Electronic Workshop - II
Project-2: Function Generator

Aditya Peketi
Electronics and Communication
International Institute of Information Technology
Hyderabad, India
peketivenkata.v@research.iiit.ac.in,
2024122001

Harsh Kapoor
Electronics and Communication
International Institute of Information Technology
Hyderabad, India
harsh.kapoor@research.iiit.ac.in,
2023112004

***Abstract*—This paper presents the design details of a Wave Generator, created as part of the Electronics Workshop-II course at IIIT-H. The objective of the project was to design and construct a wave generator capable of generating basic waveforms, i.e. Square, Triangle, Sawtooth and Sine waves, with various analog electronic components.**

I. Introduction

The primary objective of this project was to provide students with hands-on experience in the design, development, and testing of analog electronic circuits. By engaging in such a project, students gain practical insight into circuit behavior, component selection, and real-world implementation challenges. The teams for the project were formed randomly, encouraging collaboration among students and diverse perspectives on the problems being tackled. Each team was given the freedom to select their own project idea.

Our team chose to develop a Small-Scale Wave Generator with adjustable amplitude and frequency. This decision was driven by the practical utility of having a compact and portable signal generator that can be used alongside an oscilloscope. Such a device is especially useful for field testing and educational demonstrations, where low power consumption and reliable waveform generation are essential. The project timeline spanned from the second week of March to the end of April, providing ample opportunity for iterative design, simulations prototyping, and testing.

Throughout the course of the project, we underwent multiple iterations of the design, experimenting with a combination of analog and digital circuit techniques. Each iteration provided valuable insights and helped us refine the functionality, stability, and efficiency of the wave generator. A few of these design iterations are discussed in detail within the report to illustrate our developmental process. Relevant resources and references that guided our design decisions are cited in the Bibliography section.

A few of the design iterations we explored during the course of this project include:

1) An op-amp based multivibrator circuit.
2) A square wave generator using an Arduino.
3) An op-amp based relaxation oscillator.

II. Op-Amp Based Multivibrator Circuit

A. Oscillator Circuit

To generate a basic waveform such as a square wave, a multivibrator circuit is essential. One of the simplest and most effective ways to achieve this is through the use of an operational amplifier (op-amp) configured as an astable multivibrator. The operational amplifier, a highly versatile electronic component, finds applications in a wide range of analog circuits including amplifiers, filters, and signal conditioning modules. In the context of this project, we utilized the op-amp in an astable multivibrator configuration to produce a continuous square wave output. This configuration does not require any external triggering and continuously oscillates between two voltage levels, making it ideal for wave generation applications. The frequency and duty cycle of the square wave can be easily adjusted through passive components, providing a flexible and low-power solution for signal generation.

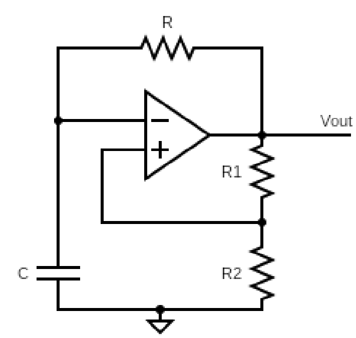


Fig. 1. Circuit Diagram of Multivibrator Circuit

The resistances R_1 and R_2 act as a Schmitt trigger to generate hysteresis. As this resistive network is connected between the amplifiers output and non-inverting (+) input, when Vout is saturated at the positive supply rail, a positive voltage is applied to the op-amps non-inverting input. Likewise, when Vout is saturated to the negative supply rail, a negative voltage is applied to the op-amps non-inverting input. The feedback fraction β is given by -

$$\beta = \frac{R_2}{R_1 + R_2}$$

$$V_{out} = V_{sat}$$

$$V_{ref} = V_{out}\frac{R_2}{R_1 + R_2} = \beta V_{sat}$$

$$\text{Therefore: } \frac{+V_{ref} = +\beta V_{sat}}{-V_{ref} = -\beta V_{sat}}$$

Where $+V_{sat}$ is the positive op-amp DC saturation voltage and $-V_{sat}$ is the negative op-amp DC saturation voltage.

So if Vin exceeds $+V_{ref}$, the op-amp switches state and the output voltage drops to its negative DC saturation voltage. Likewise when the input voltage falls below $-V_{ref}$, the op-amp switches state once again and the output voltage will switch from the negative saturation voltage back to the positive DC saturation voltage. The amount of built-in hysteresis as it switches between the two saturation voltages is defined by the difference between the two trigger reference voltages as: $V_{Hysteresis}$ = $+V_{ref}$ – $(-V_{ref})$.

The period of the output waveform is determined by the RC time constant of the two timing components and the feedback ratio established by the R_1, R_2 voltage divider network which sets the reference voltage level. If the positive and negative values of the amplifiers saturation voltage have the same magnitude, then t1 = t2 and the expression to give the period of oscillation becomes:

$$T = 2RC \times \ln\left(\frac{1+\beta}{1-\beta}\right) \text{ and therefore } f = \frac{1}{T}$$

Now if we modify the value of R we can get different frequencies, and to get different amplitudes we can use an amplifier circuit. All that is left is to generate the other signals like triangular wave and a sinusoidal wave.

B. Integrator Circuit

For the triangular wave we create an integrator circuit, and feed it the square wave as the input. The square wave when integrated over the positive cycle, gives the falling side of the triangular wave, while the negative cycle gives the rising side.

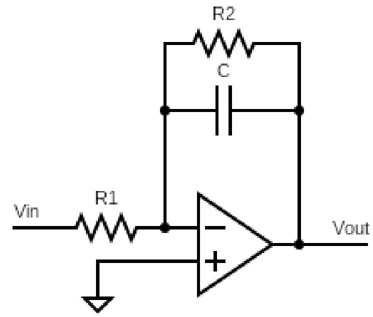


Fig. 2. Integrator Circuit

We know that from the capacitor equations, that -

$$V_c = \frac{Q}{C}$$

Where, Q is the charge on the capacitor plates and C is the capacitance.

$$V_c = V_x - V_{out} = 0 - V_{out}$$

$$-\frac{\mathrm{d}V_{out}}{\mathrm{d}t} = \frac{\mathrm{d}Q}{\mathrm{d}Ct} = \frac{1}{C}\frac{\mathrm{d}Q}{\mathrm{d}t}$$

Where dQ/dt is the current at the node, which is the input current, i.e. -

$$I_{in} = \frac{V_{in} - 0}{R_{in}} = \frac{V_{in}}{R_{in}}$$

$$I_{in} = I_f = \frac{V_{in}}{R_{in}} = \frac{dV_{out}C}{dt}$$

$$\int_0^t dV_{out} = -\frac{1}{R_{in}C}\int_0^t V_{in}dt$$

Therefore, say that from 0 to 1 second the square wave is at level 1 and from 1 to 2 second the square wave is at level -1, the integral will be $V_{out} = -\frac{t}{R_{in}C}$ which is the equation of lines and it will be shifted for the given time.

To generate a sinusoidal waveform from a square wave, we leverage the process of successive integration. This method is grounded in Fourier series analysis, where each integration step smoothens the waveform and attenuates higher-order harmonics more aggressively.

A square wave contains a series of odd harmonics with amplitudes that decay as $\frac{1}{n}$, where n is the harmonic number. Its Fourier series representation is given by:

$$x_s(t) \approx \sum_{n=0}^{\infty} \frac{1}{2n+1}\cos\left(2\pi(2n+1)t\right)$$

Integrating the square wave results in a **triangle wave**, where the harmonic amplitudes now decay as $\frac{1}{(2n+1)^2}$:

$$x_{\Delta}(t) \approx \sum_{n=0}^{\infty} \frac{1}{(2n+1)^2}\sin\left(2\pi(2n+1)t\right)$$

Further integration of the triangle wave yields a **parabolic wave**, whose harmonics decay even faster, following a $\frac{1}{(2n+1)^3}$ relationship:

$$x_p(t) \approx \sum_{n=0}^{\infty} \frac{-1}{(2n+1)^3} \cos\left(2\pi(2n+1)t\right)$$

Because of the cubic decay in the harmonic amplitudes, the parabolic waveform closely approximates a pure sine wave. The deviation from a true sinusoid is minimal and often indistinguishable in practical signal generation applications.

This approach allows for efficient analog generation of sinusoidal waveforms using simple integration stages and minimal filtering, making it well-suited for low-power, portable signal generators.

C. Final Circuit and Function Select

All this added with an amplifying circuit, we can get amplitude modulation and frequency modulation. The final circuit simulated was as follows -

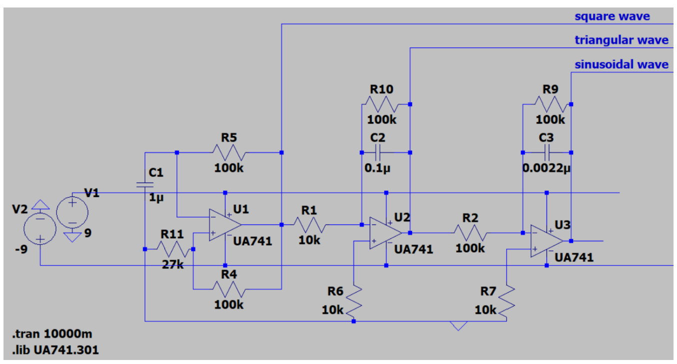


Fig. 3. Multi-Vibrator Based Circuit

Please note that due to the limitations of LT SPICE, we were unable to model potentiometers and are here seen as fixed resistances, also, the way to select the signal going to the amplifier circuit, using switches was not implementable in LT SPICE, but a circuit of the same is given below -

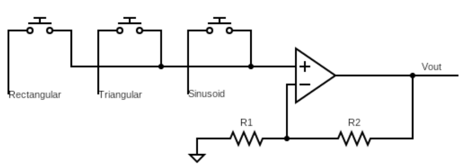


Fig. 4. Function Select

With this addition the circuit is complete. There were certain drawbacks in this circuit like having two integrators back to back decreased the amplitude of the sine wave and also introduced unwanted distortions in the signal. Also the slew rate of the oscillator was very poor and we needed a topology with better slew rate or a better op amp. Due to these reasons we explored other options.

III. Arduino Based Square Wave Generator

A. Oscillator Circuit

The oscillator circuit is the core of the square wave generator, responsible for producing a stable and periodic waveform. In this project, the oscillator functionality is implemented using the Arduino microcontroller, which generates a digital square wave by toggling its digital output pins at a specified frequency.

Unlike traditional analog oscillator circuits, this design leverages the digital processing capabilities of the Arduino to generate an 8-bit digital square wave, which is then converted into an analog signal using an external 8-bit Digital-to-Analog Converter (DAC) (Datasheet given in the references section).

The Arduino is programmed to output an 8-bit digital value on eight GPIO pins simultaneously. These 8-bit values represent discrete voltage levels of a square wave, toggling between two binary states—typically 0x00 (all LOW) and 0xFF (all HIGH)—to form a square waveform in digital format.

This parallel digital output is fed into an external DAC (the DAC0808), which converts the 8-bit digital word into a corresponding analog voltage. The DAC output is a two-level analog signal that accurately follows the transitions of the digital square wave, effectively producing an analog square wave at the desired frequency.

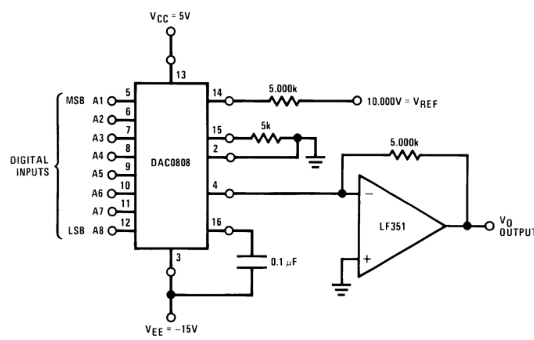


Fig. 5. Circuit for DAC IC

The frequency and duty cycle of the generated waveform can be controlled in software by adjusting the timing of digital updates in the Arduino code. This approach provides flexibility and programmability, allowing users to generate square waves with varying characteristics without changing any hardware components.

This hybrid digital-analog technique offers the precision of digital control with the smooth voltage output of analog signals, making it suitable for applications in waveform generation, signal synthesis, and communication system testing.

Once the square wave was generated, the rest of the circuit followed the similar process as mentioned above using two integrators and amplifiers. Like the previous circuit this also suffered from the drawbacks of having 2 integrators cascaded, and the poor slew rate of the op-amps being used. This motivated us to find a new topology for the circuit, which is the one we used.

IV. Relaxation Oscillator based Topology

Before explaining the sub circuits of the final topology, first lets look at the overall circuit overview. The oscillator generates both rectangular and triangular waves and the triangular wave is integrated and passed through a low pass filter to get a sinusoidal wave. These waves are passed through separate preamplifier before passing them to the function select which in turn passes the desired signal to an amplitude adjustable amplifier. The Oscillator is frequency tunable and hence we get a frequency and amplitude tunable circuit. Please refer to the flowchart below to get an idea of the circuit topology.

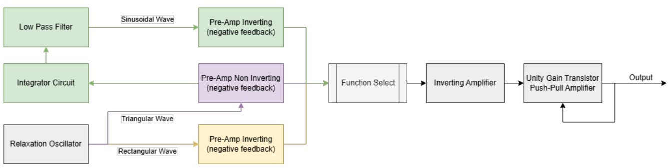


Fig. 6. Circuit Overview

A. Oscillator Circuit

The circuit is a relaxation oscillator using an op-amp with a capacitor C, resistor R, and feedback resistors R_1 and R_2.

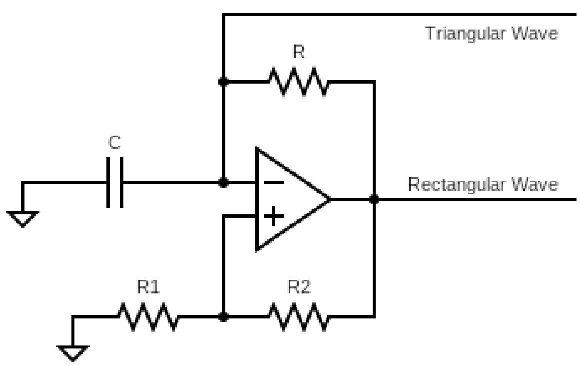


Fig. 7. Relaxation Oscillator

Assume at $t = 0$, $V_C = 0$, and $V_{\text{out}} = 12\,V$.

1) Capacitor starts charging:
$$V_C = 12\left(1 - e^{-t/RC}\right)$$
2) When $V_C > 6$, the output flips: $V_{\text{out}} = -12\,V$
3) Capacitor discharges:
$$V_C = V_{\text{init}} + (V_f - V_{\text{init}})\left(1 - e^{-t/RC}\right)$$
$$= 6 + (-12 - 6)\left(1 - e^{-t/RC}\right) = 18e^{-t/RC} - 12\,V$$
4) When $V_C < -6\,V$, the output flips back: $V_{\text{out}} = 12\,V$, and the threshold is $V_t = 6\,V$

$$V_t = \pm 12 \cdot \frac{R_1}{R_1 + R_2}$$

We want V_t to be small ($\approx\ 1\,V$) to achieve a nearly triangular waveform at the capacitor (since an exponential curve approximates a triangle in this regime).

$$\text{Choose: } R_1 = 10\,k\Omega, \quad R_2 = 100\,k\Omega$$

Let $R =$ Potentiometer (PoT) 1 $\quad\Rightarrow$ used for frequency selection

- Feedback divider ratio:
$$\beta = \frac{R_2}{R_1 + R_2}$$
- Frequency of oscillation:
$$f = \frac{1}{2 \cdot C \cdot R \cdot \ln\left(\frac{1+\beta}{1-\beta}\right)}$$
- Behavior:
  - $R \uparrow \Rightarrow f \downarrow$
  - $C \uparrow \Rightarrow f \downarrow$
- Choose: $R =$ PoT (100 k)
- $C = 1\,\mu$F initially $\Rightarrow$ lower capacitance gives higher frequency

B. Pre-Amplifier Circuits

This section will deal with the pre-amplifiers of the various functions.

*1) **Rectangular Wave Pre-Amplifier**:* The Rectangular Wave generated is a square wave oscillating between -10 V to +10 V. However, this detail is not immediately necessary. This information will be important when performing amplitude scaling.

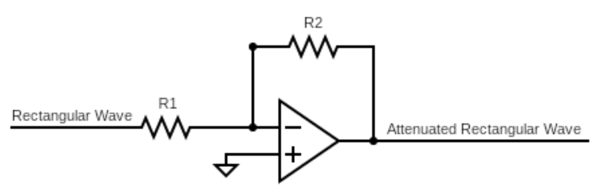


Fig. 8. Rectangular Wave Pre Amplifier

The op-amp stage is configured as an inverting amplifier to attenuate the square wave.

Gain:
$$G = -\frac{R_2}{R_1 + R_2}$$
Given:
$$R_1 = 10\text{ k}\Omega, \quad R_2 = 1\text{ k}\Omega \Rightarrow G \approx -0.1$$

This results in an attenuated square wave output, scaled down appropriately.

*2) **Triangular Wave Pre-Amplifier**:* This is used as a non-inverting amplifier to pick up the triangle wave. The non-inverting configuration is chosen due to its very high input impedance, which is critical because the charge on the capacitor is very small. Any significant current draw could disrupt the oscillator.

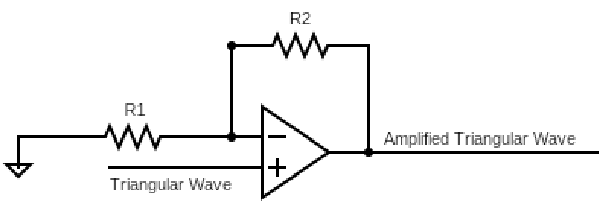


Fig. 9. Triangular Wave Pre Amplifier

Gain:

$$G = 1 + \frac{R_2}{R_1}$$

Note: Now we can scale this waveform properly due to appropriate amplifier design.

Given:

$$R_1 = 4.7\ \text{k}\Omega, \quad R_2 = 6\ \text{k}\Omega \Rightarrow G \approx 2.27$$

This results in an amplified triangular wave output, scaled up appropriately.

*3) **Sinusoidal Wave Pre-Amplifier**:* This is used as a inverting amplifier to scale up the sinusoidal wave. The values are chosen so that the wave gets scaled down by a factor of 2/3.

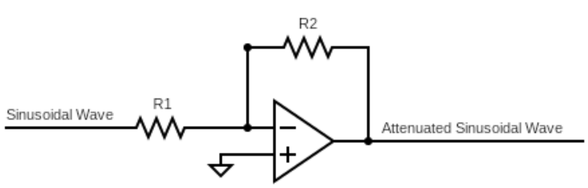


Fig. 10. Sinusoidal Wave Pre Amplifier

Gain:

$$G = -\frac{R_2}{R_1 + R_2}$$

Given:

$$R_1 = 5\ \text{k}\Omega, \quad R_2 = 10\ \text{k}\Omega \Rightarrow G \approx -0.67$$

This results in an attenuated Sinusoidal wave output, scaled down appropriately.

C. Integrator Circuit

This stage uses an op-amp configured as an **integrating amplifier**, which is frequency dependent.

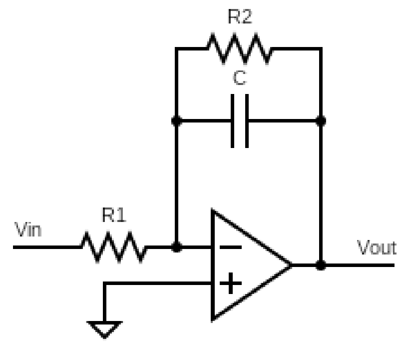


Fig. 11. Integrator circuit

Gain:

$$G = -\frac{1}{2\pi RCj}$$

This expression shows the gain depends on frequency ($j\omega$) and is inversely proportional to it.

- This configuration produces a sine wave output, but **only at high frequencies**.
- To make this stage work effectively, the capacitance was adjusted to $0.54\ \mu$F at Oscillator Stage.
- Since the gain is frequency-dependent, the capacitor value C in the integrator can be tuned based on frequency requirements.

The working of this block is already explained in the previous stage as why it gives a sinusoidal output from a triangular input.

D. Filtering Circuit

The topology uses 2 filters one after the Integrator Block and one after Preamplifier block of the sinusoidal wave. They are required to get a wave which is closer to sin wave.

*1) **RC Low Pass Filter**:* This filter is applied after the Integrator block to remove the unwanted Sinusoidal components of the Integrated results as mentioned earlier.

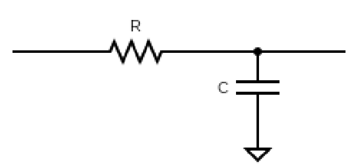


Fig. 12. RC Low Pass Filter

Here,

$$R = 10k\Omega \text{ and } C = 270pF$$

Therefore,

$$f = \frac{1}{2\pi RC} = \frac{1}{2\pi \times 10 \times 10^3 \times 270 \times 10^{-12}} Hz \approx 60kHz$$

*2) **RC High Pass Filter**:* This filter is applied after the Preamplifier block and is used to get rid of any unwanted noise signals which might have crept in.

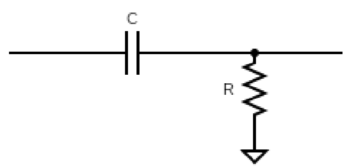


Fig. 13. RC High Pass Filter

Here,

$$R = 100k\Omega \text{ and } C = 10\mu F$$

Therefore,

$$f = \frac{1}{2\pi RC} = \frac{1}{2\pi \times 100 \times 10^3 \times 10 \times 10^{-6}} Hz \approx 10Hz$$

These two filters ensure that the sin wave generated has minimal effect of the noise and other harmonics while ensuring that the wave generated doesn't get attenuated themselves.

E. Function Select

Initially, a **4×1 multiplexer** was planned to be used for function selection. Each function could be chosen by setting the appropriate control lines via pushbuttons.

However, after discussions with our TA, **Mr. Chetanya**, a new approach was adopted: using **pushbuttons alone** for function selection, directly interfaced with a **common bus** that connects to the succeeding amplifier stage.

Advantages of Pushbutton-Based Function Select:

1) **Simplified Control:** Selecting a function only requires pressing a single pushbutton, making the user interface more intuitive and reducing the need for managing select lines as in a multiplexer.
2) **Power Efficiency:** Eliminating the additional multiplexer IC reduces the overall power consumption of the system — an important consideration in embedded and analog designs.

Implementation Details:

- The pushbutton schematic remains the same as previously designed.
- The only modification lies in the **succeeding amplifier stage**, which now interfaces with the function outputs via a shared bus instead of receiving a single output from a multiplexer.

This change not only simplifies the circuit but also aligns with practical design principles for compact and low-power systems.

F. Amplitude Select Amplifier Circuit

The circuit represents the final **amplifier stage**, used to drive high-current loads such as speakers or analog function modules. It is built using op-amps (UA741) and complementary BJTs (TIP31C and TIP32C), forming a robust class AB push-pull configuration.

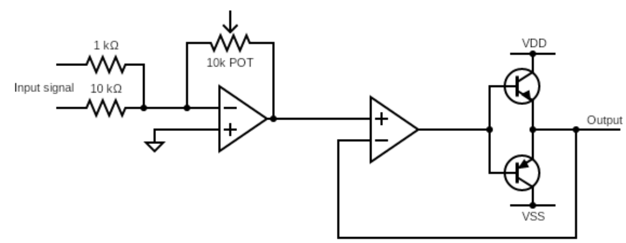


Fig. 14. Amplitude Select Circuit

- **First Op Amp**: Configured as a *non-inverting amplifier*, this op-amp conditions and amplifies the input signal. The gain is controlled by the resistor network (`1k, 10k`) and a potentiometer (`10k POT`).
- **Second Op Amp**: Acts as a *driver stage*, providing sufficient current to control the base terminals of the output BJTs, while buffering the previous stage.
- **Q1 (TIP31C, NPN) and Q2 (TIP32C, PNP)**: Form a *complementary push-pull pair* for class AB amplification. This allows the circuit to deliver both positive and negative halves of the waveform efficiently.

This amplifier stage serves as the final output driver in the analog signal chain and is particularly useful in applications requiring clean and strong signal delivery to the load.

The gain of the amplifier is given by:

$$G = -\frac{\text{POT2}}{R_q} \quad \text{(Switch Position 1)}$$

$$G = -\frac{\text{POT2}}{R_{10}} \quad \text{(Switch Position 2)}$$

Where:

- POT2 is the variable resistor in the feedback path.
- R_q and R_{10} are two different input resistors selectable via a switch.

This switchable design enables selecting a gain of either:

- **1X (unity gain)**, or
- **10X gain**, depending on the resistor path chosen.

G. Complete Circuit and LT-SPICE Simulations

The complete circuit was made and simulated in LT-SPICE. The circuit and the output waveforms are given in the following images. Since the values were fixed, the frequency was - $1kHz$.

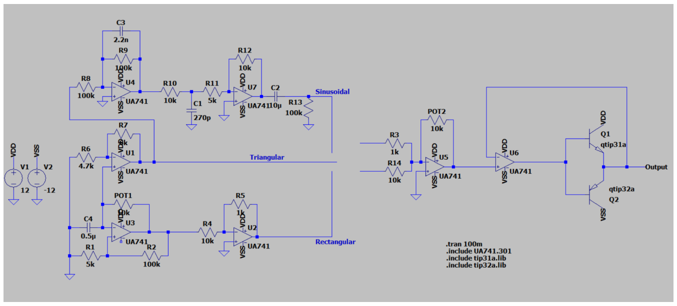


Fig. 15. LT SPICE Circuit

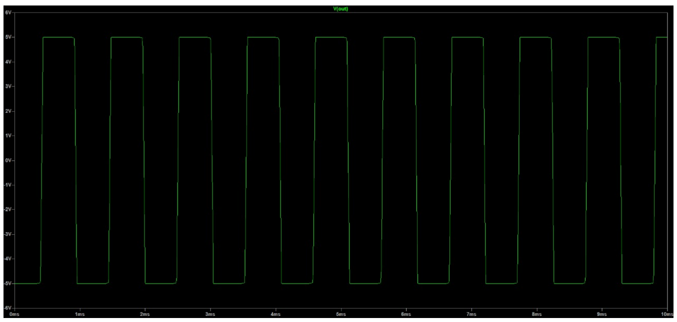


Fig. 16. Rectangular Wave

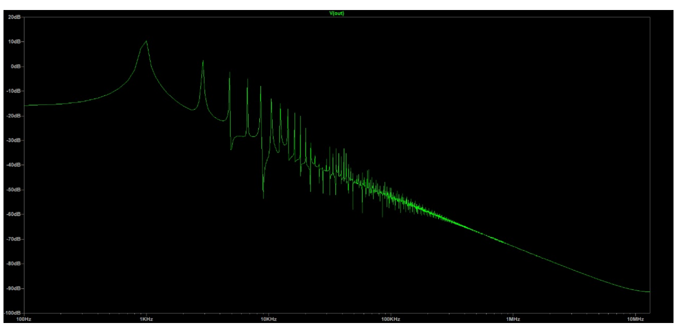


Fig. 17. FFT of the Rectangular Wave

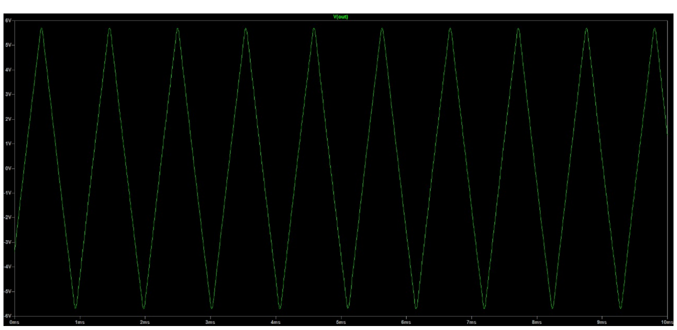


Fig. 18. Triangular Wave

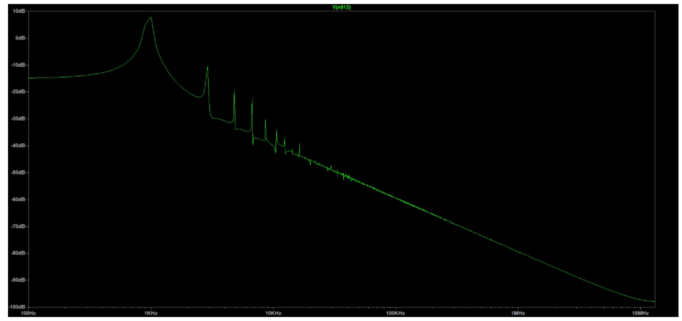


Fig. 19. FFT of Triangular Wave

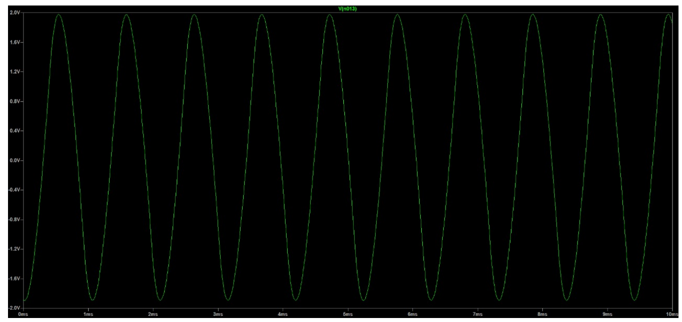


Fig. 20. Sinusoidal Wave

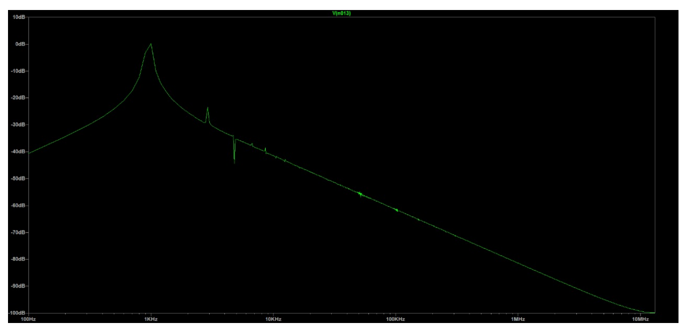
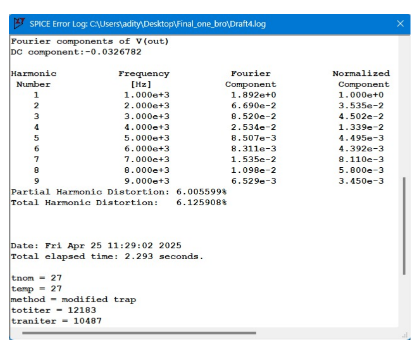


Fig. 21. FFT of Sinusoidal Wave



SPICE Error Log: C:\Users\adity\Desktop\Final_one_bro\Draft4.log

Fourier components of V(out)
DC component:-0.0326782

Harmonic Number	Frequency [Hz]	Fourier Component	Normalized Component
1	1.000e+3	1.892e+0	1.000e+0
2	2.000e+3	6.690e-2	3.535e-2
3	3.000e+3	8.520e-2	4.502e-2
4	4.000e+3	2.534e-2	1.339e-2
5	5.000e+3	8.507e-3	4.495e-3
6	6.000e+3	8.311e-3	4.392e-3
7	7.000e+3	1.535e-2	8.110e-3
8	8.000e+3	1.098e-2	5.800e-3
9	9.000e+3	6.529e-3	3.450e-3

Partial Harmonic Distortion: 6.005599%
Total Harmonic Distortion: 6.125908%

Date: Fri Apr 25 11:29:02 2025
Total elapsed time: 2.293 seconds.

tnom = 27
temp = 27
method = modified trap
totiter = 12183
traniter = 10487

Fig. 22. THD of Sinusoidal wave at 1kHz

H. Oscilloscope Waveforms

*1) **Rectangular Wave:*** These are the plots for square wave

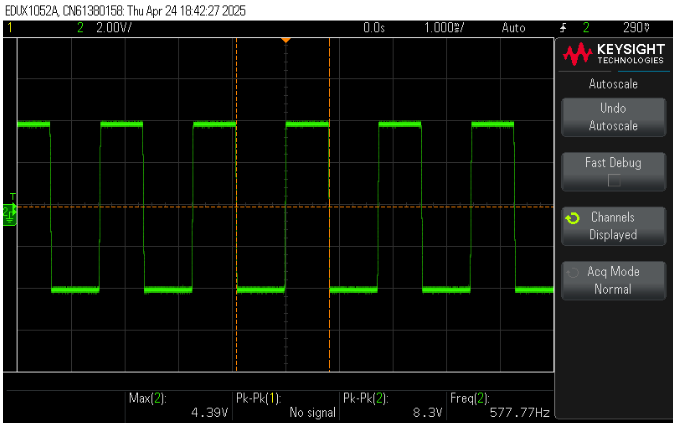


Fig. 23. Square Wave $8.3V_{pp}$ and 580Hz

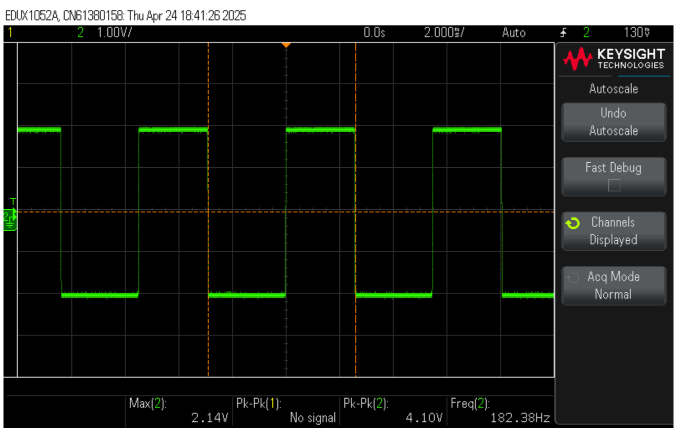


Fig. 24. Square Wave $4.1V_{pp}$ and 180Hz

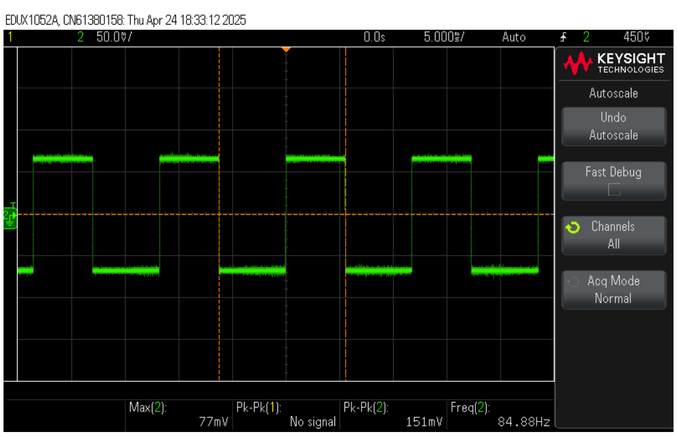


Fig. 25. Square Wave $150mV_{pp}$ and 85Hz

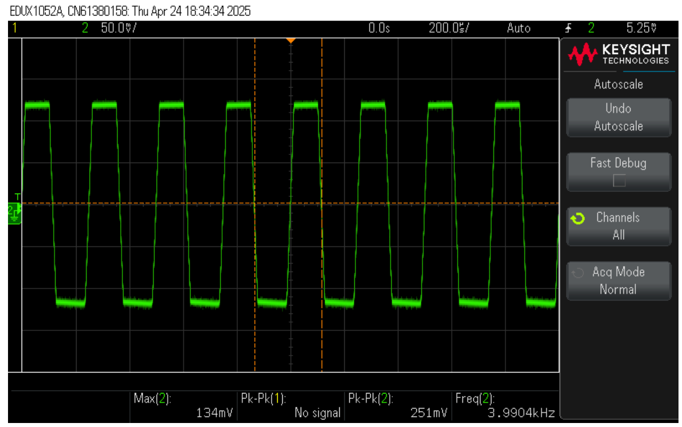


Fig. 26. Square Wave $250mV_{pp}$ and 4kHz

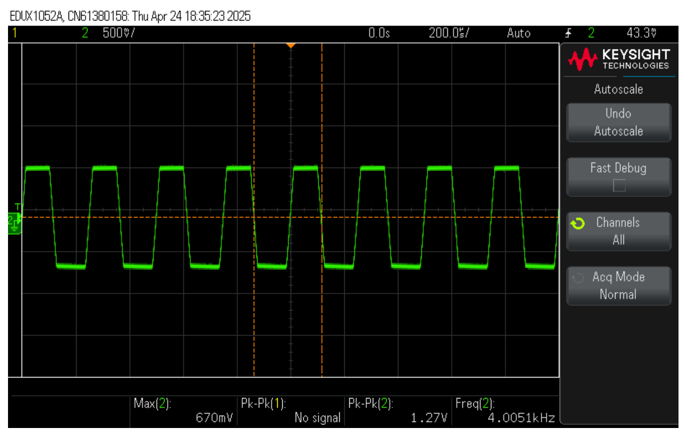


Fig. 27. Square Wave $1.3V_{pp}$ and 4kHz

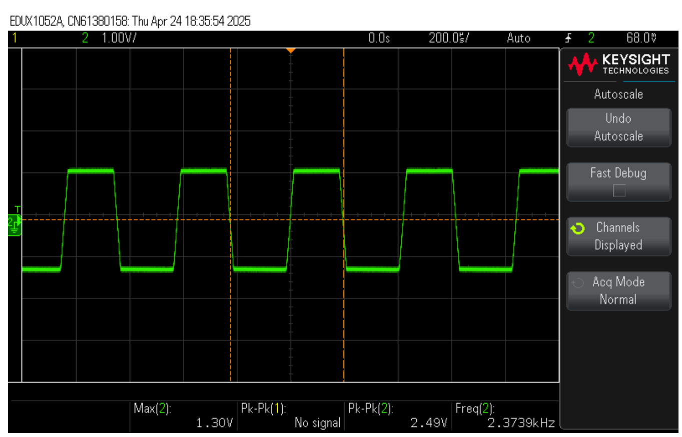


Fig. 28. Square Wave $2.5V_{pp}$ and 2.4kHz

*2) **Triangular Wave Plots**:* These are the plots for trig wave

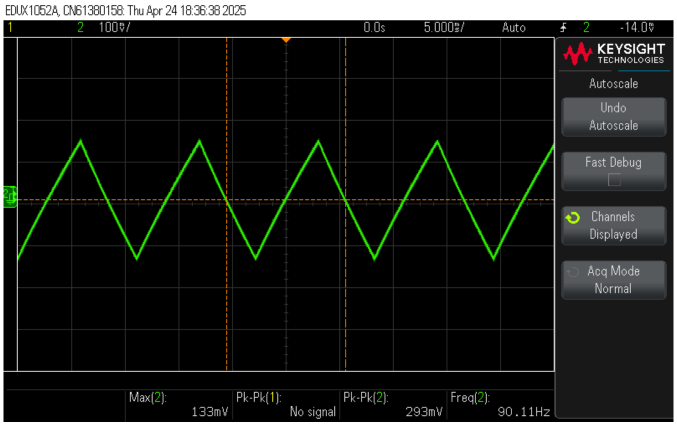


Fig. 29. Triangular Wave $300mV_{pp}$ and 90Hz

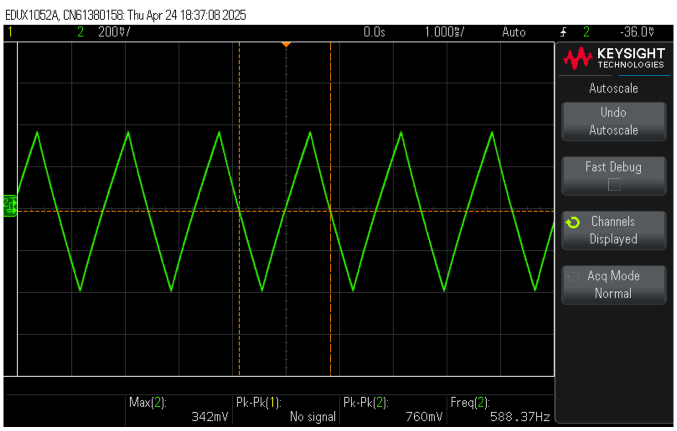


Fig. 30. Triangular Wave $750mV_{pp}$ and 600Hz

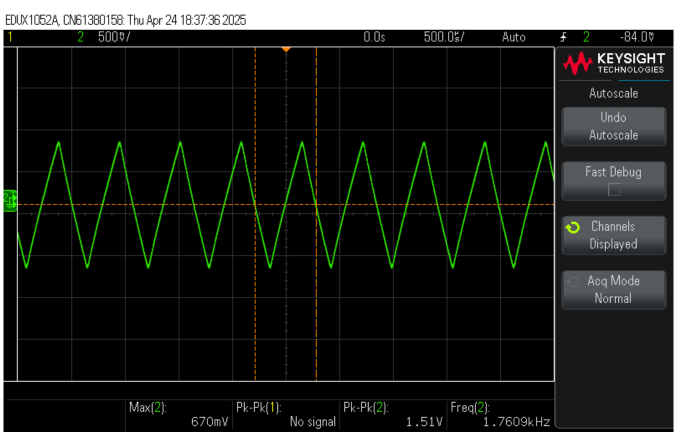


Fig. 31. Triangular Wave $1.5V_{pp}$ and 1.75kHz

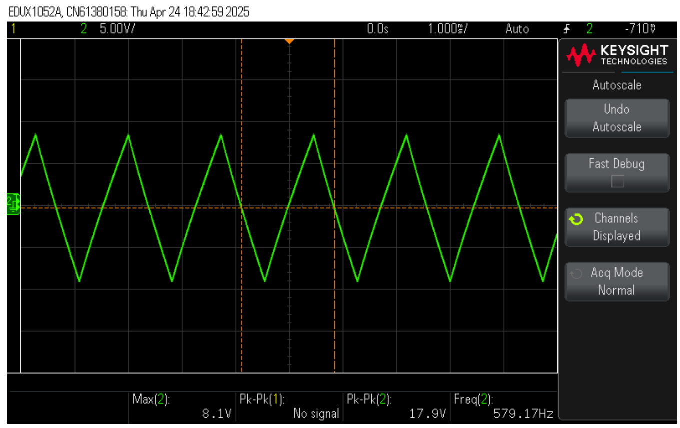


Fig. 32. Triangular Wave $18V_{pp}$ and 580Hz

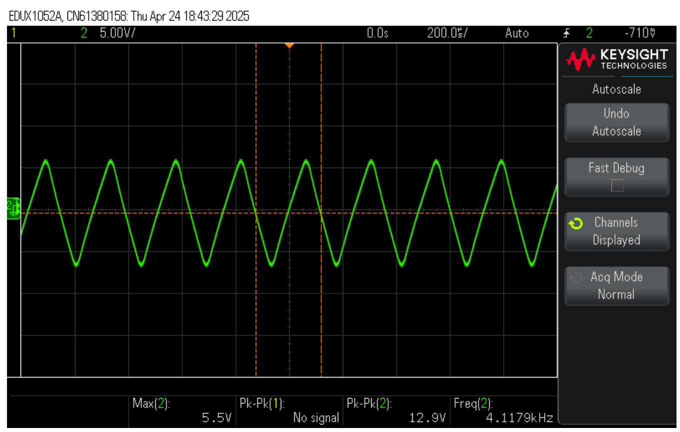


Fig. 33. Triangular Wave $13V_{pp}$ and 4.1kHz

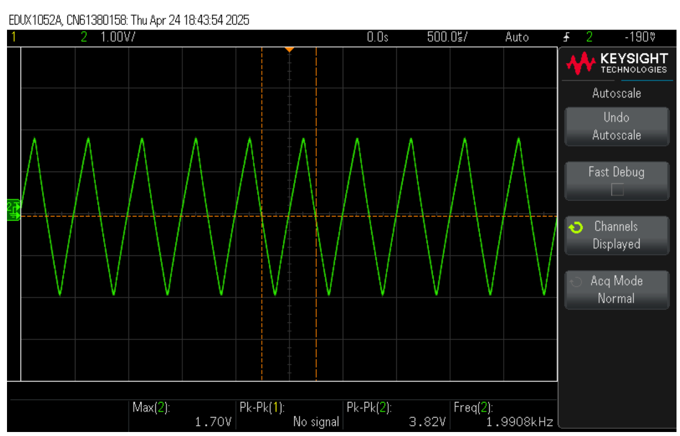


Fig. 34. Triangular Wave $3.8V_{pp}$ and 2kHz

*3) **Sinusoidal Wave Plots**:* These are the plots for Sin wave

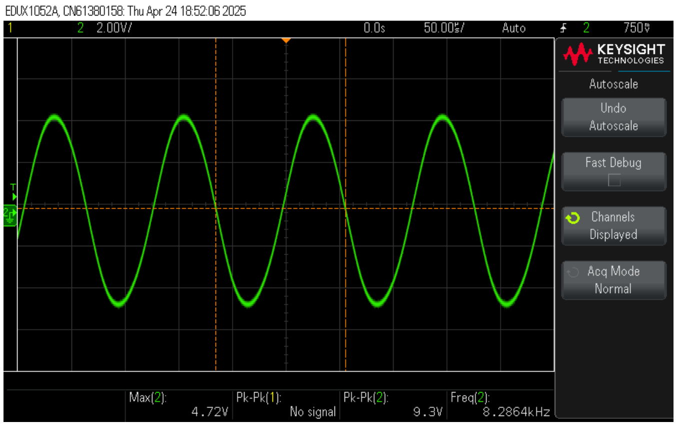


Fig. 35. Sinusoidal Wave $9V_{pp}$ and 8kHz

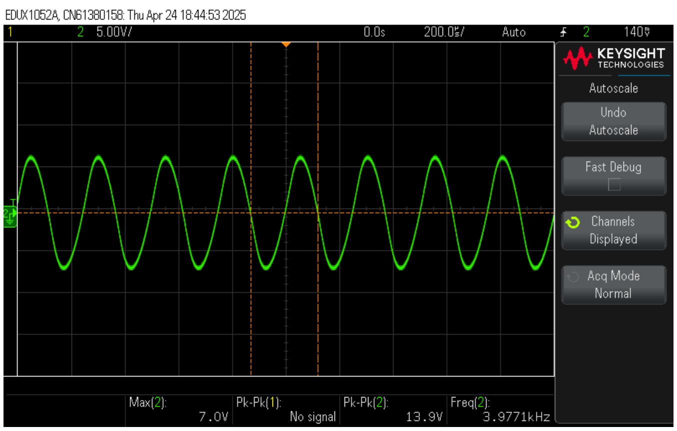


Fig. 36. Sinusoidal Wave $14V_{pp}$ and 4kHz

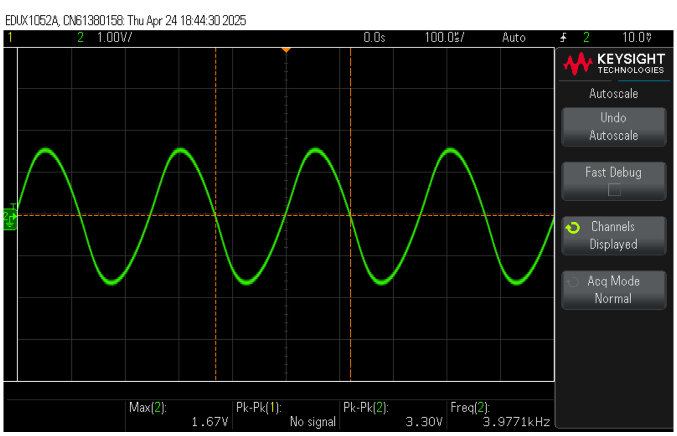


Fig. 37. Sinusoidal Wave $3.3V_{pp}$ and 4kHz

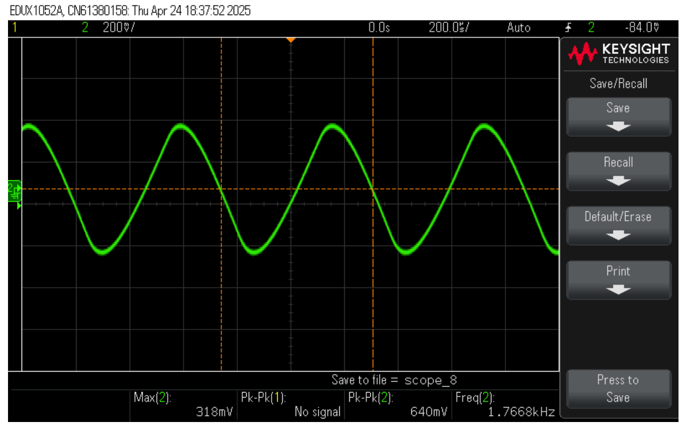


Fig. 38. Sinusoidal Wave $640mV_{pp}$ and 1.8kHz

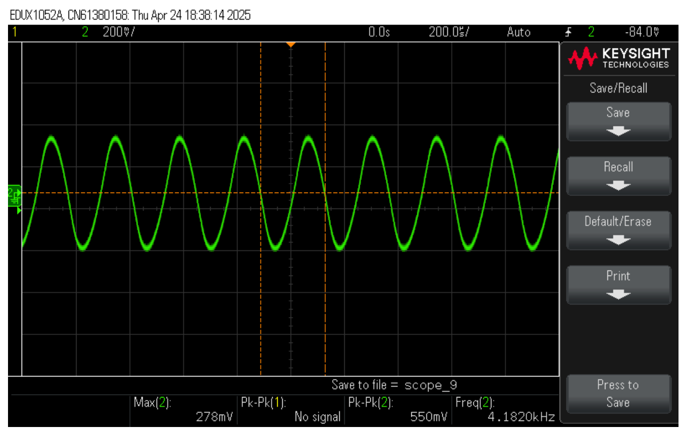


Fig. 39. Sinusoidal Wave $550mV_{pp}$ and 4.2kHz

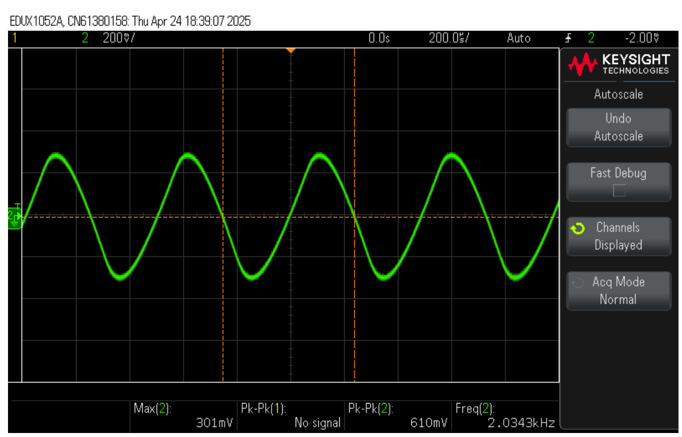


Fig. 40. Sinusoidal Wave $610mV_{pp}$ and 2kHz

I. Hardware Circuit & Performance Review

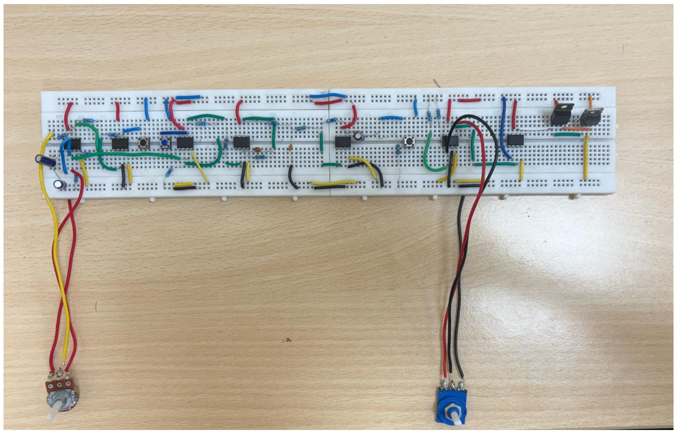


Fig. 41. Hardware Implementation of the circuit

With this our project comes to a close. The final ranges of values are as follows -

Acknowledgment

We would like to express our sincere gratitude to all those who supported and guided us throughout the completion of this project.

First and foremost, we extend our heartfelt thanks to our Professors, for providing us with the opportunity to work on this project and for their valuable insights, continuous encouragement, and constructive feedback. Their guidance has been instrumental in shaping our understanding of the subject.

We would also like to acknowledge the efforts of our TAs for providing us with the necessary resources and a conducive learning environment. Additionally, we are grateful to our peers for their collaboration, discussions, and suggestions, which helped us refine our work. This experience has been both enriching and rewarding for both of us.

Bibliography

- YouTube Video Link
- Multi Vibrator Reference
- Integrator Circuit Reference
- Triangular wave to Sinusoid conversion
- 8-bit DAC Datasheet
- Relaxation Oscillator Topology Reference
- GitHub Repository of all files