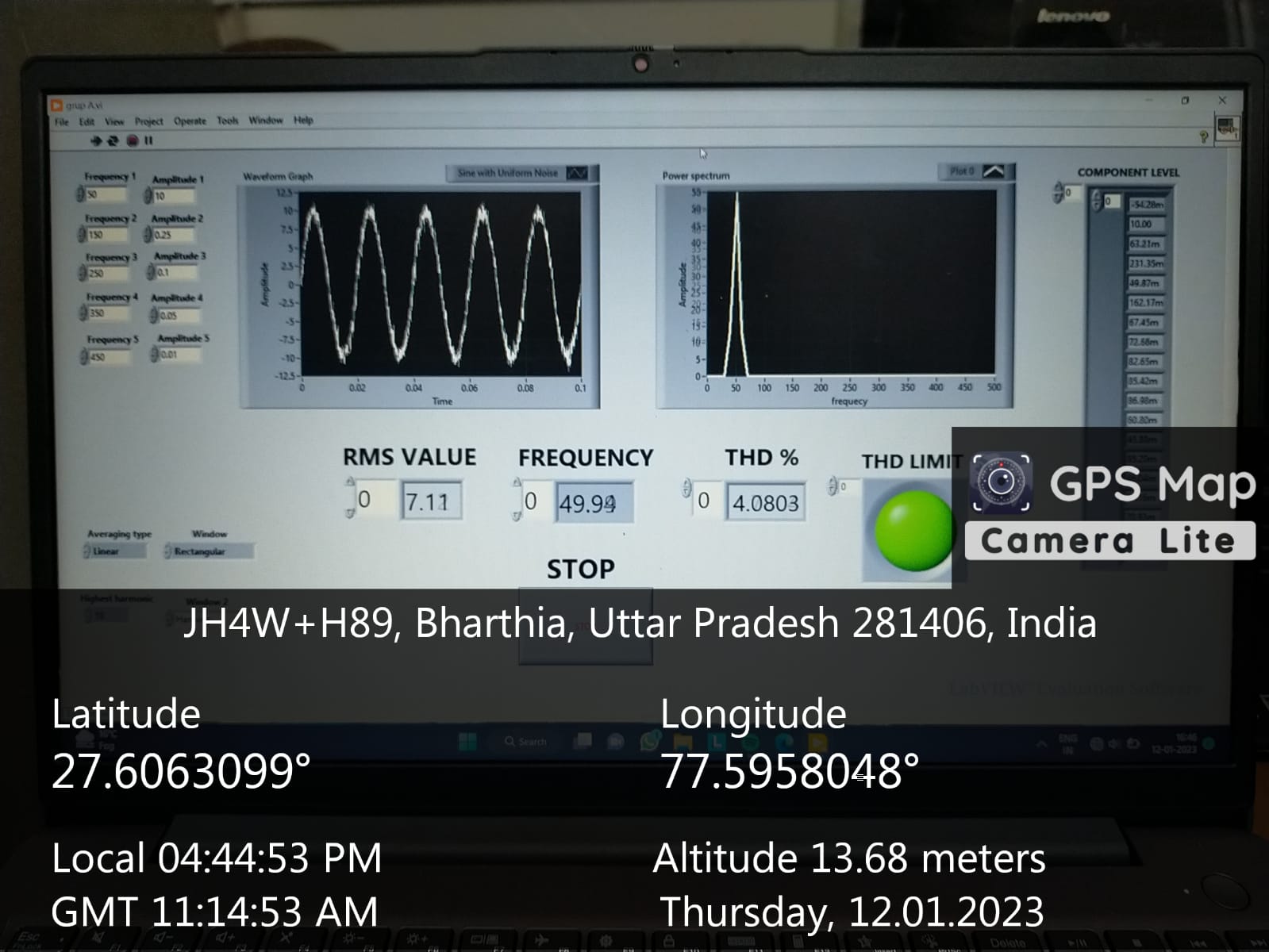


***If THD is less than 5-***



***4.3 Geotag images :***





***References-***

* **The content of this project has been done with the help of our respected supervisor Dr. Mohd. Zuhaib.**
* Some of the information is taken from the internet also.
* [What is LabVIEW? Graphical Programming for Test & Measurement - NI](https://www.ni.com/en-in/shop/labview.html)
* LabVIEW user manual 8.5.1.
* M. Regula, A. Otcenasova, M. Roch, R. Bodnar and M. Repak, "Software for power quality monitoring in model smart grid with using LabView," 2016 ELEKTRO, Strbske Pleso, Slovakia, 2016, pp. 355-358, doi: 10.1109/ELEKTRO.2016.7512096.
* Mishra, Debani & Ray, Papia. (2015). Calculation of power quality factor of supply system using LabVIEW. 10.1049/cp.2015.1622.
* Jovitha Jerome, Virtual Instrumentation using LabVIEW, 2010 PHI, Learning Private Limited, New Delhi, ISBN-978-81-203-4030-5.
* https://youtube.com/playlist?list=PL8oKCBEYiHMnUPowK0B7awUpej22UEc52