Project Title

Project report submitted in partial fulfillment of the degree of
Electronics and Communication Engineering

For the course

Design Lab 1
Final Report

Barnabh Chandra Goswami
19UEC161

Department of Electronics and Communication Engineering
The LNM Institute of Information Technology, Jaipur

17 September 2020

Abstract

In this project we will design a **Function Generator with a single output**, that is, it can give you one waveform at a time. The Function Generator which we are designing in this project after completion will be able to produce 4 fundamental wave-forms which are Square Wave, Triangular Wave, Sinusoidal Wave and Sawtooth Wave.

- FIrstly, we will make Function Generator using electrical components. After this our real work starts which is designing the Function Generator. As per our previous knowledge and basics we gained in this, we will cascade the LM324's and try to check the outputs from each branch of LM324, you can say that this will be periphery of our project, the main real analysis we will do in that respective section.
  So now we have got an idea about this, after this we will talk about the cases how and when we will get the distortion or perfection in the output? What should be the value of resistors and capacitors in the circuit? What could happen when we put the wrong values?
- Secondly, we will make Function Generator using **Arduino Uno**. **Arduino Uno** is a microcontroller board based on the **ATmega328P**, for writing the code we will use Arduino Uno IDE compiler and for running it we use same simulation tool as in first scenario.
  In the era where everything is getting simplified, using an Arduino Uno simplifies the amount of hardware and software development you need to do in order to get a system running. Our current study includes development of a function generator using Arduino Uno and LCD to generate waveforms of different frequencies.

For designing this circuit, we will use an online simulation tool called ***Proteus 8 Professional***. The reference path of all the images and circuit diagrams which we will use in later section will be given just below them and some of them are of Proteus also (observation that will be taken from my laptop), after that in our result we will see our desired outputs one by one and also take all output at once. We will describe some problems which we will face (maybe) and if so, we will discuss their solutions also.
At last like an ordinary report we will write our conclusions and future work, and then provide references of some contents to you.

CONTENTS

1 Motivation **6**

2 Proposed Solution **8**

2.1 About Function Generator 8

2.1.1 **Function Generator using Operational Amplifiers** 9

2.1.2 Astable Multivibrator 10

2.1.3 Op-Amp Integrator 15

2.1.4 About LM324 IC 17

2.1.5 Components Required 18

2.1.6 Whole Circuit 19

2.1.7 Astable Multivibrator Circuit 20

2.1.8 Integrator and Sawtooth Generator Circuits 23

2.2 About Arduino Uno 31

2.2.1 **Function Generator using Arduino Uno** 34

2.2.2 Components Required 34

2.2.3 16x2-bit LCD Display 35

2.2.4 8-bit DAC (Digital to Analog converter) ladder 37

2.2.5 Varying Frequency Circuit 40

2.2.6 Button Switch Circuit 41

2.2.7 Whole Circuit 42

2.2.8 Code in Arduino Uno IDE 42

2.2.9 Explanation of above the Arduino Uno Code 49

3 Results **51**

3.1 **From Operational Amplifiers** 51

3.1.1 One by one Output 51

3.1.2 All Outputs together 55

3.1.3 Distortion Case 57

3.2 **From Arduino Uno** 59

3.2.1 Some Running Shots from Overall Circuit 59

4 Conclusion and future work **65**

List of Figures

2.1 Function Generator 8
2.2 Outline of making Function Generator 9
2.3 Circuit of Astable Multivibrator 10
2.4 Circuit operation 1 12
2.5 Circuit operation 2 13
2.6 Charging and Discharging 14
2.7 Integrator Circuit 15
2.8 Integrator Operation in Graph 15
2.9 LM324 IC Basic Circuit Diagram 17
2.10 Function Generator Circuit 19
2.11 Astable Multivibrator Circuit 20
2.12 Closer look of Circuit 21
2.13 Output from Astable Multivibrator 22
2.14 Integrator A and Sawtooth Generator Circuit 23
2.15 Closer look of Circuit 24
2.16 Output from Integrator A 25
2.17 Updated Circuit with change in resistance 26
2.18 Output from Sawtooth Generator 27
2.19 Integrator B Circuit 28
2.20 Closer look of Circuit 29
2.21 Output from Integrator B 30
2.22 Arduino Uno 31
2.23 Arduino Uno Pin Diagram 33
2.24 16x2 Bit LCD Display 35
2.25 Connection of 16x2 Bit LCD Display with Arduino Uno Board 36
2.26 8-bit DAC ladder with a buffer Op-Amp 37
2.27 Connection of 8-Bit DAC Ladder with Arduino Uno Board 39
2.28 Varying Frequency CIrcuit 40
2.29 Circuit of Button Switch 41
2.30 Whole Circuit of Function Generator using Arduino Uno Board 42
3.1 Square wave Output 51
3.2 Triangular wave Output 52
3.3 Sinusoidal wave Output 53
3.4 Sawtooth wave Output 54

3.5 All Outputs Together 55
3.6 Closure look of all Outputs 56
3.7 Distorted Outputs 57
3.8 Closer look of Distorted Outputs 58
3.9 Sinusoidal Wave of 0.3 kHz 59
3.10 Sinusoidal Wave of 0.6 kHz 60
3.11 Sawtooth Wave of 0.2 kHz 61
3.12 Sawtooth Wave of 0.8 kHz 62
3.13 Triangular Wave of 0.1 kHz 63
3.14 Triangular Wave of 0.9 kHz 64

1 Motivation

As my Project title suggests **"Function Generator with a Single Output"**, it will give you the Square wave, Triangular wave, Sinusoidal wave and finally the Sawtooth wave one at a time in the Oscilloscope. Any viewer who wants to test this can see the waves they required at output by just using the Switch i.e. Square mode, Triangular mode, Sinusoidal mode and Sawtooth mode.

I have taken this project because I thought that this will be the best opportunity to analyse the Interior of Function Generator (roughly) which we used to see every time in our Electronics Lab in Universities or Collages, since it is widely and commonly used in Electronics among students and researchers also.

Well indeed my circuit was not that much in detail i.e. having Frequency Controller, Amplitude Controller, DC Offset etc. My work in this project basically shows you the main fundamental waves generating from my Function Generator.

Now if I talk about my approach, I use the LM324's as a fundamental component in this. Why? Because at the present time I think I feel most comfortable with LM324 in IC's and I have also studied this IC very well rather than other IC's like 555 timers, L293D etc. The circuit is also quite understandable to me. Hence due to these reasons I think that the use of LM324 IC will be perfect for me.

There are also many other ways to make a basic Function Generator like: -

- We can make a Function Generator with the help of LM7812 and LM7912 IC's.
- We can make a Function Generator with the help of IC 555 Timer also.
- We can make a Function Generator with the help of ICL8038 Signal generator, with this it will become quite easy to make a simple Function Generator.
- We can make a Function Generator with the help of IC XR2206 which is a monolithic generator and like IC8038 it provides a very simple function generator with only a few external components.
- And we can make a Function Generator through IC 741 Op-Amp or LM324(which will be used in this one).

Here we describe the components by which they can be made easily, there are other methods also.

Also we are going to use Arduino Uno Board later in this Project where we will see how can we design by writing the Code in Arduino Uno of Function Generator in many ways, like we can make firstly Codes of each types of waves then in the end we will combine all of them or we can also start as a whole or we can use DDR and PORTD in it or we can simply use serial.print also, here in this project we will follow the approach using **DDR (Data Direction Registers)** where we will also describe:

- 8-Bit DAC ladder
- 16x2 Bit LCD Display
- Some Connections made with Arduino Uno microcontroller ATmega328P to other Circuits.

With the help of Arduino Uno we will get more and more comfortable with hardware and software due to Arduino we can easily make Function Generator with a Varying Frequency and even Amplitude with a vast range scale, you can yourself imagine how easy it becomes to just think a logic of making it and apply it to the code using ordinary Coding appraoch, which we see will later in this report.

At last I am greatly motivated to work in this project because it provides me the extra knowledge about working of Function Generator which might help me future also. And since I feel I know the basics of the circuits used in this Function Generator (like Astable Multivibrator, Integrator, Differentiator etc), that's why I feel that I can shoot it.

2 PROPOSED SOLUTION

2.1 ABOUT FUNCTION GENERATOR

So many of the experiments in our collages or universities workplace build use of oscilloscopes and Function generators as a result of it's helpful to find out their general operation. because the Function Generators produces signal sources which give a identifiable voltage applied over a identifiable time, like a "Sinusoidal wave", "Triangular wave" or "Square wave" signal etc.

A Function generator has several sensible applied science applications, it is usually necessary to supply completely different styles of wave-forms to check and rectify various electronic circuits and devices.

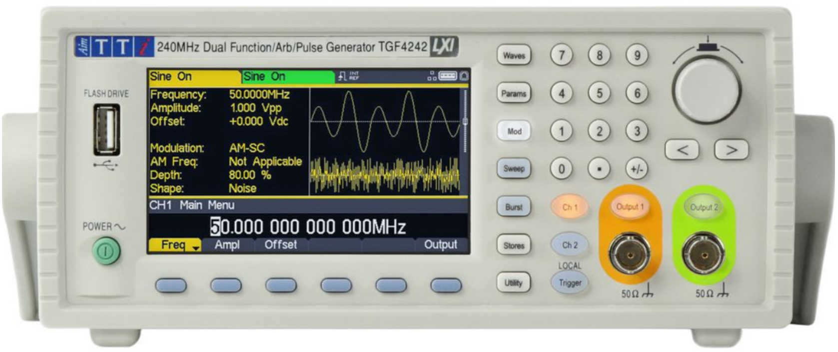

Figure 2.1: Function Generator

2.1.1 FUNCTION GENERATOR USING OPERATIONAL AMPLIFIERS

My **outline** in making Function Generator looks like this: -

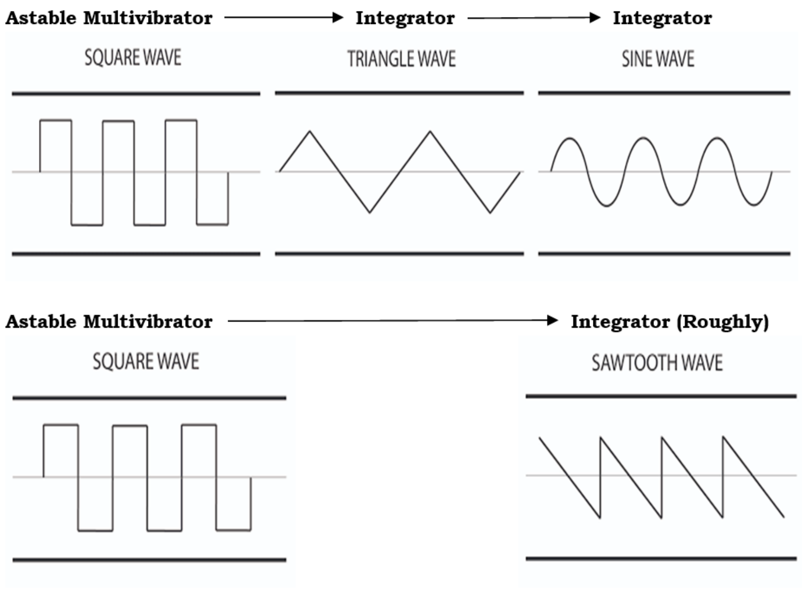

Figure 2.2: Outline of making Function Generator

In the making of Function Generator, the first step I take is the making of the Astable Mutlivibrator.

2.1.2 ASTABLE MULTIVIBRATOR

The Astable Multivibrator is additionally referred to as as a free-running Multivibrator. it has 2 quasi-stable states. we do not have to be compelled to given any external signal to provide the changes in its state. The part values determine the time that for which circuit remains in every state, since the Astable Multivibrator oscillates between 2 states, it is also handy to generate Square waves.

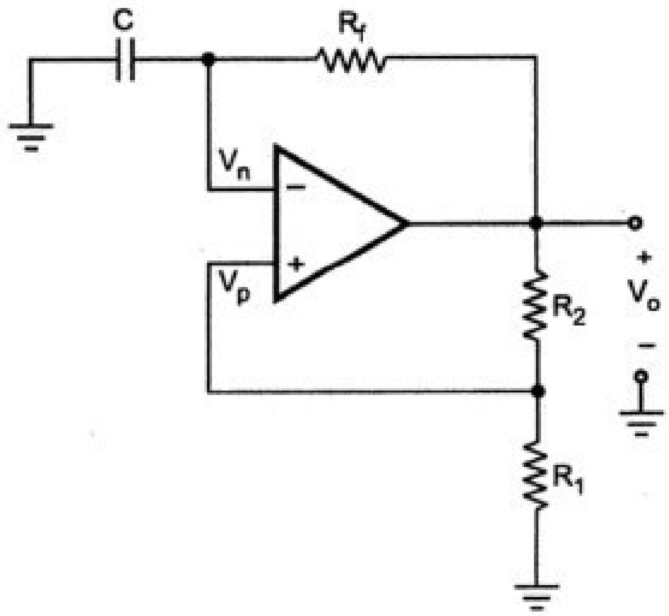

Figure 2.3: Circuit of Astable Multivibrator

The circuit is similar to Schmitt trigger except that the input voltage is replaced by the capacitor. As shown in the Figure 2.3 the comparator and positive feedback resistors R1 and R2 form an inverting Schmitt trigger.

When V_0 is at V_{Sat}, the feedback voltage is called the upper threshold voltage V_{UT} and is given as

$$V_{UT} = R_1 \frac{V_{sat}}{R_1 + R_2}$$

When V_0 is at $-V_{Sat}$, the feedback voltage is called the lower threshold voltage V_{LT} and is given as

$$V_{LT} = -\frac{R_1 V_{sat}}{R_1 + R_2}$$

Circuit operation

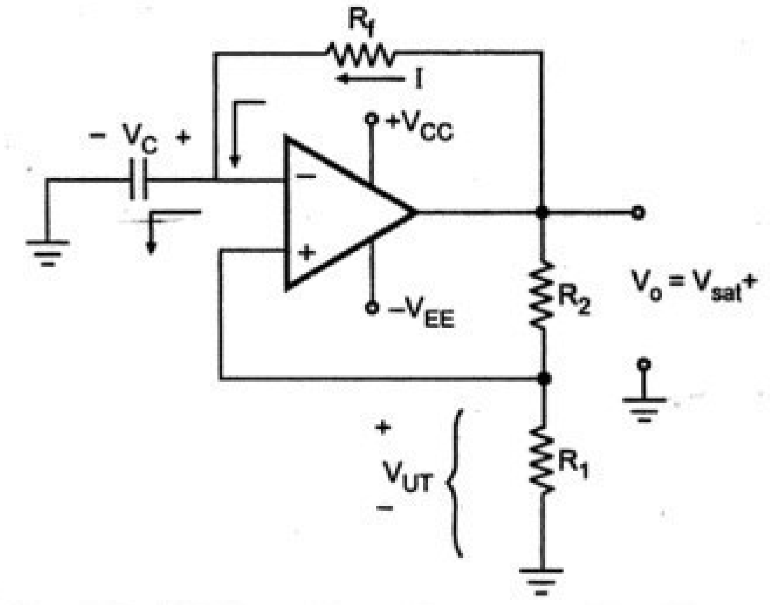

Figure 2.4: Circuit operation 1

1. When power is turned ON, V_0 mechanically swings either V_{Sat} or to $-V_{Sat}$ since these are the only sole stable states allowed by Schmitt trigger. Let us assume it swings to $+V_{Sat}$.
2. Now capacitor starts charging towards $+V_{Sat}$ through the feedback path provided by the resistor Rf to the inverting input. With the provision that the capacitor voltage V_C is less than V_{UT}, the output voltage remains constant at V_{Sat}.

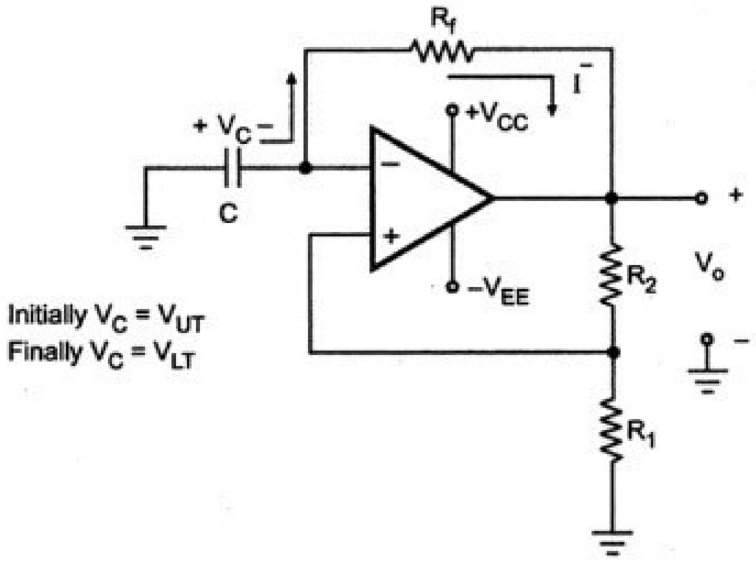

Figure 2.5: Circuit operation 2

3. As soon as V_C gets charged to a value faintly greater than V_{UT}, the input goes positive with respect to the input. This switches the output voltage from $+V_{Sat}$ to $-V_{Sat}$.
4. As V_0 switches to $-V_{Sat}$, capacitor starts discharging through the current I discharge capacitor to 0 V and recharges capacitor to V_{LT}. When V_C becomes slightly more negative than the feedback voltage V_{LT}, output voltage switches back to again $+V_{Sat}$.

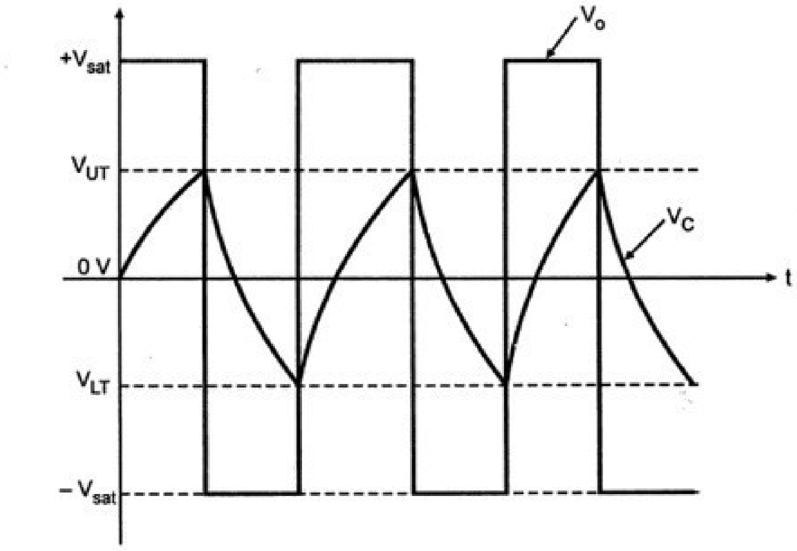

Figure 2.6: Charging and Discharging

Charging and Discharging Equations.

$$T_C = RC \; \ln\left(\frac{1+\beta}{1-\beta}\right)$$

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

$$T_C = T_d$$

Total Time Period (T)

$$T = T_C + T_d$$

$$\boxed{T = 2RC \; \ln\left(\frac{1+\beta}{1-\beta}\right)}$$

From Astable Multivibrator now we get the Square wave at output and now this Square wave will work as an input for an Integrator to convert it into a Triangular wave or Sawtooth wave.

Therefore, my second step is to feed output of Astable Multivibrator to input of Integrator.

2.1.3 OP-AMP INTEGRATOR

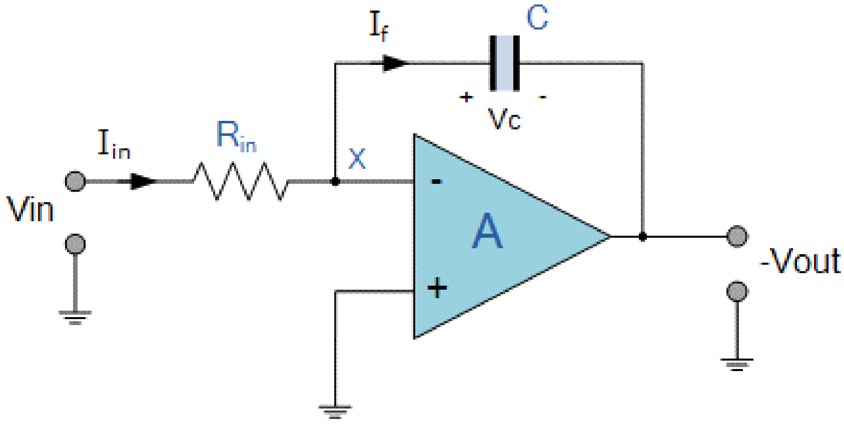

Figure 2.7: Integrator Circuit

The Op-amp Integrator is a type of operational amplifier circuit that performs the mathematical process of Integration, that is we will cause the output to reply to changes within the input voltage over time since the op-amp integrator gives us an output voltage which is directly proportional to the integral of the input voltage. That is, if we tend to feed any arbitrary wave to it, it will give us the Integration of that wave.

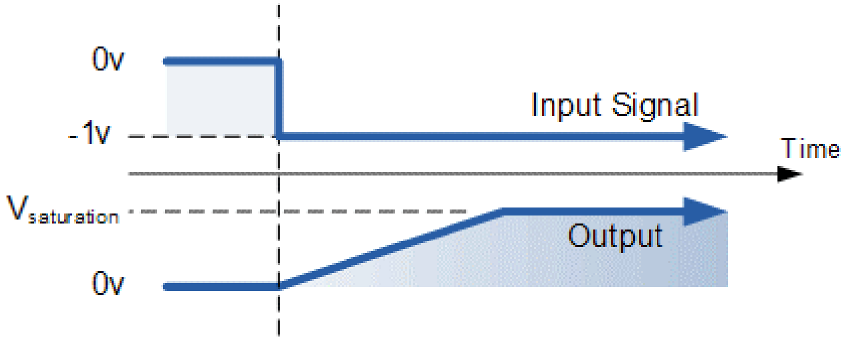

Figure 2.8: Integrator Operation in Graph

$$V_c = \frac{Q}{C}, \qquad V_c = V_x - V_{out} = 0 - V_{out}$$

$$\therefore -\frac{dV_{out}}{dt} = \frac{dQ}{Cdt} = \frac{1}{C}\frac{dQ}{dt}$$

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

The current flowing through the feedback capacitor C is given as:

$$I_f = C\frac{dV_{out}}{dt} = C\frac{dQ}{Cdt} = \frac{dQ}{dt} = \frac{dV_{out}.C}{dt}$$

Assuming that the input impedance of the op-amp is never ending that is infinite (ideal conditions in Op-Amp), therefore no current flows into the op-amp terminal. So, by the use of Kirchhoff's Voltage law the nodal equation at the inverting input terminal is given as:

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

$$\therefore \frac{V_{in}}{V_{out}} \times \frac{dt}{R_{in}C} = 1$$

From which we derive an ideal voltage output for the Op-amp Integrator as:

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

On simplifying it we can also write it as:

$$V_{out} = -\frac{1}{j\omega RC}V_{in}$$

2.1.4 About LM324 IC

LM324 IC is a Quad op-amp integrated with four operational amplifiers sourced by a ordinary power reservoir. The differential input voltage range can be equivalent to that of power supply voltage. The default input offset voltage is very low which is of magnitude 2 mili-Volts. The operating temperature ranges from 0 ^{o}Celsius to 70 ^{o}Celsius at ambient whereas the maximum junction temperature can be up to 150 ^{o}Celsius. Generally, op-amps can perform mathematical operations. It is mainly used as a comparator, also can be used as transducer amplifier, DC gain block etc. It has immense dc voltage gain of about 90-100dB. This IC can be handled over wide range of power supply from 3V to 32V for mono power supply or from ±1.5V to ±16V for dual power supply and it also supports large output (O/P) voltage swing.

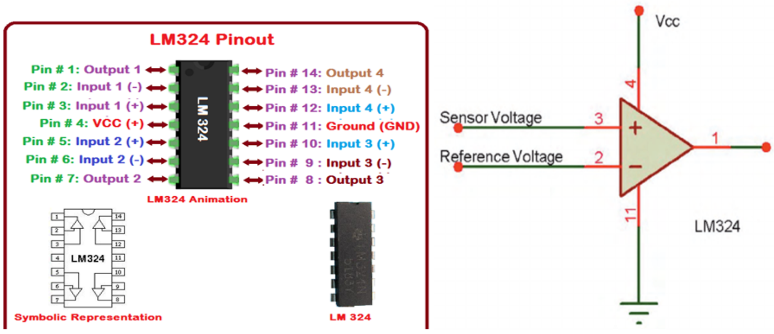

Figure 2.9: LM324 IC Basic Circuit Diagram

Now using all the above logics, we can now understand the working of our Function Generator with single output.

2.1.5 Components Required

Before starting let me tell you all the components used in making Function Generator with single output: -

1. 11 Resistors (22 k, 100 k, 200 k and 220 kOhm).
2. 2 Variable Resistors (10k and 50kOhm).
3. 4 Electrolytic Capacitors (10nF, 33nF, 33nF and 1uF).
4. 5 LM324's (Astable Multivibrator A, Integrator A, Integrator B, Sawtooth Generator A and Sawtooth generator B).
5. 4-Terminal Switch.
6. Oscilloscope.

2.1.6 Whole Circuit

Now let's look at the overall circuit first: -

Function Generator Circuit Diagram in PROTEUS

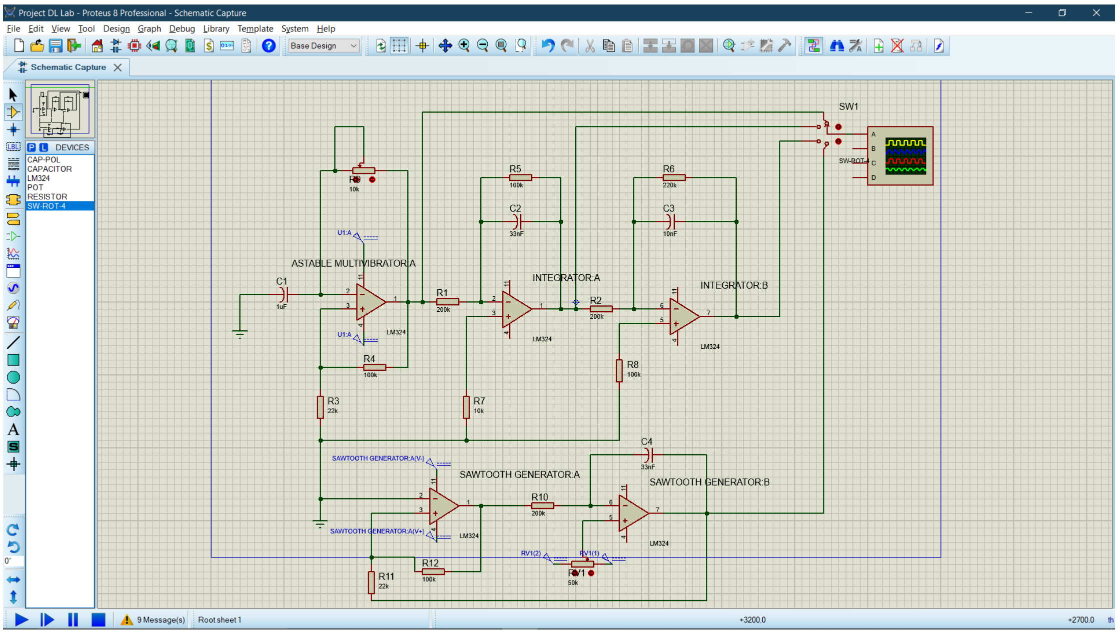

Figure 2.10: Function Generator Circuit

The above circuit looks clumsy but don't worry we will describe all one by one.

2.1.7 ASTABLE MULTIVIBRATOR CIRCUIT

First of all, let start with the ASTABLE MULTIVIBRATOR A: -

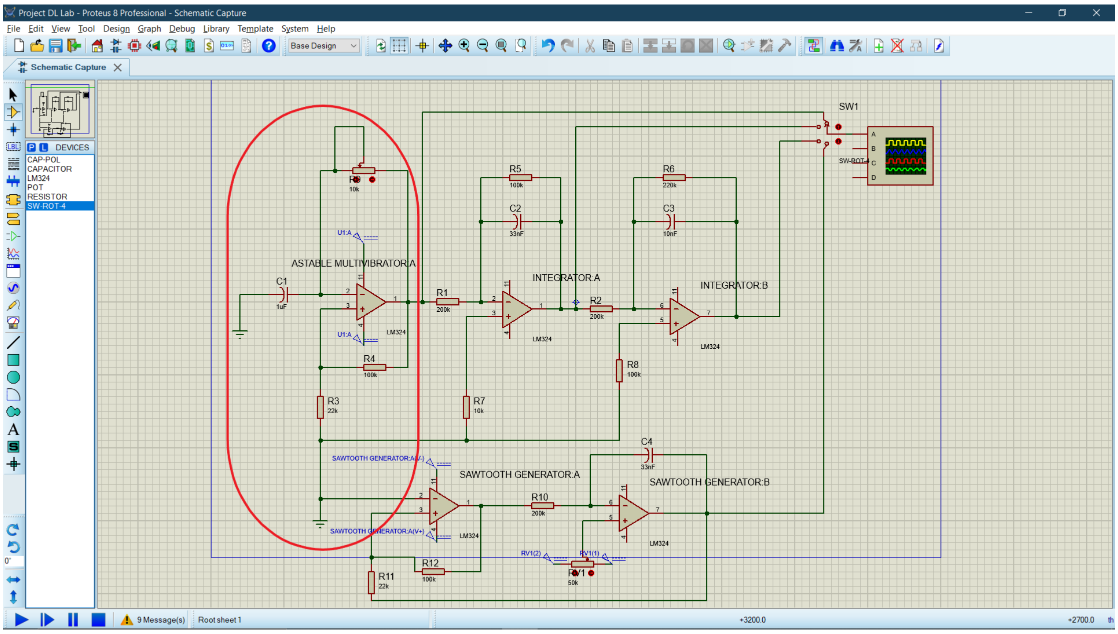

Figure 2.11: Astable Multivibrator Circuit

As you can see in the circuit diagram above in the red capsule region our Astable Multivibrator A (LM324) is there.

Let's take a closer look!

ASTABLE MULTIVIBRATOR

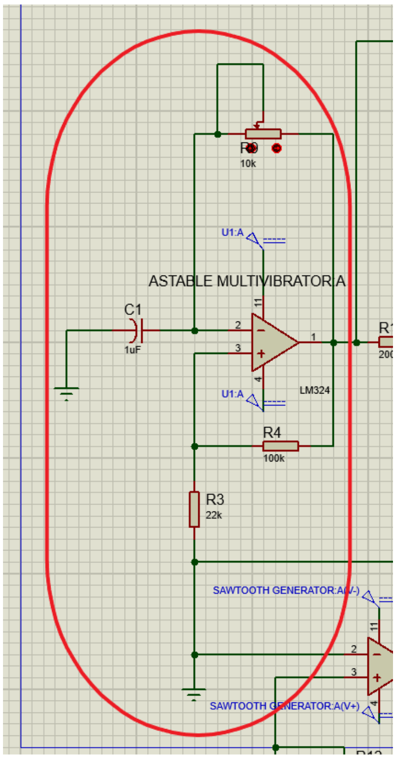

Figure 2.12: Closer look of Circuit

As you can see circuit of Astable Multivibrator A closely we can understand better now how it works because we have already answered its working in the above sections so there will be no problem now, we can clearly see that it will produce the Square wave, only difference is that I used the Potentiometer **R9** of 10 kOhm instead of using the normal resistor so that I can check the perfect formation of Square wave. We also give the +/-5V DC Supply in first LM324 (ASTABLE MULTIVIBRATOR A).
We observe that greater the value of Potentiometer resistor **R9** greater will be the distortion in the output.

Let's see our **output** from ASTABLE MULTIVIBRATOR A.

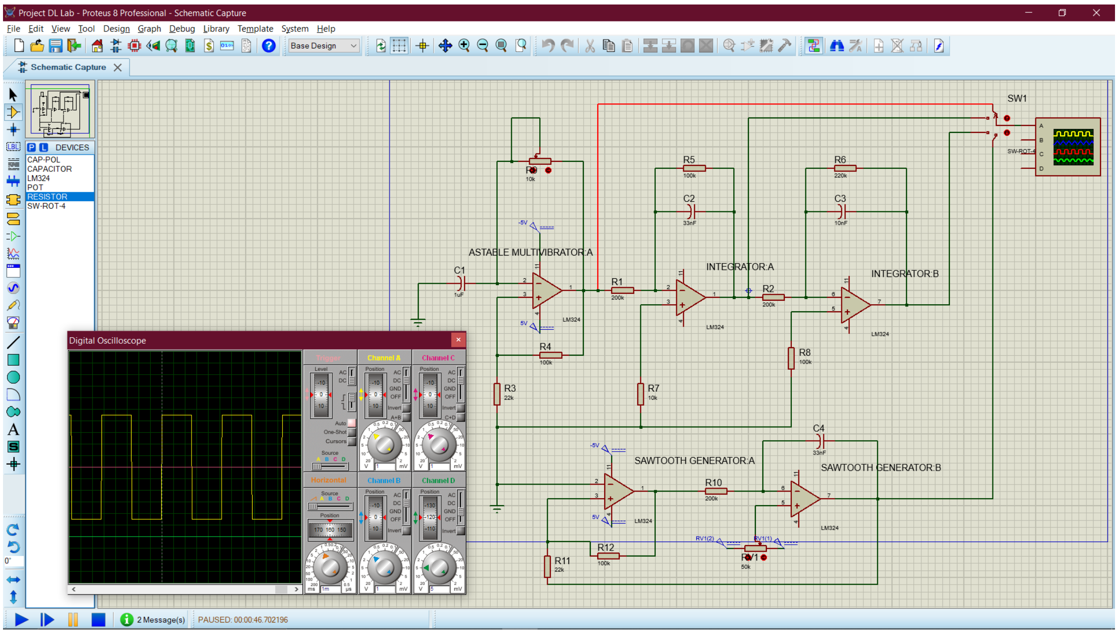

Figure 2.13: Output from Astable Multivibrator

You can clearly see that our output is the Square wave in the Oscilloscope, and near the oscilloscope you will see a 4-Terminal Switch which is set at the 1st terminal which takes the output from ASTABLE MULTIVIBRATOR A. The red wire shows the taken output.
Now I want to generate a Triangular wave and a Sawtooth wave, so I can generate this by cascading INTEGRATOR A (LM324) to ASTABLE MULTIVIBRATOR A (LM324), by which we will get the integration of Square wave which is our required Triangular wave or Sawtooth wave.

Let's make it easy by looking into circuit of INTEGRATOR A and SAWTOOTH GENERATOR A and B: -

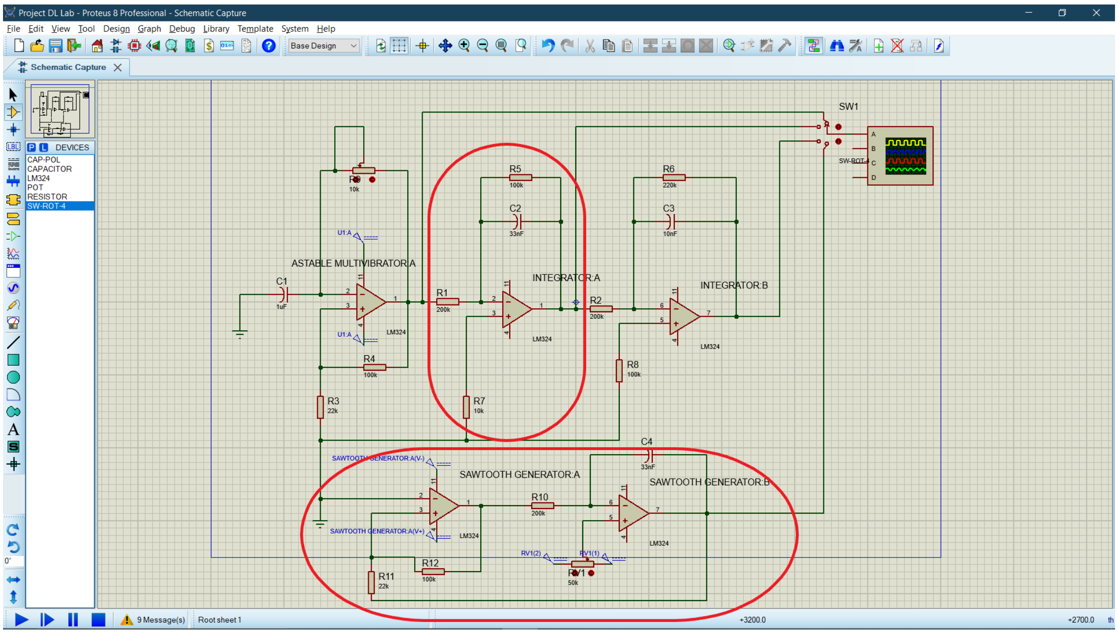

Figure 2.14: Integrator A and Sawtooth Generator Circuit

As you can see in the circuit diagram above in the red capsule region our Integrator A (LM324) and Sawtooth Generator A and B are there.

Let's take a closer look!

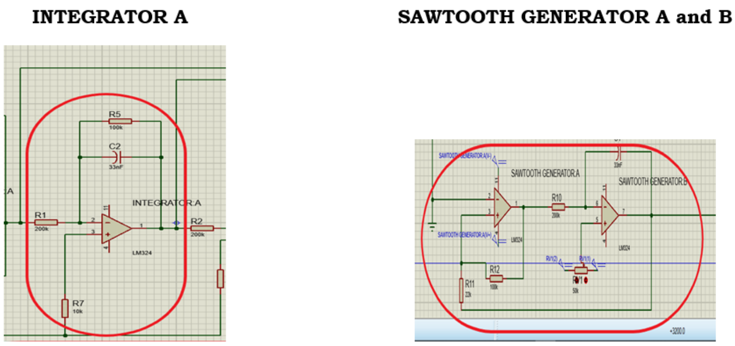

Figure 2.15: Closer look of Circuit

In the **Integrator A** we can see that we use the combination of R and C such that we can get the best Triangular wave from its output at 2nd terminal of 4-Terminal Switch.
We observe that when we decrease R5 and C2 any one of that or both then our generated Triangular wave at output gets distorted, and the same thing happens when we decrease R1 also.

Let’s see our **output** from INTEGRATOR A.

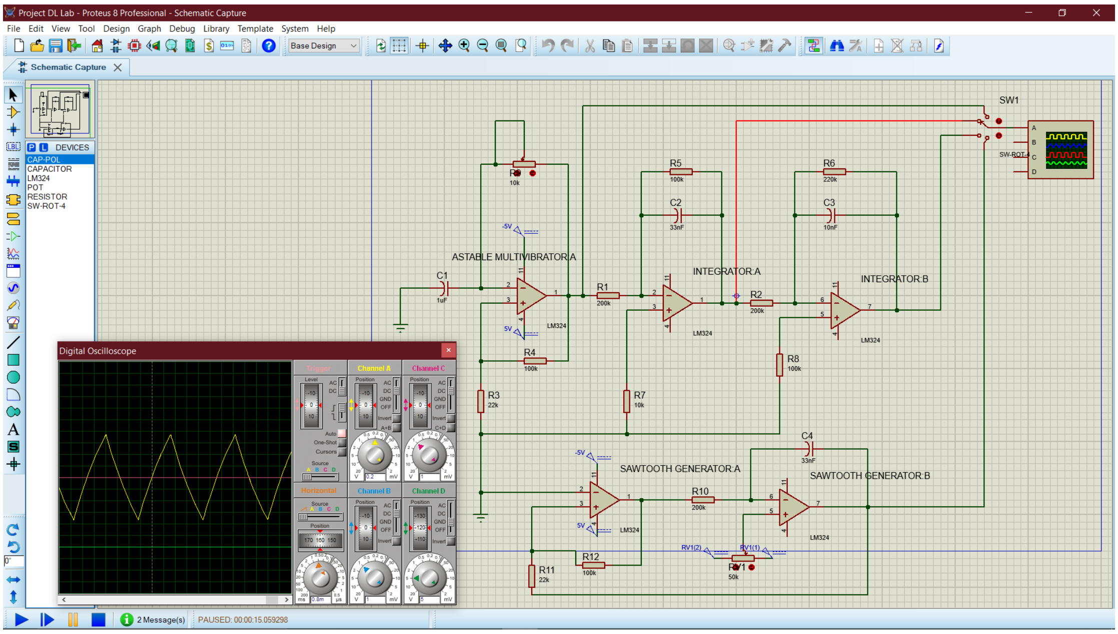

Figure 2.16: Output from Integrator A

We are clearly getting a Triangular wave at the output; near the oscilloscope you will see a 4-Terminal Switch which is set at the 2nd position which takes the output from INTEGRATOR A. The red wire shows the taken output. As we increase the value of R5 or C2 we will get a more perfect Triangular wave.

Let me show you what happens when we increase the R5 from 100 kOhm to 1000 kOhm.

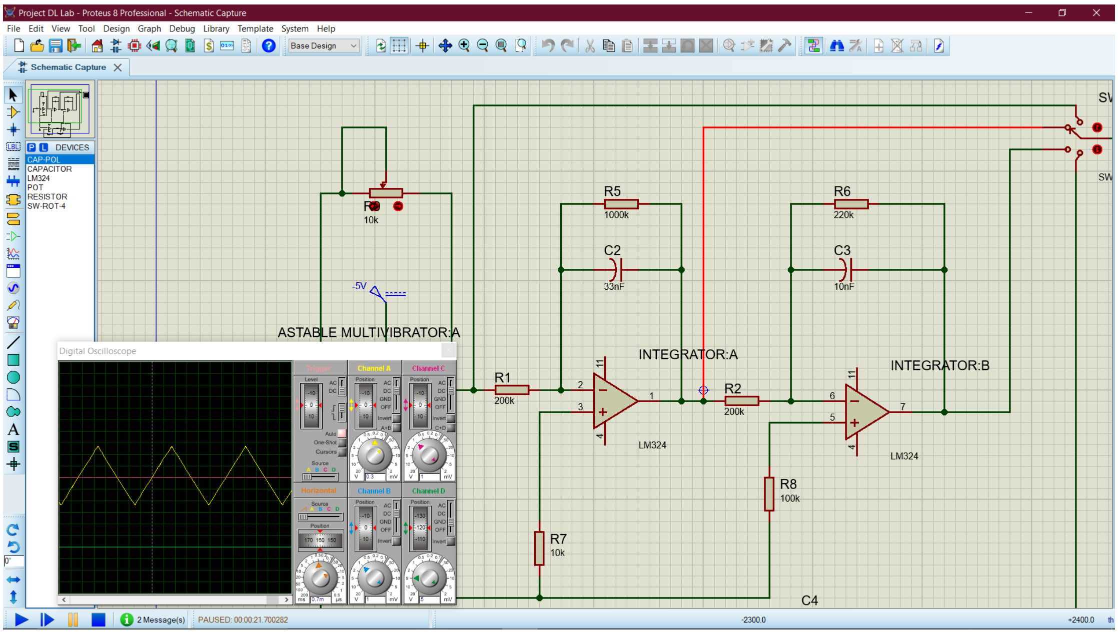

Figure 2.17: Updated Circuit with change in resistance

You can clearly see that the triangular waves are now more perfect than before. The red wire shows the taken output.

Now let's talk about the **Sawtooth Generator A and B.**

In Sawtooth Generator we use again LM324 Square wave circuit and cascade with other LM324, the only difference from changing Triangular wave to Sawtooth wave is to connect a +/- DC source Supply at the positive terminal of other LM324 (SAWTOOTH GENERATOR B) that is 5th pin of it with a variable resistor of 50kOhm, by varying the this resistor we can get our required Sawtooth wave according to it.
And this Sawtooth wave we will get at the 4th position of our 4-Terminal Switch near the oscilloscope.

Let 's see the Sawtooth wave from our **output** now.

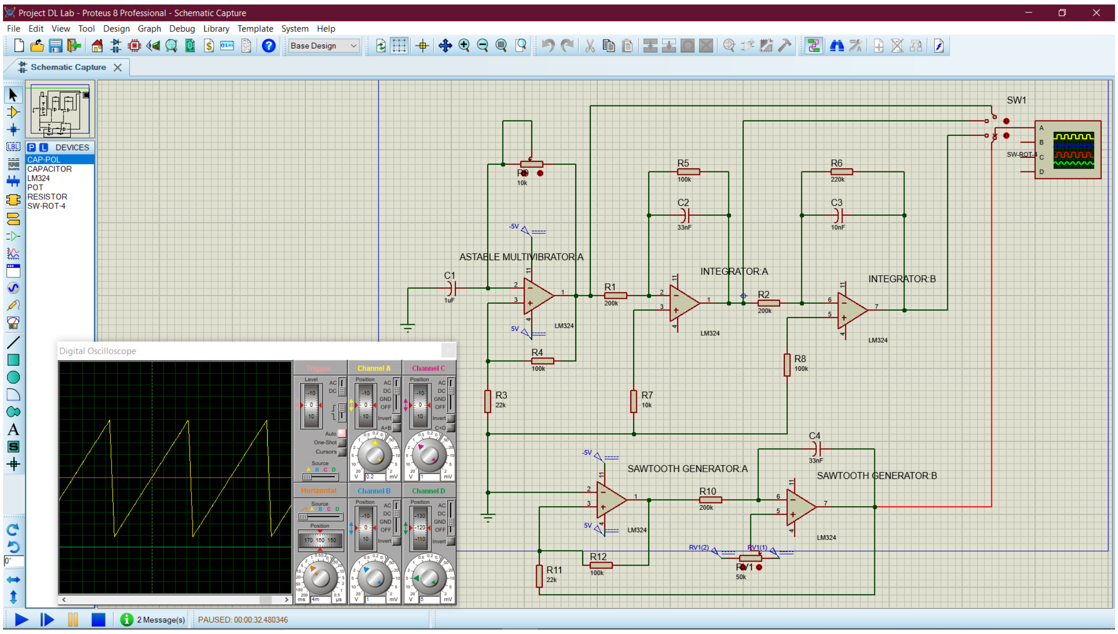

Figure 2.18: Output from Sawtooth Generator

You can clearly see our result!!! means that the circuit is working fine. The red wire shows the taken output.

Now since we get the Triangular wave and Sawtooth wave as our output so our last task is to again integrate the Triangular wave and make it finally a Sinusoidal wave, therefore we need an integrator again that's why we cascade INTEGRATOR A to INTEGRATOR B so that we can now get Sinusoidal wave at last.

Let's see it through looking into the circuit of INTEGRATOR B.

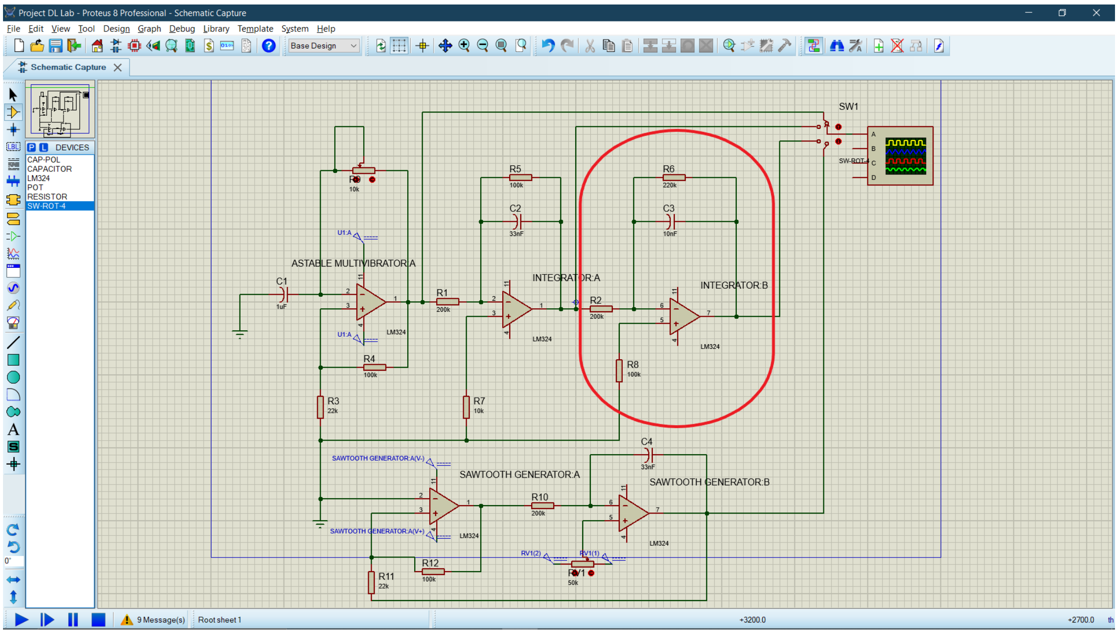

Figure 2.19: Integrator B Circuit

As you can see in the circuit diagram above in the red capsule region our Integrator B (LM324) is there.

Let's take a closer look!

INTEGRATOR B

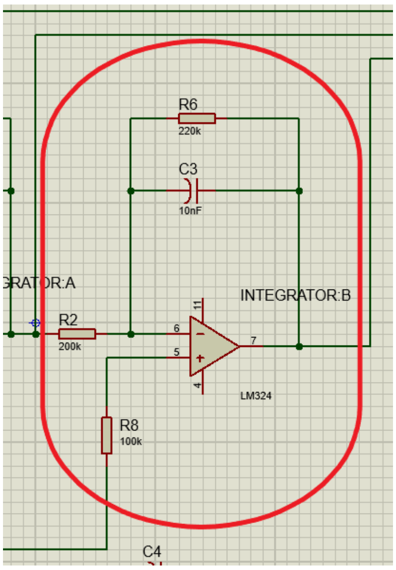

Figure 2.20: Closer look of Circuit

As you can see in our final circuit of other integrator that is INTEGRATOR B, all things and theories remain same, the only difference this time is that we want a Sinusoidal wave so our output from INTEGRATOR A which is a Triangular wave will act as a input for the INTEGRATOR B so we will now do the integration of Triangular wave and we know that the integration of Triangular wave is Sinusoidal wave.

Hence our output will be Sinusoidal wave.

Let's see our final **output** (Sinusoidal wave) of the whole circuit.

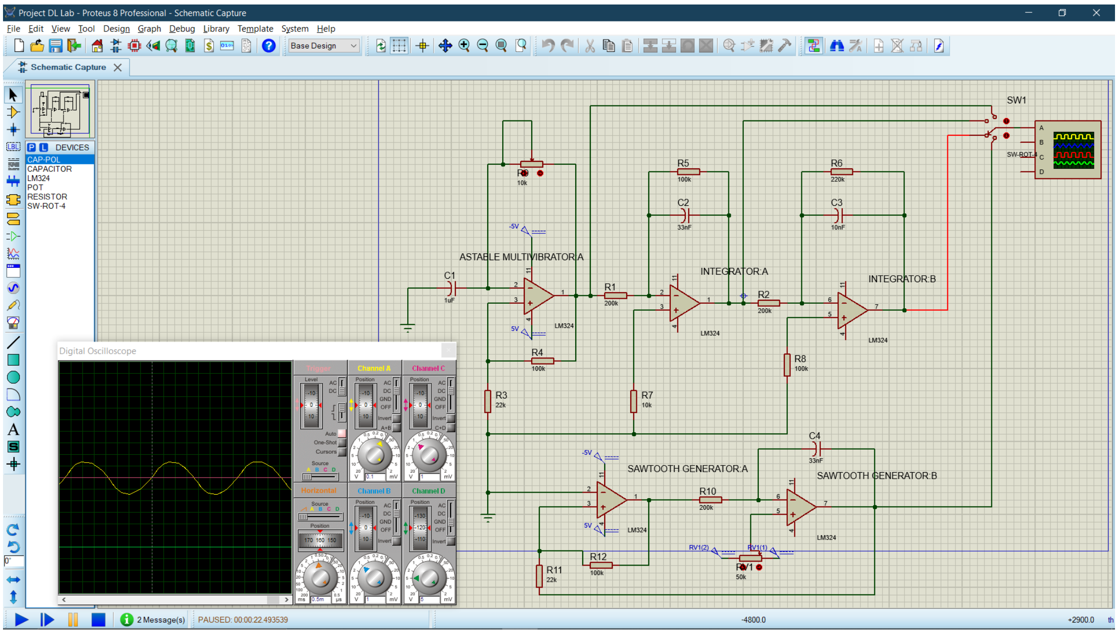

Figure 2.21: Output from Integrator B

Finally, we can clearly see that our output is the Sinusoidal wave in the Oscilloscope, and near the oscilloscope you will see a 4-Terminal Switch which is set at the 3rd terminal which takes the output from INTEGRATOR B. and as always, the red wire shows the taken output.
Hence now our final work is to collect all these theories and observations into one sentence, which we will do in Results Section.

2.2 About Arduino Uno

Arduino Uno is a microcontroller board having a 8-bit ATmega328P microcontroller together with ATmega328P, it consists of different components like crystal oscillator, voltage regulator, serial communication etc, to support the microcontroller. Arduino Uno has 14 digital i/p or o/p pins (out of which 6 can be used as Pulse Wave Modulation outputs), 6 analog i/p pins, a USB connection, an influencing power barrel jack, an ICSP header and a push reset button.

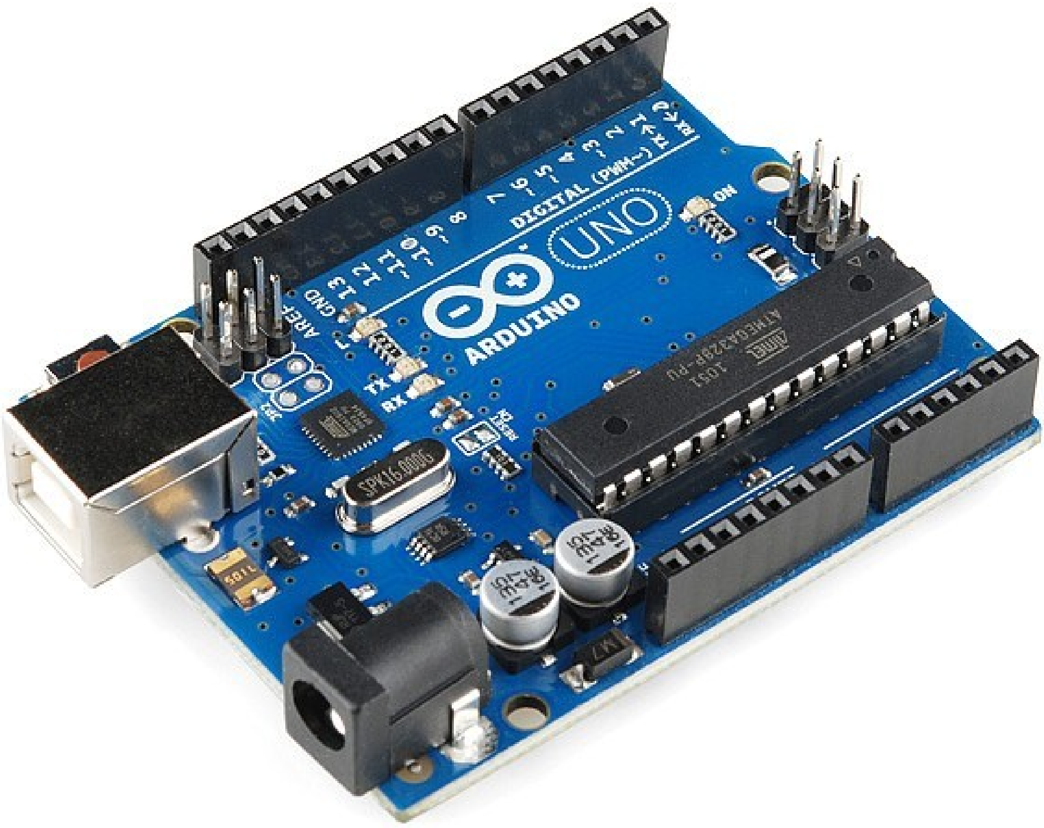

Figure 2.22: Arduino Uno

Some details about Arduino Uno Pins.

- **Vin**: This is the input voltage pin of the Arduino board used to take input supply from the external power source.
- **5V**: This pin of the Arduino board is used as a regulated power supply voltage and it is used to give supply to the board as well as onboard components.
- **3.3V**: This pin of the Arduino board is used to provide a supply of 3.3V which is produced from the voltage regulator on the board.
- **GND**: This pin of the board is used to ground the Arduino board.
- **Reset**: This pin of the board is used to reset the microcontroller.
- **Analog Pins**: The pins A0 to A5 are used as an analog inputs and it is in the range of around 0-5V.
- **Digital Pins**: The pins 0 to 13 are used as a digital i/p or o/p for the Arduino board.
- **Serial Pins**: These pins are also known as a UART pin. It is used for communication between the Arduino board and the computer or other electronic devices. The transmitter pin number 1 and receiver pin number 0 is used to transmit and receive the data.
- **External Interrupt Pins**: This pin of the Arduino board is used to produce the External interrupt and it is done by pin numbers 2 and 3.
- **PWM Pins**: This pins of the board is used to convert the digital signal into an analog by varying the width of the Pulse. The pin numbers 3,5,6,9,10 and 11 are used as a Pulse Wave Modulation (PWM) pin.
- **SPI Pins**: This is the Serial Peripheral Interface pin, it is used to maintain SPI communication with the help of the SPI library. SPI pins includes:
  - SS: Pin number 10 is used as Slave Select
  - MOSI: Pin number 11 is used as Master Out Slave In
  - MISO: Pin number 12 is used as Master In Slave Out
  - SCK: Pin number 13 is used as Serial Clock
- **LED Pin**: The board has an inbuilt LED, using digital pin-13 the LED glows only when it becomes high.
- **AREF Pin**: This is an analog reference pin of the Arduino board. It is used to produce a reference voltage from an external power supply.

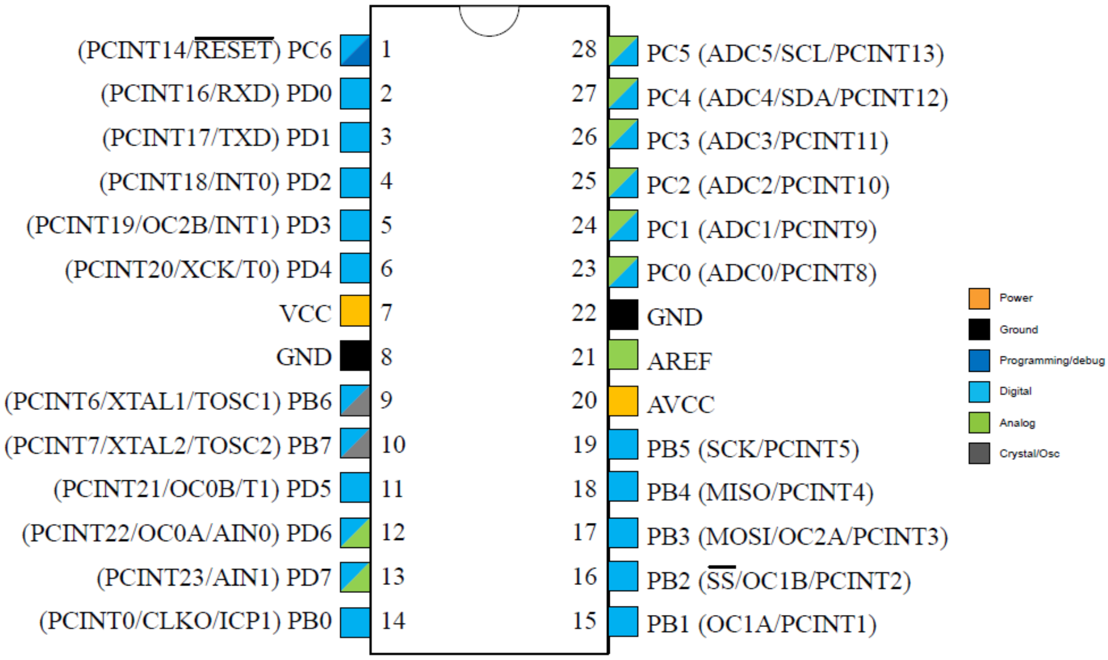

Figure 2.23: Arduino Uno Pin Diagram

2.2.1 FUNCTION GENERATOR USING ARDUINO UNO

After learning about Arduino Uno board we can now move ahead in making Arduino Uno Code of "Function Generator having single output" and our aim is to make such code which will give us types of waves with varying frequency from 1 Hz to 1000 kHz on output.

2.2.2 COMPONENTS REQUIRED

Now we have to design Function Generator using Arduino Uno board, so the components we will use are:

1. Arduino Uno Board
2. 16x2 bit LCD Display
3. 8-bit DAC (Digital to Analog converter) ladder
4. Circuit for "Varying Frequency"
5. And at last a "Button Switch Circuit" to make one pin HIGH at a time.

Since we have already learned about **Arduino Uno** Board, so let's move ahead to **16x2 bit LCD Display**

2.2.3 16X2-BIT LCD DISPLAY

16×2 LCD is named so because it has 16 Columns and 2 Rows. There are a lot of combinations available in LCD's like 8×1, 8×2, 10×2, 16×1 etc. but the most used is the 16×2 LCD. So, it we can have (16×2=32) 32 characters in total and each character will be made of 5×8 Pixel Dots.

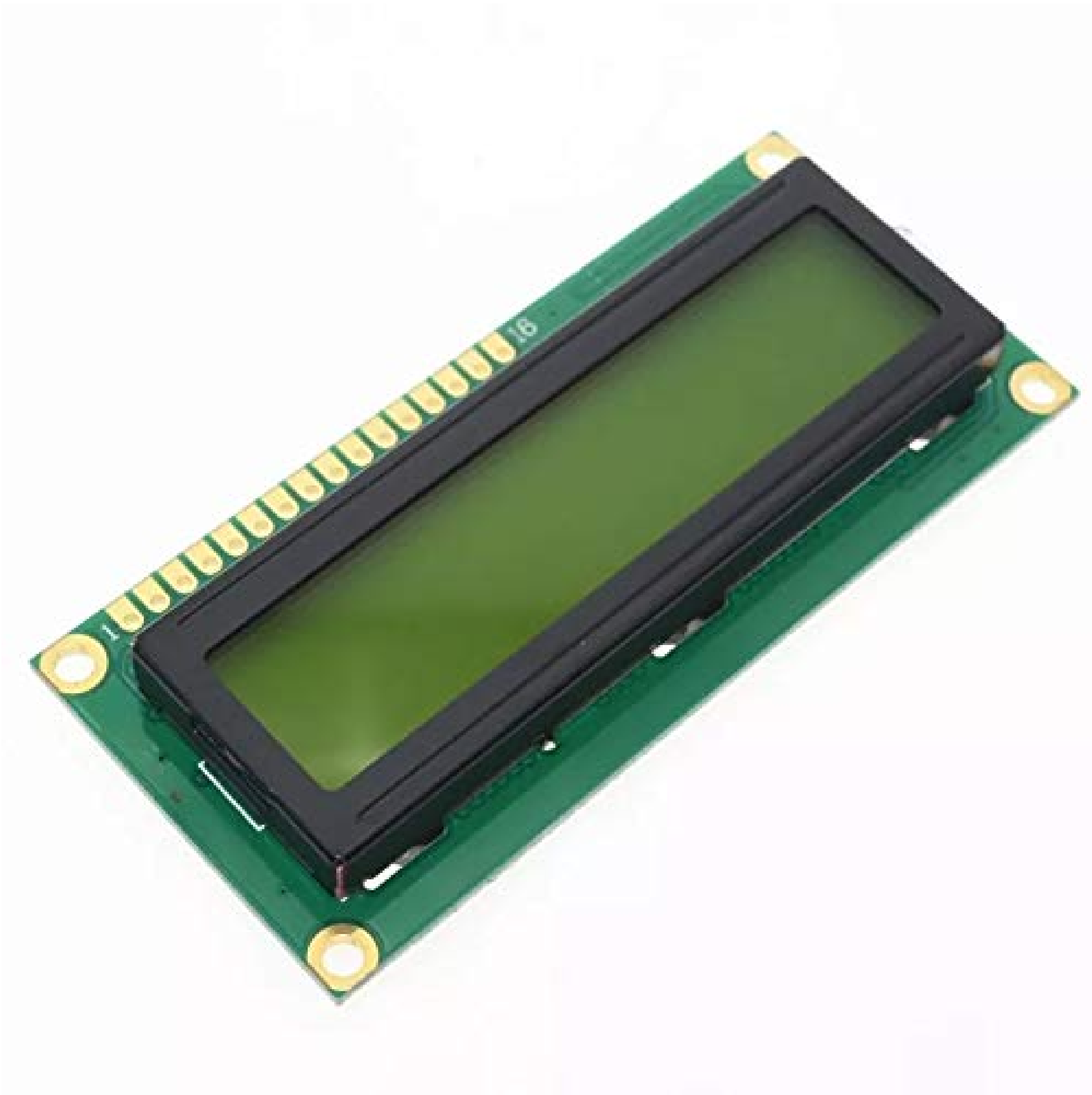

Figure 2.24: 16x2 Bit LCD Display

Some information about pins of this LCD.

- **PIN 1:** This is a ground pin of LCD
- **PIN 2:** This is the supply voltage pin of LCD
- **PIN 3:** Adjusts the contrast of the LCD.
- **PIN 4:** Toggles between Command/Data Register
- **PIN 5:** Toggles the LCD between Read/Write Operation
- **PIN 6:** Must be held high to perform Read/Write Operation
- **PIN 7 to 14:** Pins used to send Command or data to the LCD.
- **PIN 15:** Normal LED like operation to illuminate the LCD
- **PIN 16:** Normal LED like operation to illuminate the LCD connected with GND.

Now we learn how to connect it with Arduino Uno Board with the help of just a diagram.

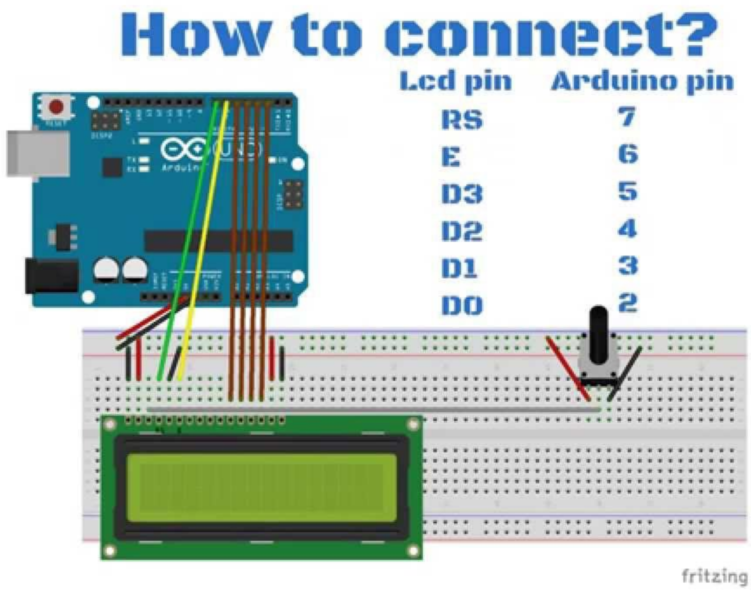

Figure 2.25: Connection of 16x2 Bit LCD Display with Arduino Uno Board

2.2.4 8-BIT DAC (DIGITAL TO ANALOG CONVERTER) LADDER

The R-2R ladder DAC is an 8-bit Digital to Analog Converter. An 8-bit value from the micro-controller port will be converted into an equivalent analog voltage with the help of the R-2R resistor network. The R-2R ladder DAC makes use of only two resistor values, but it requires 2N resistors so that it can functions N-bit conversion. The resistors values are in the ratio of 1:2. The R-2R DAC board output is provided with the Operational Amplifier which is used as a buffer in order to drive loads with current consumption of up to some milli-Amperes.

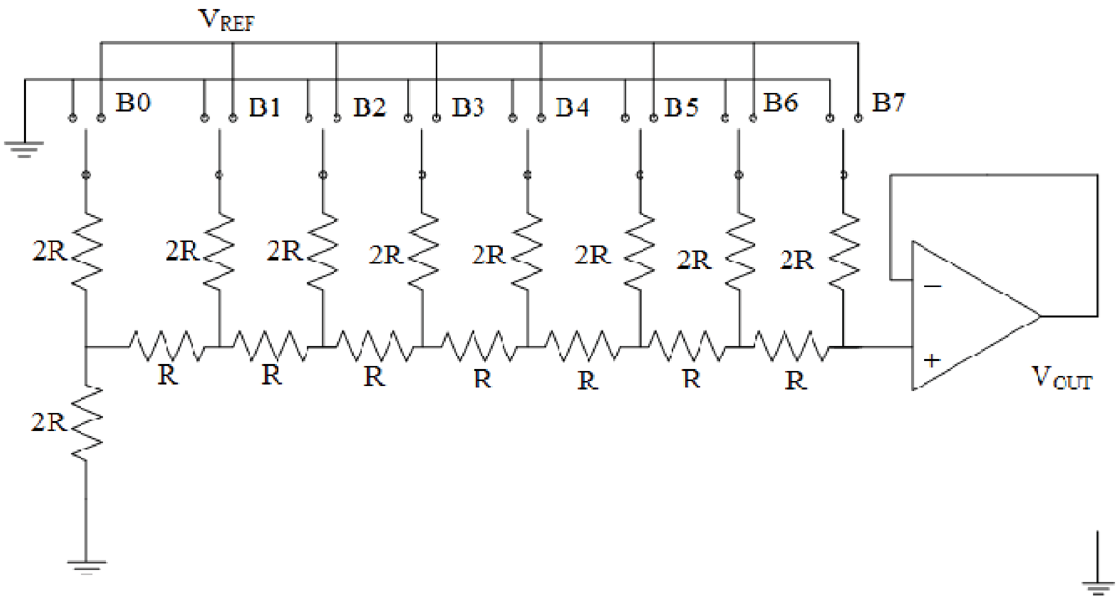

Figure 2.26: 8-bit DAC ladder with a buffer Op-Amp

And since we know that how it works, we will recall from our previous semester that for a n-bit ladder we have V_{Output} is:

$$V_{output} = \frac{V_{b0}}{2^8} + \frac{V_{b1}}{2^7} + \frac{V_{b2}}{2^6} + \frac{V_{b3}}{2^5} + \frac{V_{b4}}{2^4} + \frac{V_{b5}}{2^3} + \frac{V_{b6}}{2^2} + \frac{V_{b7}}{2^1}$$

And in our scenario the Output voltage From 8-bit DAC ladder is:

$$V_{output} = \frac{V_{b0}}{2^8} + \frac{V_{b1}}{2^7} + \frac{V_{b2}}{2^6} + \frac{V_{b3}}{2^5} + \frac{V_{b4}}{2^4} + \frac{V_{b5}}{2^3} + \frac{V_{b6}}{2^2} + \frac{V_{b7}}{2^1}$$

Now same as before we will see how our 8-bit DAC ladder is connected with our Arduino Uno Board using a diagram.

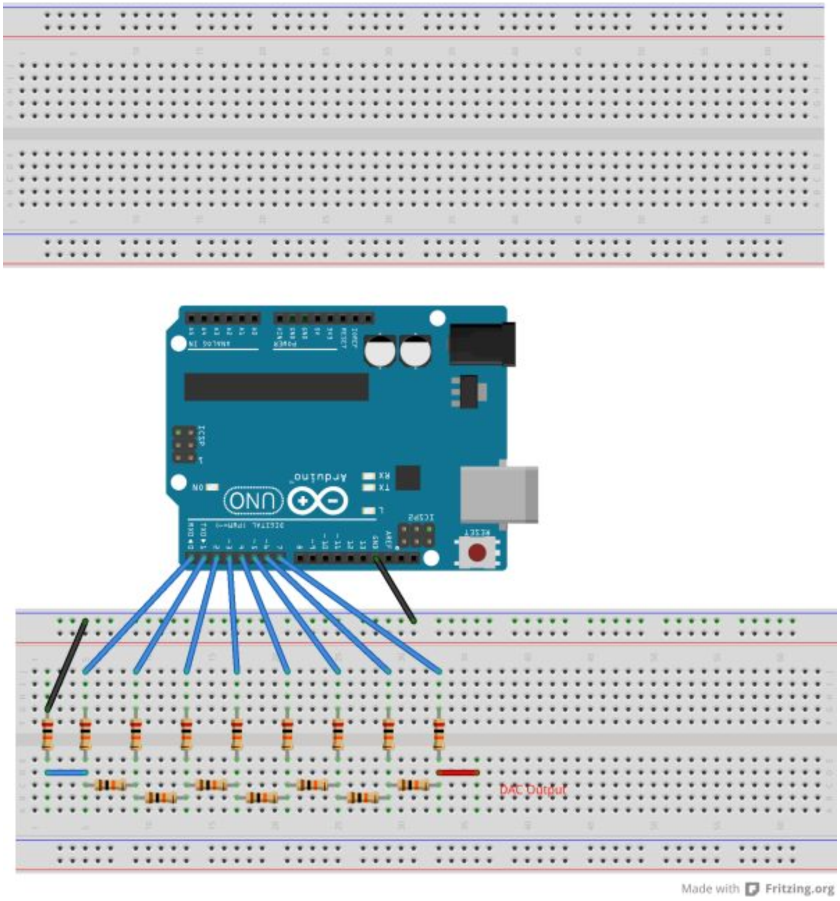

Figure 2.27: Connection of 8-Bit DAC Ladder with Arduino Uno Board

2.2.5 Varying Frequency Circuit

For this we have simple variable resistor circuit shown below:

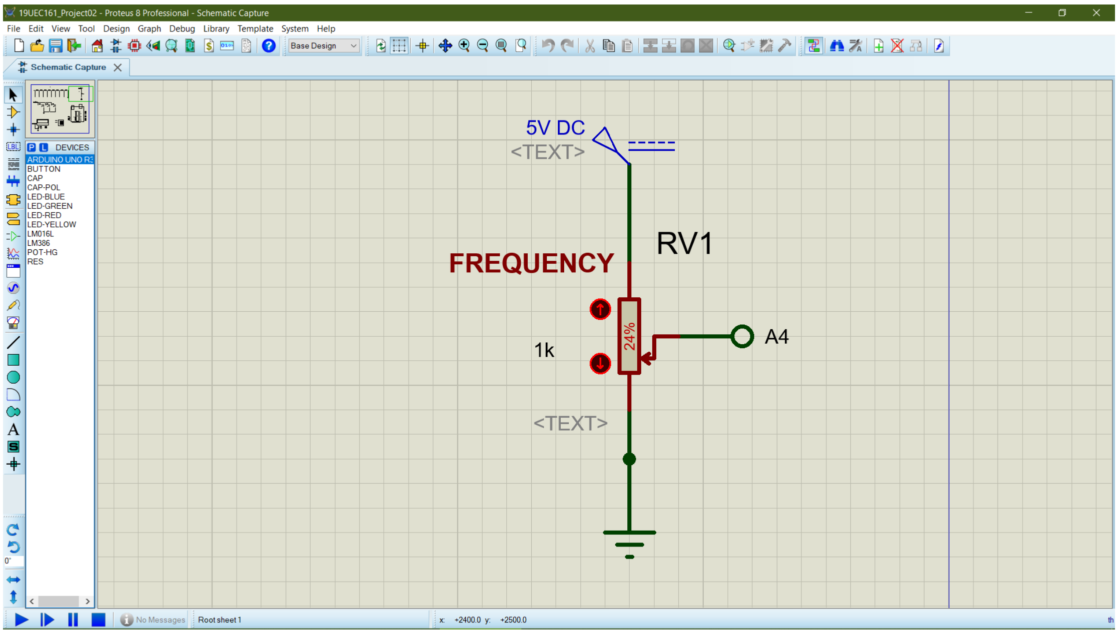

Figure 2.28: Varying Frequency CIrcuit

2.2.6 Button Switch Circuit

As done in previous part we have similarly the circuit of Button Switch shown below:

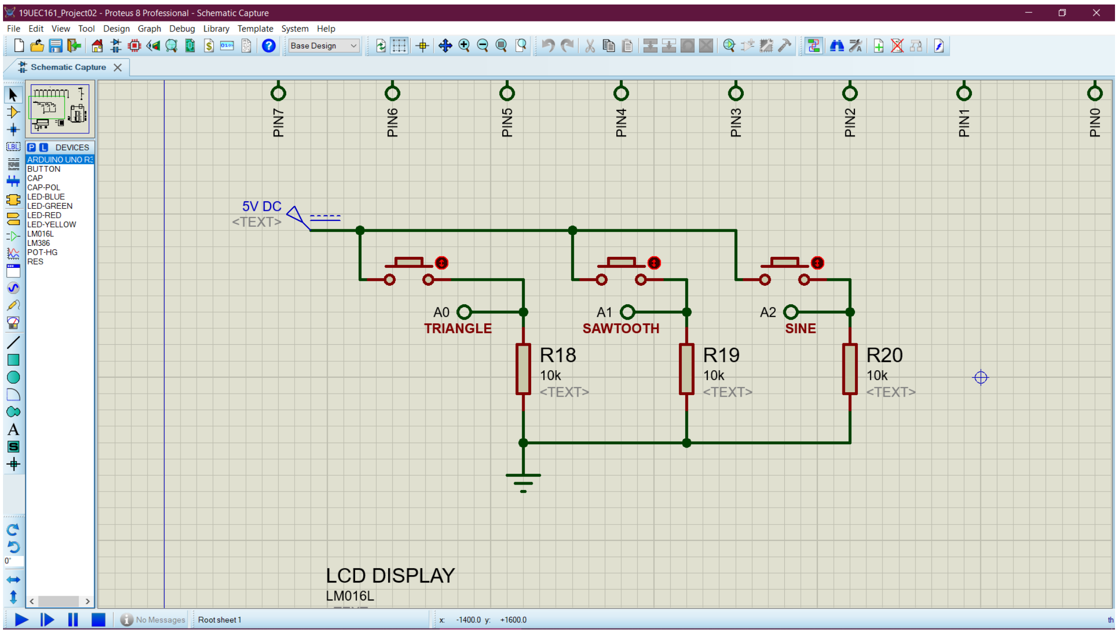

Figure 2.29: Circuit of Button Switch

2.2.7 Whole Circuit

Now we combined all of these components so that we can move one step ahead in making of Function Generator using Arduino Uno Board.

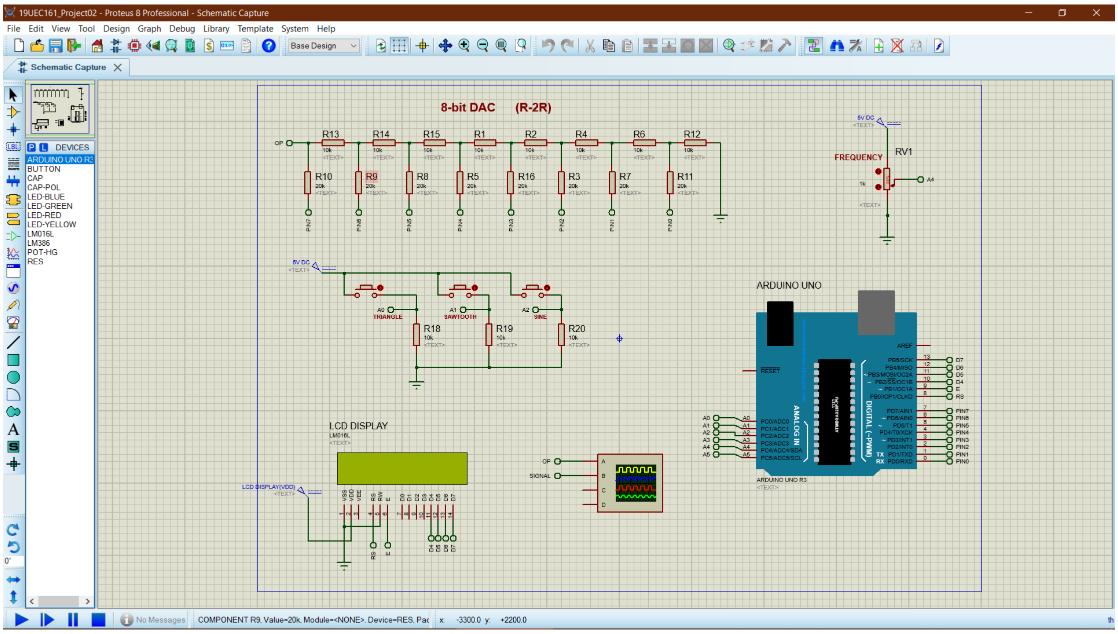

Figure 2.30: Whole Circuit of Function Generator using Arduino Uno Board

2.2.8 Code in Arduino Uno IDE

Since we have now our circuit ready therefore we will now write the code of Function Generator in Arduino Uno IDE.
Our Code will be:

```
// Library and Pins for the Display 16x2
#include <LiquidCrystal.h>
LiquidCrystal lcd(8, 9, 10, 11, 12, 13);

// Definitios of global variables
int sine[255];
const int POT = A0;
int value = 0;
int freq = 0;

int data1 = 0;
int data2 = 0;
int data3 = 0;

//variables for frequency pot monitoring
int frequency;
int freqCurrent;
unsigned int freqscaled;

void setup() {

  lcd.begin(16,2);
  lcd.print ("19UEC161");
  lcd.setCursor(0,1);
  lcd.print ("DL PROJECT 02");

  //  Set port/pin mode
  DDRD = 0xFF;          // all outputs PINS 0-7
```

```
  DDRC = 0x00;          // all inputs PINS A0-A5
  DDRB = 0xFF;          // all outputs PINS 8-13

  //  Initialize variables
  frequency = analogRead(A4);        //  initialize frequency
  freqscaled = 48*frequency+1;       //  from 1 to 48,001\
  period = samplerate/freqscaled;

  delay (3000);                      // So we can see the nice splash screen
  lcd.setCursor(0,0);
  lcd.print("Wave:             ");
  lcd.setCursor(0,1);
  lcd.print("     kHz          ");


  // Generate the date of a Sine function
  float x,y;
  for(int i=0;i<255;i++){
    x=(float)i;
    y=sin((x/255)*2*PI);
    sine[i]=int(y*128)+128;
    }
}


/*
 * @Description:  This function generate a Sine signal
 * @input:        freq
```

```
 */
void Sine_Function(int freq){

  for (int i=0; i<255; i++){
    PORTD = sine[i];
    delayMicroseconds(freq);
    }
}


/*
 * @Description:  This function generate a Triangle signal
 */

void Triangle_Function(int freq){

  for (int i=0; i<255; i++){
    PORTD = i;
    delayMicroseconds(freq/10);
    //delay(FREQ/100);
    }

  for (int i=255; i>0; i--){
    PORTD = i;
    delayMicroseconds(freq/10);
    //delay(FREQ/100);
    }
}
```

```
/*
 * @Description:  This function generate a saw signal
 */

void Sawtooth_Function(int freq){

    for (int i=0; i<255; i++){
    PORTD = i;
    delayMicroseconds(freq/10);
    }
}

/*
 * @Description:  This function check the value of the input Analog 4 (A4),
 *                which configure the frequency of the signal.
 *                This value will be displayed by the display
 */
void checkFreq() {
  freqCurrent = analogRead(A4);
  lcd.setCursor(0,1);
  lcd.print(1-(freqCurrent/1000.000));
}


/*
 * @Description:  This function check the value of the input Analog 0 (A0) until A3,
 *                to determined the signal to generate.
```

```
 *                  The name of the signal will be displayed by the display
 */

void checkShape() {

  if (digitalRead(A0)==HIGH){
    lcd.setCursor(5,0);
    lcd.print(" Triangle  ");
  }
  else if (digitalRead(A1)==HIGH){
    lcd.setCursor(5,0);
    lcd.print(" Saw       ");
  }
  else if (digitalRead(A2)==HIGH){
    lcd.setCursor(5,0);
    lcd.print(" Sine      ");
  }
  else if (digitalRead(A0)==LOW & digitalRead(A1)==LOW & digitalRead(A2)==LOW){
    lcd.setCursor(5,0);
    lcd.print("             ");
  }

}


/*****************************************************/

void loop() {
```

```
  value = analogRead(A4);
  freq = value*10;

  data1 = analogRead(A0);
  data2 = analogRead(A1);
  data3 = analogRead(A2);

  checkFreq();
  checkShape();

  if(data1>0){
    Triangle_Function(freq);
  }
  else if(data2>0){
    Sawtooth_Function(freq);
  }
  else if(data3>0){
    Sine_Function(freq);
  }
  else if(data1<0 & data2<0 & data3<0){
    }

}

/* End */
```

2.2.9 Explanation of above the Arduino Uno Code

In this code first of all I included LiquidCrystal.h in library to access some more commmands, then we use command LiquidCrystal lcd(8, 9, 10, 11, 12, 13) which tells us which Pins of **Arduino Uno** are connected in our LCD.
Then we define our required variables like array of sine[255], and const int to A0 for read only purpose i.e, its value cannot be changed, then in void setup we define PinModes as Input or Output.
After this we begin with our 8-Bit 16x2 LCD Display to function, then we print the letters on LCD display like in this code I write firstly my roll number as you have see in code using (coloumn, row) format.
lcd.print command prints our Roll number at default position i.e, (0, 0), and then by using set.cursor our LCD prints the Subject name at a specific position.
After this I used **DDR (Data Direction Registers)** or port manipulation, we have written DDRB = 0xFF which means all Ports or Pins in 'B' will act as output, below this we have written DDRC = 0x00 where this means that in our 'C' all the Ports or Pins will act as input, similarly for DDRD = 0xFF we will have same conditions for 'D' port.

Note:
B port - Digital Pin 8 to 13.
C port - Analog Pin A0 to A5.
D port - Digital Pin 0 to 7.
Also Unsigned int stores only positive values or 2-bit value having range from 0 to 65535 (2^{16}-1).

Then after this we used AnalogRead() to read a specific value from the analog pins and initializes variables. After all of these we delay(3000) which delays or halts the program for 3000 ms after this we print wave and its frequency to display it on screen (in kHz).
(Actually our frequency value is A4 which is set to read only mode and taken from Rheostat's 3rd Pin from **Varying Frequency Circuit** i.e, 1 Ohm to 1000 Kilo-Ohm.

After we make the sine wave, and write **y=sin[x/255]*2*PI** where 'x' is float type, and 'x/255' is our sampling time period, actually here we are using sampling time property, then after this we write **sin[i]=int(y*128)+128**, the reason we write this line is because that our Sin function ranges from -1 to +1 but our Arduino Uno gives value from 0 to 255 bits or 0 to 5V, therefore to make -1 to +1 range to 0 to 255 bits we first multiply 'y' by 128 to make it -128 to +128 then we add 128 to it, so that it will finally become 0 to 255 range.

Now our Sin wave is ready and we then generate it by *for* loop and we use **PORTD** now what **PORTD** is do is to maps 'D' port of Arduino Uno.
Then we simply use delayMicroseconds() to delay the program in microseconds (us), this is only used when delays are shorter than 1000 us. After this I create trianglular wave by simply using an increasing loop (like x=y graph) and decreasing loop (like x= -y graph) using delays between them and at last taking output from digital Pins using **PORTD**.
Similarly in Sawtooth wave I use only increasing loop then repeat it from going 0 to 255 after every cycle to get Sawtooth wave.
Then we display frequency (variable) on screen which is taken from A4 Pin and we will get frequency (in kHz) on screen, we subtracted it by '1' as you can see in the code it is due to because we know that decreasing value of Rheostat will increase the Current value in A4, which results in high frequency wave, but since our input is directly that current to A4 due to which LCD screen will show increased frequency in wider waves and vice versa, therefore we have to make opposite of it, that's why we subtracted '1' by it to get the correct value.
Now we determine which signal we have to generate, by using simple if-else statements, I wrote this, that if we have A0 = HIGH then we will get Triangular Wave, if we have A1 = HIGH then we will get Sawtooth Wave, if we have A2 = HIGH then we will get Sin Wave and if none of them is HIGH then no wave is generated at output.
At last all of the Digital Pins are connected to our 8-Bit DAC ladder Circuit to produce analog waves since our final waves are analog in nature, also Arduino Uno gives 8-Bit binary values for Pulse Width Modulation.

At last in void loop() we make this program to run continuosly using another if-else statement.

3 RESULTS

3.1 FROM OPERATIONAL AMPLIFIERS

Now our Function Generator circuit is completed so let's see its single output in oscilloscope. I mainly use the 4-Terminal Switch to get different outputs at different terminals. In this circuit we have generated 4 different kinds of waves: -

1. Square wave.
2. Triangular wave.
3. Sinusoidal wave.
4. Sawtooth wave.

Let's see the all outputs one by one.

3.1.1 ONE BY ONE OUTPUT

Output 1: - Square wave. (The red wire)

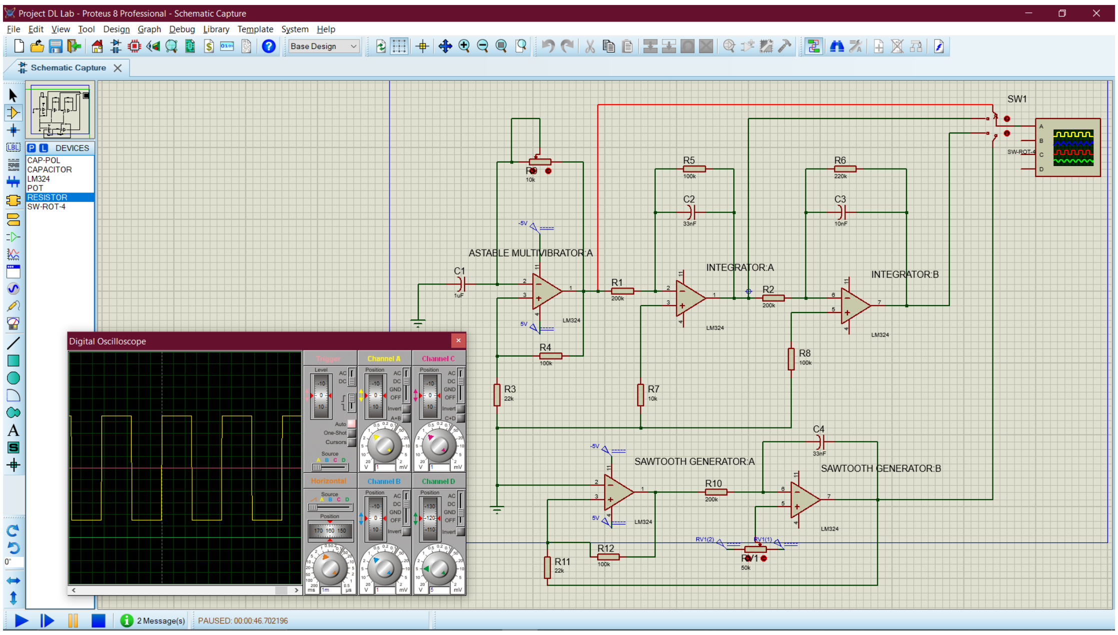

Figure 3.1: Square wave Output

Output 2: - Triangular wave. (The red wire)

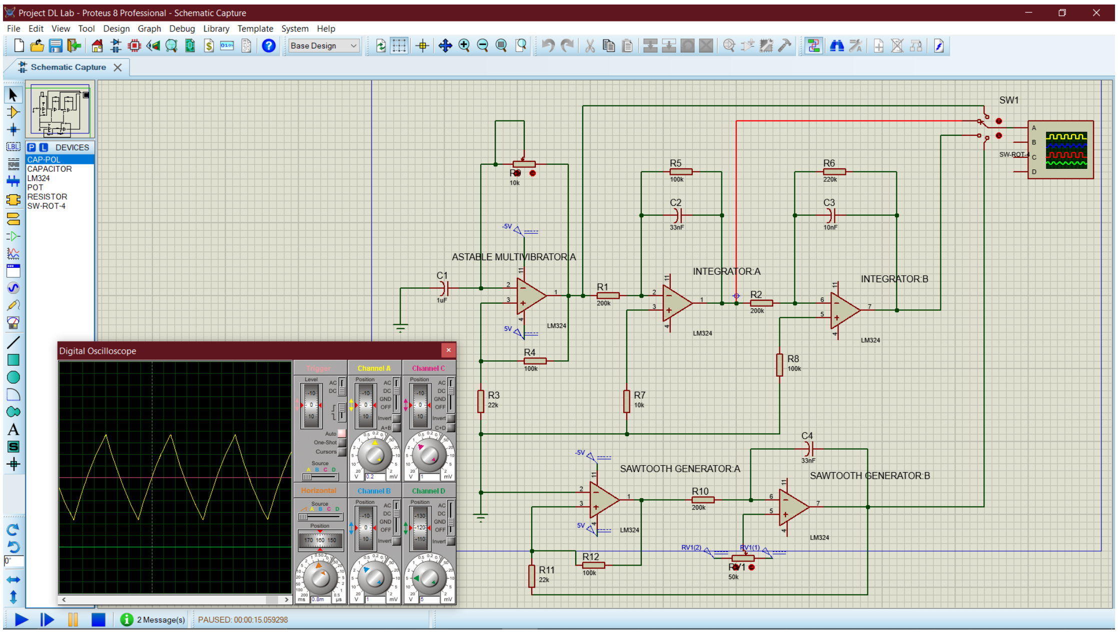

Figure 3.2: Triangular wave Output

Output 3: - Sinusoidal wave. (The red wire)

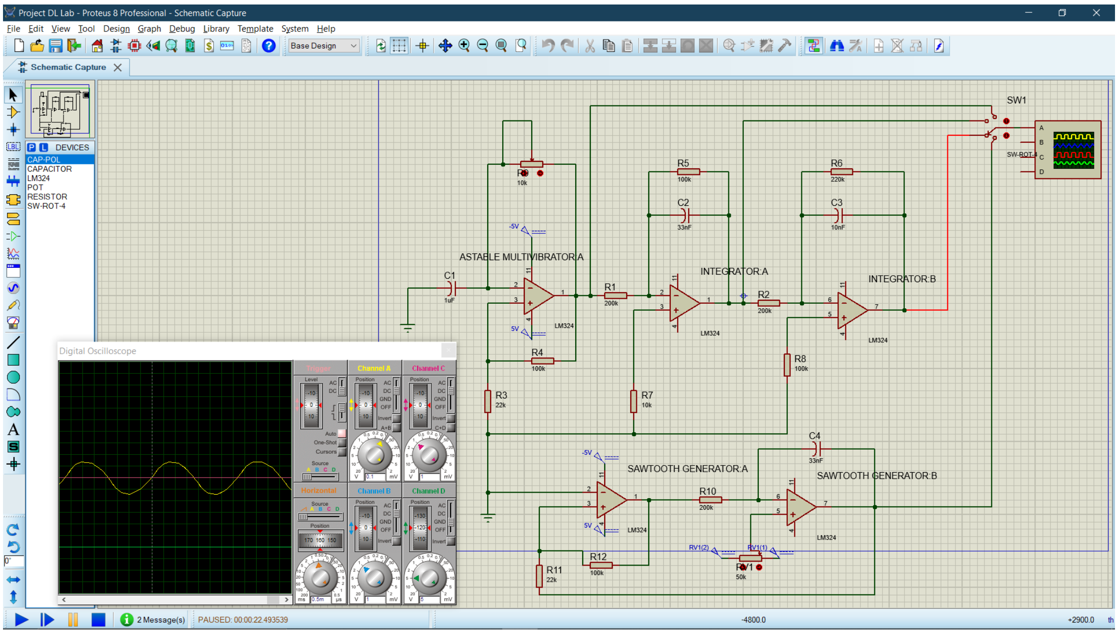

Figure 3.3: Sinusoidal wave Output

Output 4: - Sawtooth wave. (The red wire)

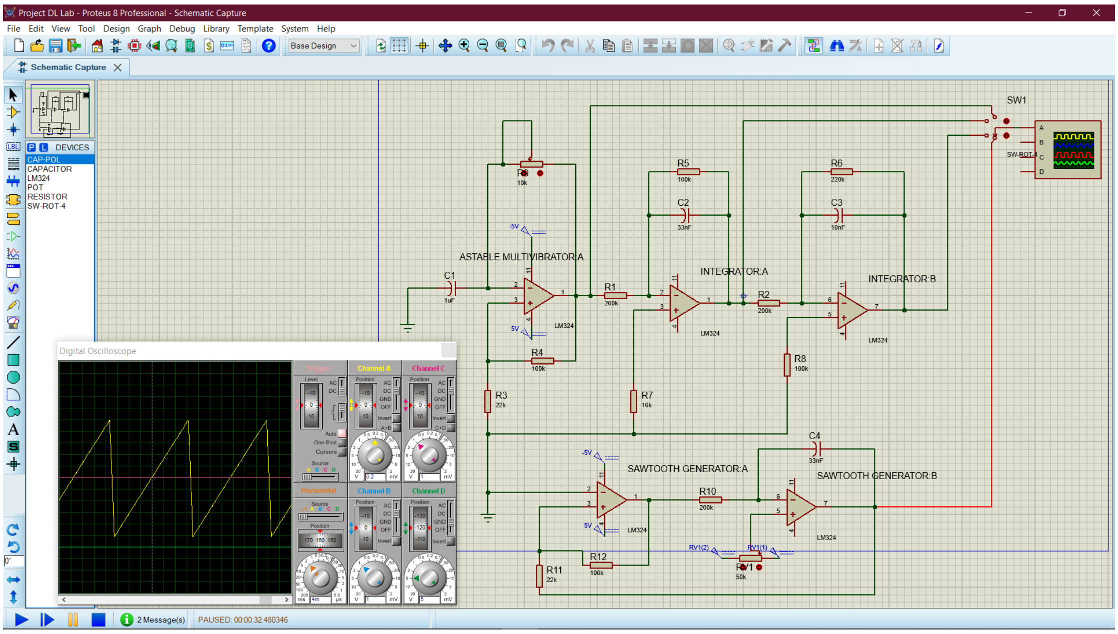

Figure 3.4: Sawtooth wave Output

Now let's see all the outputs together.

3.1.2 All Outputs together

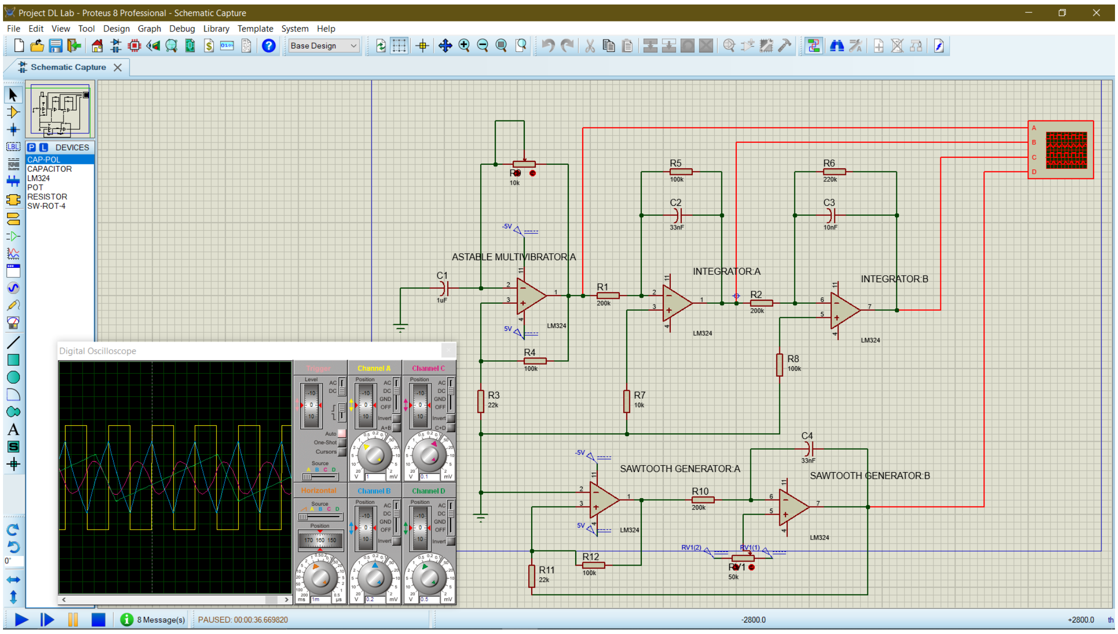

Figure 3.5: All Outputs Together

Let's take a closer look!

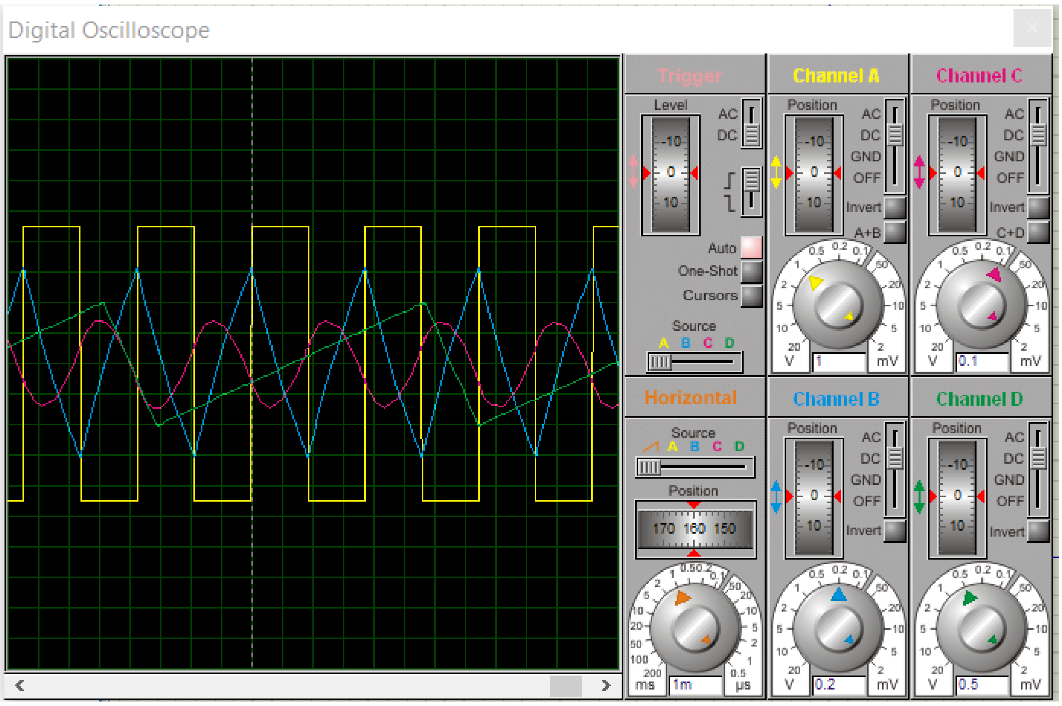

Figure 3.6: Closure look of all Outputs

These are all 4 waves which we produced above. All the diagrams which you see above starting from making of Function Generator circuit was made in the software used for Simulation of electronic circuits called ***Proteus 8 Professional.***

As we have already discussed that if we want more perfect results of any wave which we are getting, just make the time constant of the respective capacitors greater so that there will be less distortion and more perfection, or we can also vary the potentiometer in first LM324 (Astable Multivibrator).

We have also observed that if we increase resistances which are between cascaded LM324's, amount of distortion gets reduced.

3.1.3 Distortion Case

Now we will show you what happens when we make the time constant smaller of respective LM324's or the increasing resistance of potentiometer in Astable Multivibrator or decreasing the resistances which are between cascaded LM324's.

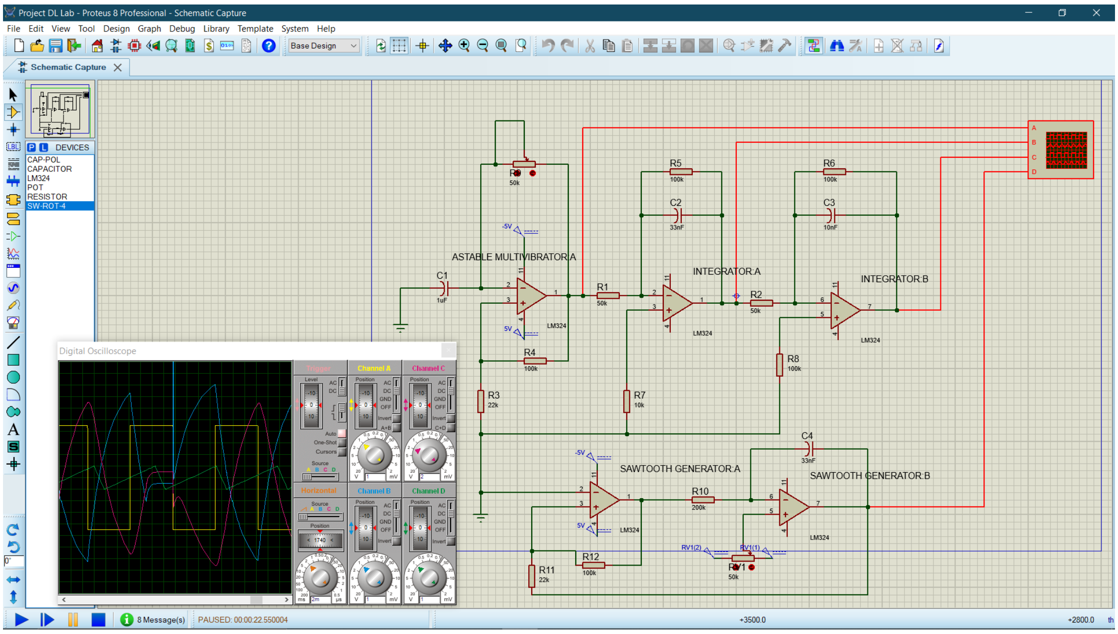

Figure 3.7: Distorted Outputs

Let’s take the closer look of this also!

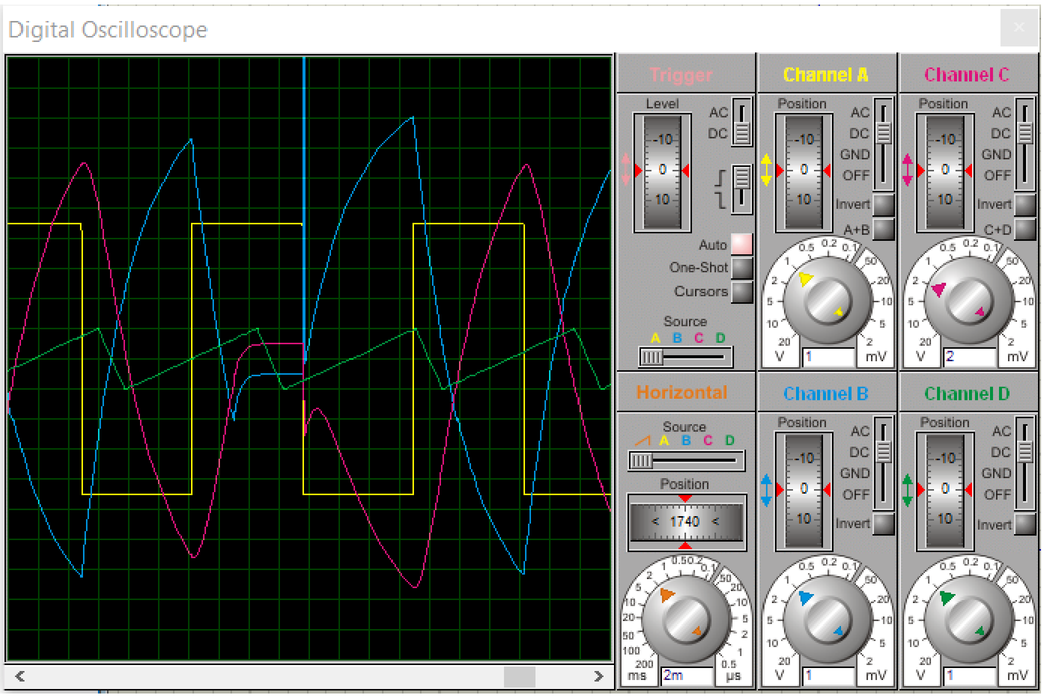

Figure 3.8: Closer look of Distorted Outputs

I also faced many significant problems during this project.

1. I started running into significant issues. Getting the design of the integrator block was very tricky and even when I finally saw a sine wave output, getting this amplitude to remain relatively constant without sacrificing too much of the sine wave quality was the most significant design challenge I faced.
2. The packaging facet of the project was also very difficult. I had trouble in finding ways to attach my potentiometers to the LM324’s.

3.2 FROM ARDUINO UNO

3.2.1 SOME RUNNING SHOTS FROM OVERALL CIRCUIT

Now we will have some results shots from our simulation software ***Proteus 8 Professional***

1st Output: Sinusoidal Wave having Frequency 0.3 kHz.

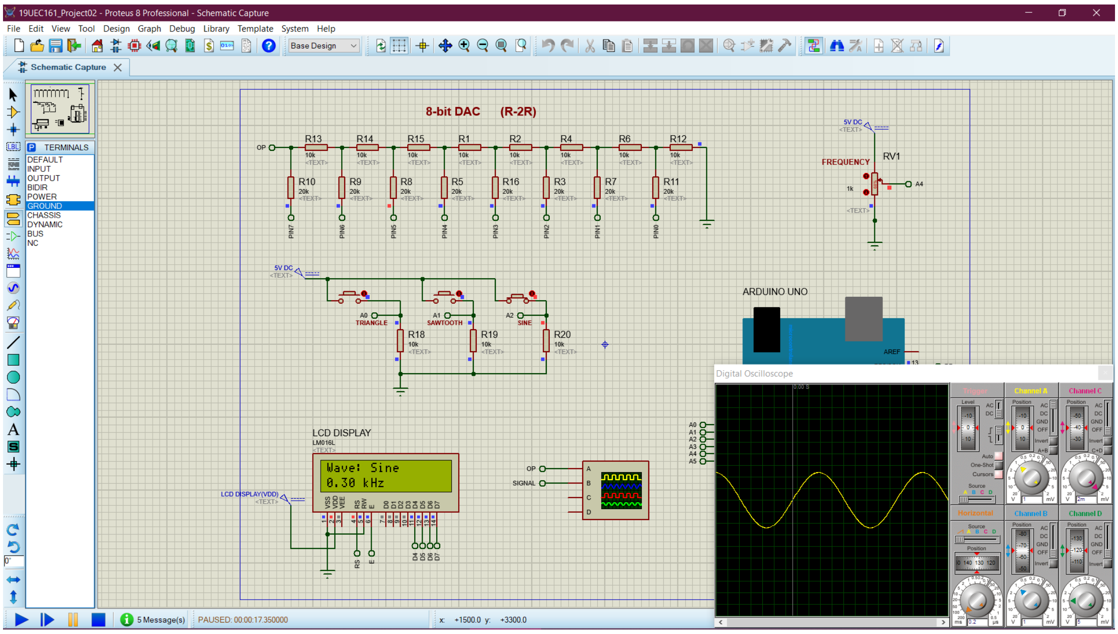

Figure 3.9: Sinusoidal Wave of 0.3 kHz

2nd Output: Sinusoidal Wave having Frequency 0.6 kHz.

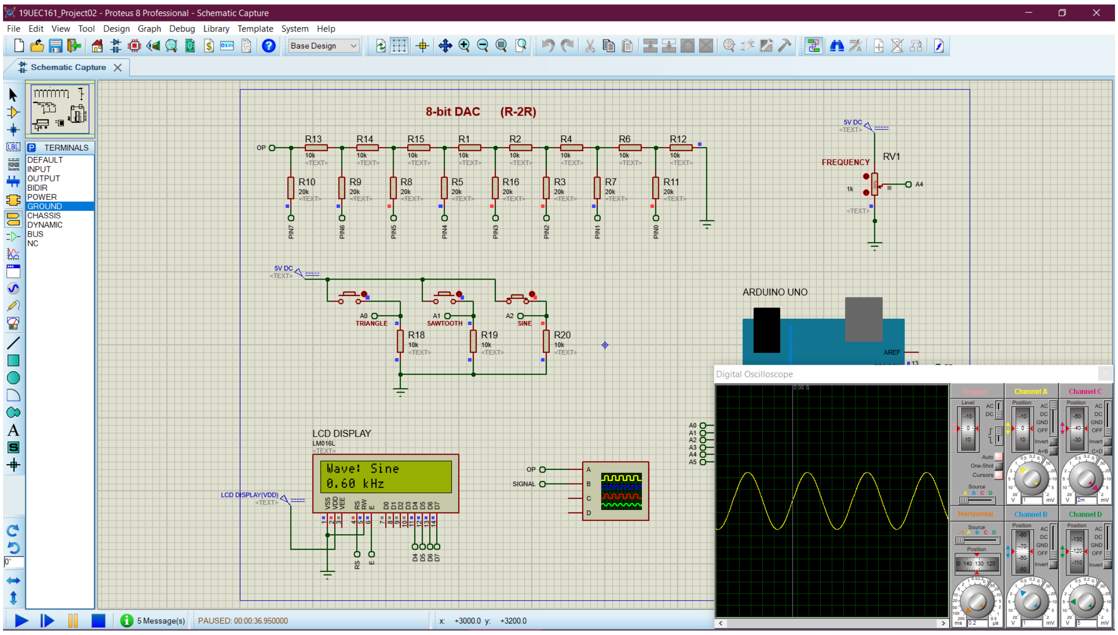

Figure 3.10: Sinusoidal Wave of 0.6 kHz

You will see that Sine Button is **"ON"** here in the ***Proteus 8 Professional***

3rd Output: Sawtooth Wave having Frequency 0.2 kHz.

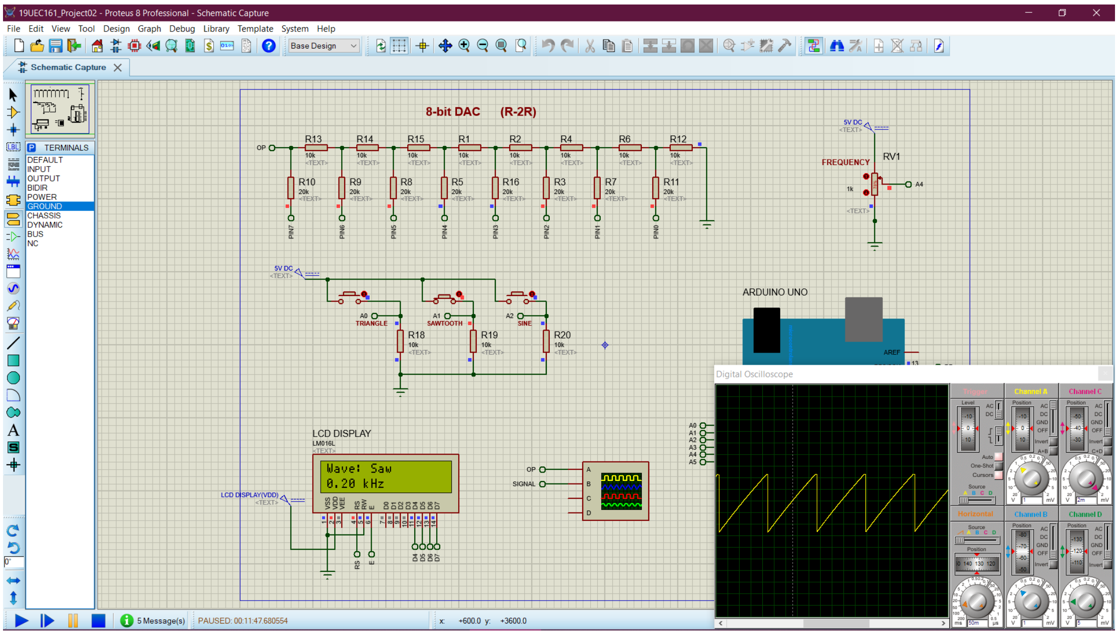

Figure 3.11: Sawtooth Wave of 0.2 kHz

4th Output: Sawtooth Wave having Frequency 0.8 kHz.

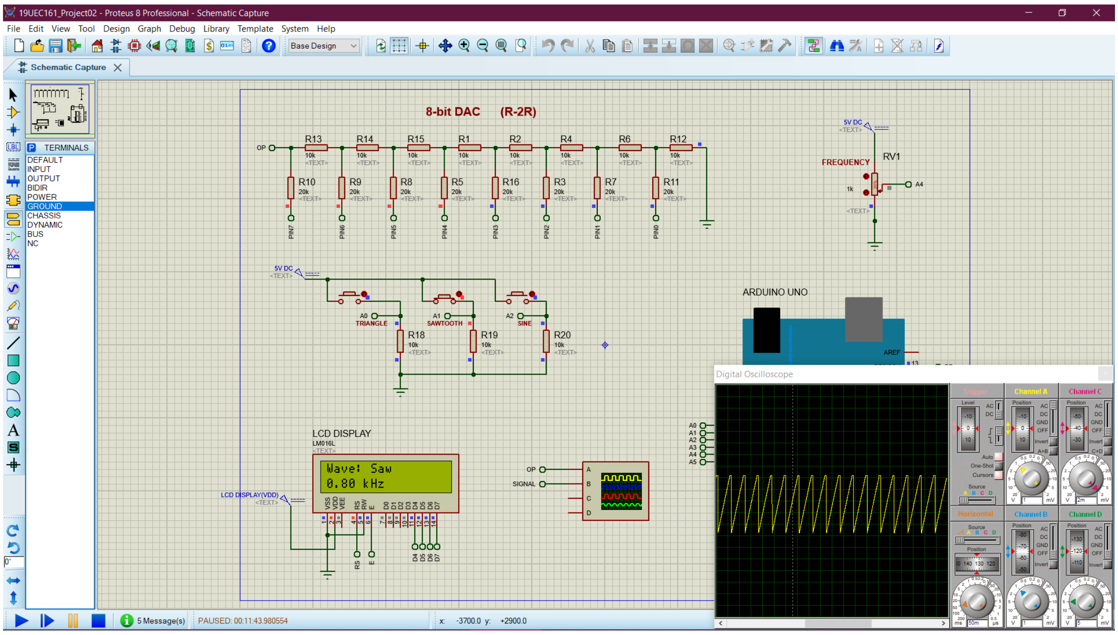

Figure 3.12: Sawtooth Wave of 0.8 kHz

You will see that Sawtooth Button is **"ON"** here in the ***Proteus 8 Professional***

5th Output: Triangular Wave having Frequency 0.1 kHz.

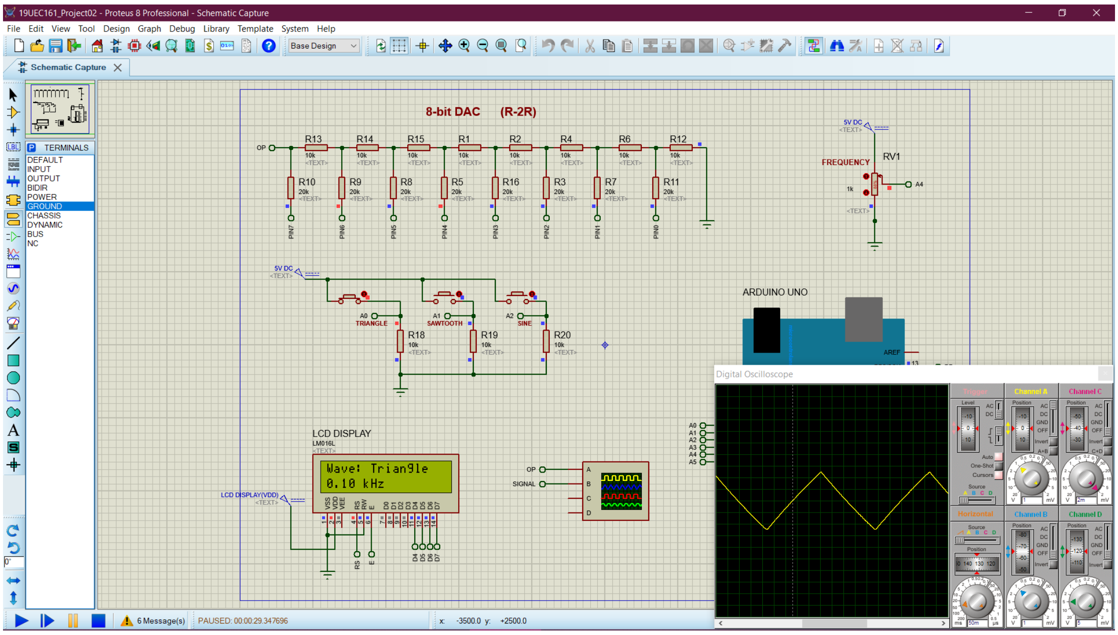

Figure 3.13: Triangular Wave of 0.1 kHz

6th Output: Triangular Wave having Frequency 0.9 kHz.

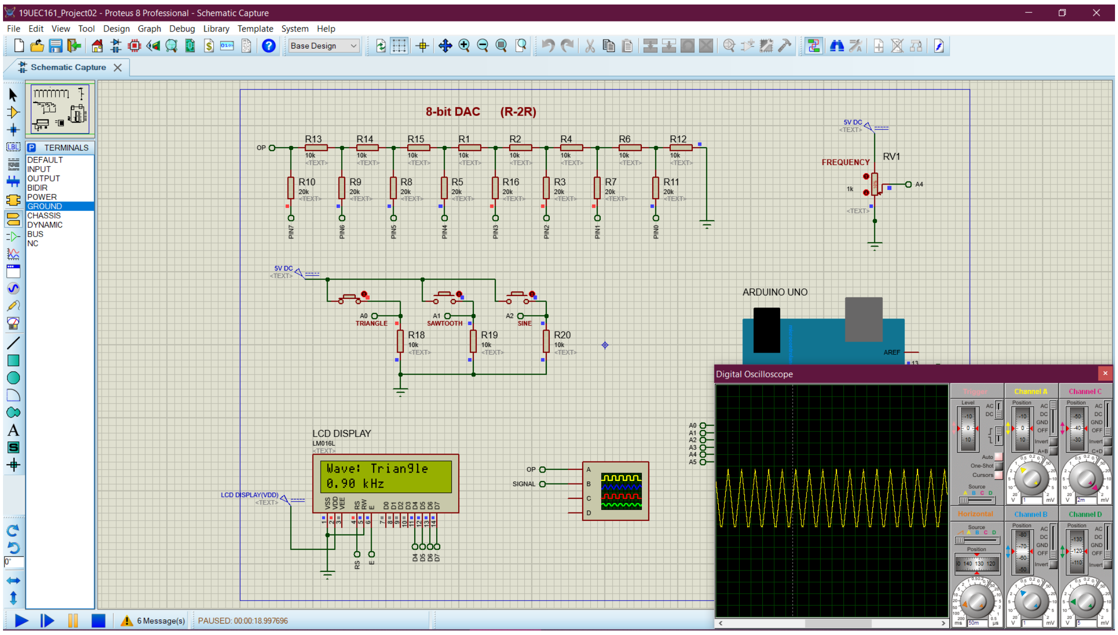

Figure 3.14: Triangular Wave of 0.9 kHz

You will see that Triangle Button is **"ON"** here in the ***Proteus 8 Professional***

Hence our Code works perfectly as you can saw different waves of different frequencies. Some difficulties which I have faced in Arduino Uno were:

1. I think to write the code of each type of wave roughly then join them at last or combine the code but it is not easy according to my opinion, because it didn't worked for me in starting but somehow I managed it.
2. I did too many mistakes related with syntax errors but thanks for saving me IDE compiler.
3. And at last there were some mistakes which I did and they were silly and I corrected it by revising the Code many times.

4 CONCLUSION AND FUTURE WORK

During the course of this project, I learned several things. Like I learned about the basics of the Function Generator, about LM324, revised the Integrator and Astable Multivibrator Circuits, 8-bit DAC ladder, and something about 16x2 Bit LCD Display, I learned about how to design various analog circuit blocks to produce specific wave-forms.
Also with use Arduino Uno, I develop a new programming skill, my software and hardware synced together perfectly which is a good thing for me. Due to having similar syntax like in C programming it becomes handy for me to work with Arduino Uno. I found the theory of how it all works beautiful - my circuit analysis classes finally came to life and helped me produce something meaningful. This project also helped me gain experience in making my work look nice and polished - it is a very useful skill to have and it is often neglected in engineering classes. Within the globe, it isn't enough for a product to figure. It needs to have aesthetic attractiveness - This is often why Apple is triple-crown. Other than learning about the importance of packaging, I learned many other things in this project.

1. Op-amps are incredibly powerful and it is very easy to chain them together without worrying about loading effects due to high input impedance.
2. Theory is nice, however in follow, you frequently ought to attempt things rather than thinking too arduous i.e, you often need to try things instead of thinking too hard. Experimenting with different values for different components was very useful, especially when it came to the design of the integrator and compensating for its gain.
3. I found many new types of approach in Arduino Uno to write a Code of a Function Generator with many ways and many logics.
4. It is always a pleasant plan to keep a notebook. It helped me remember what I did and when, what values I used, and what I need to do next time. It also helped me write this report!

There are also many improvements that could be made to this project.

1. By changing the potentiometer set up on the LM324's or the value of capacitor, we can easily get more perfect waves.
2. By varying the resistor on first LM324 IC or the by varying the resistances between the cascaded LM324's or by changing the time constants of respective LM324's capacitors, again we can get more perfect waves.

3. The quality of the sine wave may be improved by some additional through some careful experimentation.
4. It also would be nice to form a far better interface for connecting to the our Function generator than exposing 2 pins.
5. It will be more perfect if I increase the range of varying frequency to get vast range in Arduino Uno by increasing the resistance of Rheostat in Frequency Varying Circuit.
6. We can also add many other types of waves in Arduino Uno to make more vast.

Reference from our honourable teachers [1]

Diagram of Function Generator was taken from [2]

Waves use in my outline were taken from [3]

For understanding the working of Astable Multivibrator I read it from [4]

Similarly for understanding the working of Op-Amp Integrator I read it from [5]

For getting basic idea about LM324 I read it from [6]

For more detailed knowledge of LM324 go to [7]

For more details regarding connection of LCD with Arduino Uno go to [8]

For getting more detailed knowledge of how Pins of Arduino Uno worked go to [9]

REFERENCES

[1] Sandeep Saini, Umang Sehgal, and Vineet Sahula. Relative clause based text simplification for improved english to hindi translation. In *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, pages 1479–1484. IEEE, 2015.

[2] Tgf4042 arbitrary function generator, 2x 40mhz aim-tti. https://www.distrelec.biz/en/arbitrary-function-generator-2x-40mhz-aim-tti-tgf4042/p/30143711. (Accessed on 09/14/2020).

[3] Types of sound waves | thedawstudio.com. https://www.thedawstudio.com/types-of-sound-waves/. (Accessed on 09/14/2020).

[4] Astable multivibrator using op-amp | circuit, operation and analysis – eeebooks4u. https://eeebooks4u.wordpress.com/2017/03/25/astable-multivibrator-using-op-amp/. (Accessed on 09/15/2020).

[5] Op-amp integrator, operational amplifier integrator. https://www.electronics-tutorials.ws/opamp/opamp_6.html. (Accessed on 09/15/2020).

[6] Introduction to lm324 - the engineering projects. https://www.theengineeringprojects.com/2017/06/introduction-to-lm324.html. (Accessed on 09/15/2020).

[7] Lm324 comparator ic circuit working and its applications. https://www.watelectronics.com/lm324-comparator-ic-working-applications/. (Accessed on 09/15/2020).

[8] Arduino lcd tutorial | how to connect an lcd to arduino. https://howtomechatronics.com/tutorials/arduino/lcd-tutorial/. (Accessed on 11/01/2020).

[9] Arduino pin configuration: A detailed guide (2020) | robu.in. https://robu.in/arduino-pin-configuration/. (Accessed on 11/01/2020).