By Okan DÜZYEL

An oscilloscope is a laboratory instrument used to graphically display electrical signals and how they change over time. It is essentially a visualizer for electrical activity, similar to how a speaker makes sound waves audible or a monitor makes light waves visible. Think of it as a window into the world of electricity, allowing you to see the invisible flow of electrons and analyze their behavior.

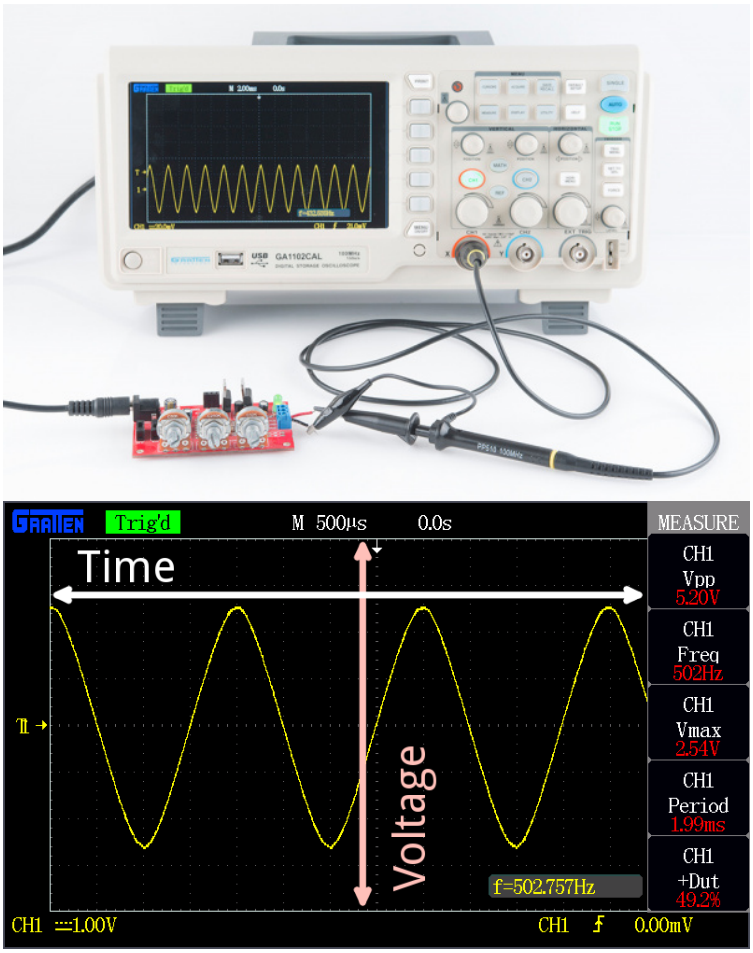


Figure1: Digital oscilloscope and its screen

Oscilloscopes are crucial tools in various fields, including electronics, engineering, physics, and even medicine. They are used to troubleshoot electronic circuits, design new technologies, analyze sound waves, and even monitor heartbeats in an electrocardiogram (ECG).

The oscilloscope screen is the primary interface for visualizing electrical signals. It typically consists of two main parts:

1. Grid: This is a rectangular area covered by a grid of horizontal and vertical lines. The horizontal lines represent time, with each line marking a specific time interval. The vertical lines represent the voltage or amplitude of the signal. The grid provides a reference for measuring the different properties of the waveform being displayed.

2. Trace: This line moves across the screen, representing the changes in the voltage of the signal over time. The brightness or intensity of the trace can also be used to indicate the amplitude of the signal.

INSTRUCTIONS

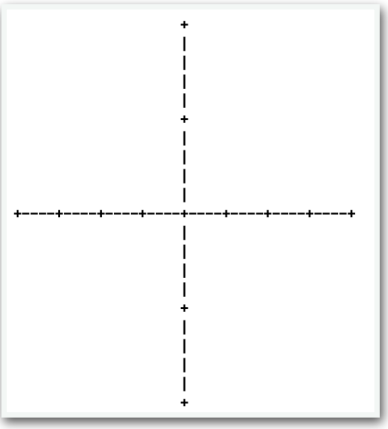
1) Define **#include <stdio.h> and #include <math.h>** libraries that you will use.
2) Ask for two values which are magnitude and frequency, and store them in double variables. (**1-2** have **15 pts**)
3) Define **PI** and **plotArray** globally like right. The plotArray is your screen array which stores char characters (**5 pts**)
4) Define a double function like a blow. I am giving the formula to you. (**5 pts**)

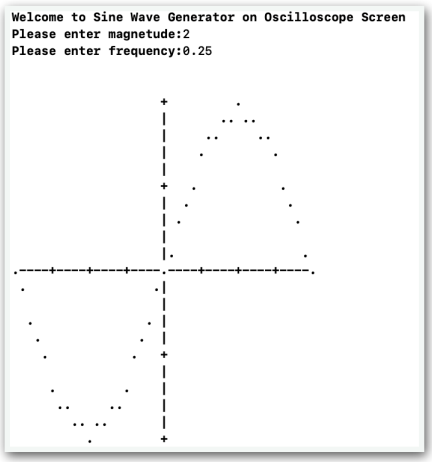
```
Welcome to Sine Wave Generator on Oscilloscope Screen
Please enter magnetude:2
Please enter frequency:0.5
```

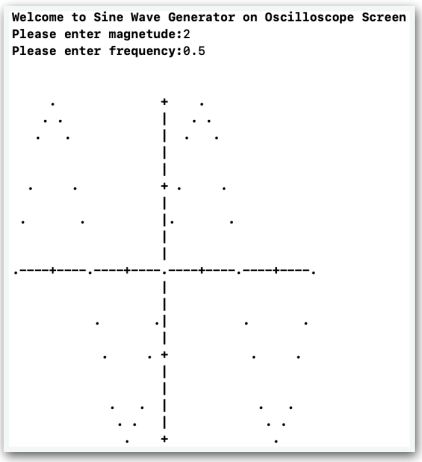
```c
#define PI 3.14159265

// Create a 2D char array for plotting
char plotArray[21][41];
```

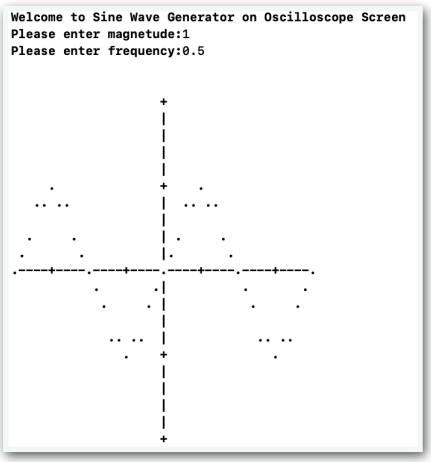
```c
// Function to generate a sine waveform
double generateSinusWave(double time, double Amplitude, double frequency) {
    return Amplitude * sin(0.4 * PI * (time*frequency));
}
```

5) Write a function that fills the **plotArray** with **space characters** (clear the array) **void** clearScreen(**void**). (**10 pts**)
6) Write a function that creates a coordinate system using **"-", "+", and "|"** characters like on the right. Each **5th character** must be plotted as **"+"** on both X and Y axes. (**15 pts**)
7) Write a function that **prints the plotArray** to the screen correctly. **void** printScreen(**void**) (**10 pts**)
8) Write a function that **creates waveform** and writes it into plotArray. You will accept Amplitude and Frequency values and then calculate sinus magnitude for all x coordinate points. Is it logical to use it for a loop? **void** plotWaveform(**double** Amplitude, **double** frequency) (**30 pts**)
9) **Call all functions in the main function** and get the results that are given below. (**10 pts**)









Be careful!!! Sin function returns values only between -1 and 1. However, In our case, 1v refers to 5 characters. Thus, you may use this kind of scale process and type casting. **int** plotValue = (**int**)(5.0 * value);