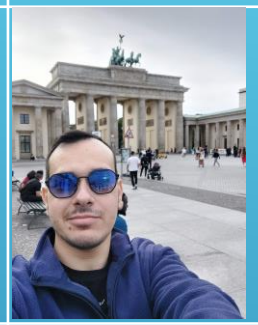
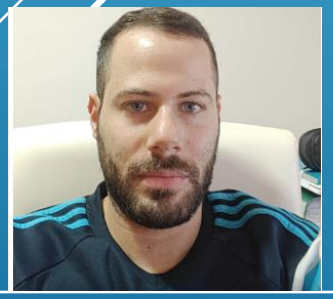
Electronic Instrumentation

Students Micro-Project 2-GROUP NUMBER #6

	ID	Name	Photo of the student
Student #1	**316352210**	**Omri pony**	
Student #2	**316146836**	**Steve Reznikov**	
Student #3	**313601866**	**Or Salganik**	

To be submitted until 7/5/2024 at 23:59

https://www.tinkercad.com/things/euZzRB0egxH-mp2finalforsubmission

Lecturer Dr. Samuel Kosolapov

STRATEGY TO FIND THE AC DIFFERENTIAL VOLTAGE – TINKERCAD AND MULTISIM – INTRODUCTION TO P22 AND THE ENTIRE PROJECT – IMPORTANT ANNOUNCEMENT

We decided to work in two paths – both in TinkerCad (and Arduino) as necessary but to simulate also in Multisim

TinkerCad simulation will use the Arduino Uno through 2 analogical inputs (A0 and A5) and will use the MAP function to calculate the voltage and print it on the serial monitor (sample using a push button if pressed)

Multisim simulation that will have more 741 components to simulate a simpler multimeter (in case that we don't have Arduino next to us and we want to evaluate the AC differential voltage)

INTRODUCTION - WHAT IS AN INSTRUMENTATION AMPLIFIER? (PART 1)

(PREPARATION TO P21) [2]

Instrumentation Amplifier is a **voltage – gain device** that amplifies the difference between the voltage existing at its two input terminals with additional input buffer stages.

In other words, we can say that a differential operational amplifier circuit that provides high input impedance with ease of gain adjustment through the variation of a single resistor, is called **instrumentation amplifier**.

The input of an instrumentation amplifier is always the output from a transducer - that is very small signal.
So, the main purpose of an instrumentation amplifier is to amplify small signals.

INTRODUCTION - WHAT IS AN INSTRUMENTATION AMPLIFIER? (PART 2)

(PREPARATION TO P21)

The two non-inverting amplifiers form a differential input stage acting as buffer amplifiers with a gain of $(1 + \frac{2R_2}{R_1})$ for differential input signals and unity gain for common mode input signals.

Since amplifiers A_1 and A_2 are closed loop negative feedback amplifiers, we can expect the voltage at V_a to be equal to the input voltage V_1. Likewise, the voltage at V_b to be equal to the value at V_2.

As the op-amps take no current at their input terminals (virtual earth), the same current must flow through the three-resistor network of R_2, R_1 and R_2 connected across the op-amp outputs.
This means then that the voltage on the upper end of R_1 will be equal to V_1 and the voltage at the lower end of R_1 to be equal to V_2.

This produces a voltage drop across resistor R_1 which is equal to the voltage difference between inputs V_1 and V_2, the differential input voltage, because the voltage at the summing junction of each amplifier, V_a and V_b is equal to the voltage applied to its positive inputs.

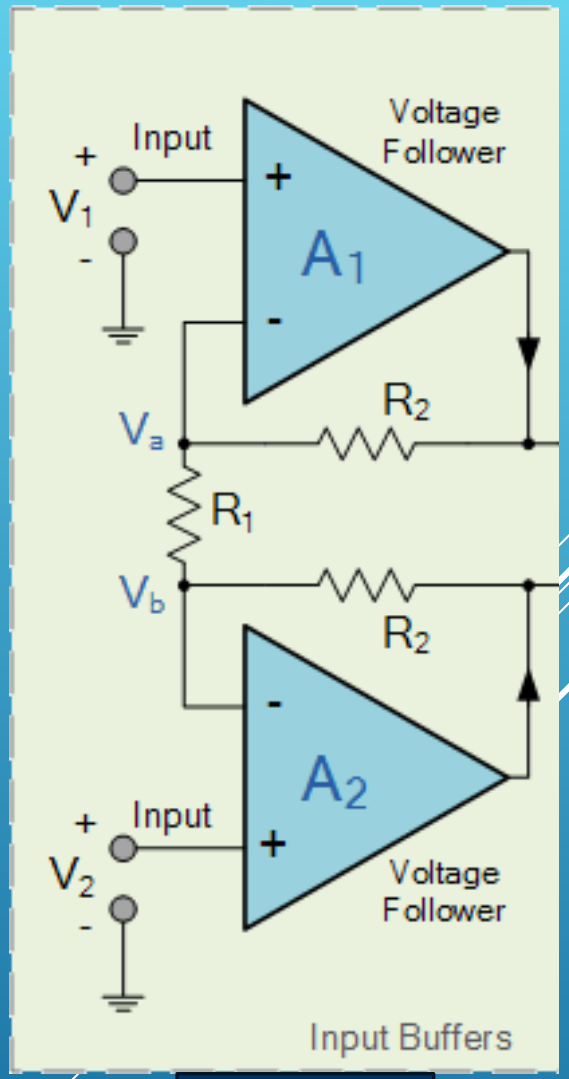


Figure 1

https://www.electronics-tutorials.ws/opamp/opamp_lmth.5

INTRODUCTION - WHAT IS AN INSTRUMENTATION AMPLIFIER? (PART 3)

(PREPARATION TO P21)

However, if a common-mode voltage is applied to the amplifier's inputs, the voltages on each side of R_1 will be equal, and no current will flow through this resistor.
Since no current flows through R_1 (nor, therefore, through both R_2 resistors, amplifiers A_1 and A_2 will operate as unity-gain followers (buffers).
Since the input voltage at the outputs of amplifiers A_1 and A_2 appears differentially across the three-resistor network, the differential gain of the circuit can be varied by just changing the value of R_1.

The voltage output from the differential op-amp A_3 acting as a subtractor, is simply the difference between its two inputs ($V_2 - V_1$) and which is amplified by the gain of A_3 which may be one, unity, (assume $R_3 = R_4$):

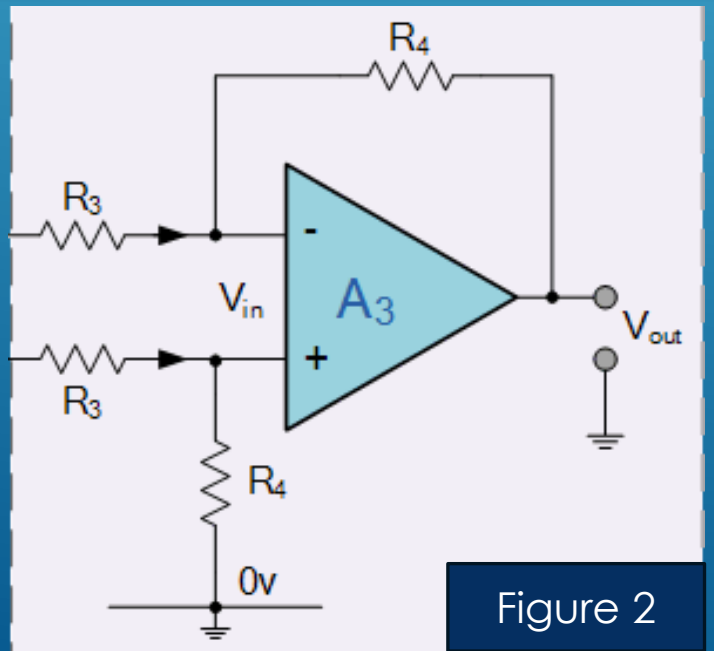


Figure 2

The differential amplifier (in figure 2) is a voltage subtractor circuit which produces an output voltage proportional to the voltage difference of two input signals applied to the inputs of the inverting and non-inverting terminals of an operational amplifier.

https://www.electronics-tutorials.ws/opamp/opamp_lmth.5

INTRODUCTION - WHAT IS AN INSTRUMENTATION AMPLIFIER? (PART 4)

(PREPARATION TO P21)

Then we have a general expression for overall voltage gain of the instrumentation amplifier circuit as:

$$V_{OUT} = (V_2 - V_1)\left[1 + \frac{2R_2}{R_1}\right]\left(\frac{R_4}{R_3}\right)$$

Advantages of using Instrumentation Amplifier:

1. The input impedance is very high due to the emitter follower configurations of amplifiers A_1 and A_2.
2. The output impedance is very low due to the difference amplifier A_3.
3. The CMRR of A_3 is very high and almost all the common mode signal will be rejected.
4. The gain of the three operational amplifiers instrumentation amplifier circuit can be easily varied by adjusting the value of only one resistor R_1 .

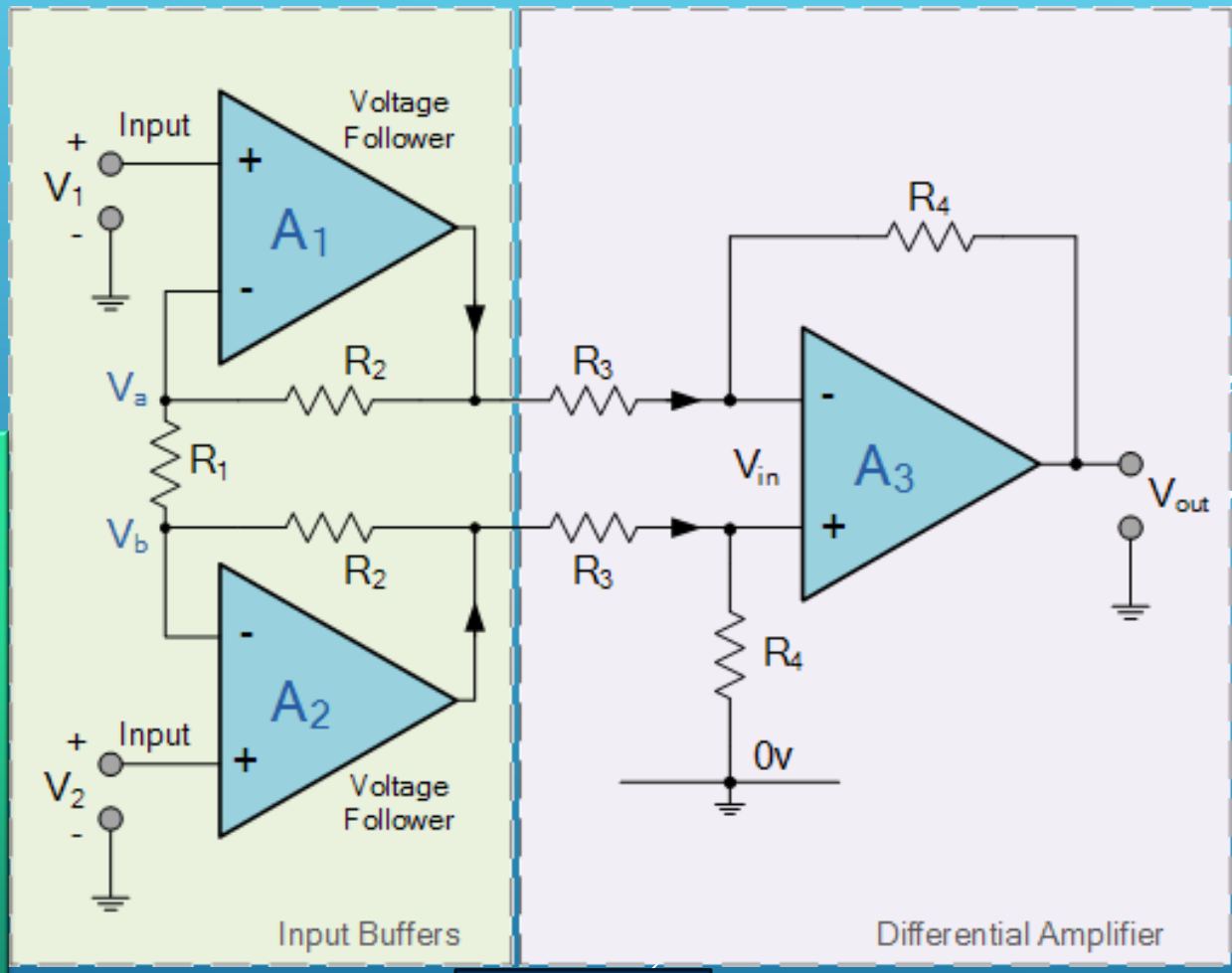


Figure 3

INTRODUCTION - WHAT IS THE NON INVERTING OPERATIONAL AMPLIFIER?

(PREPARATION TO P21)

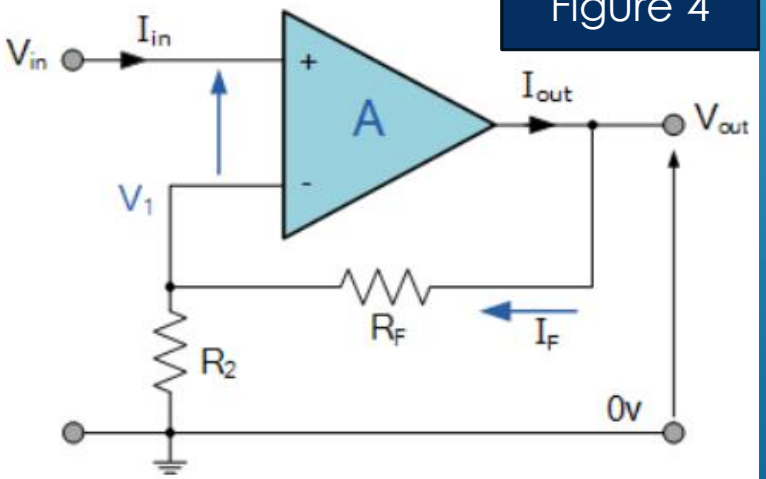
In non-inverting operational amplifier configuration, the input voltage signal V_{in} is applied directly to the non-inverting (+) input terminal which means that the output gain of the amplifier becomes "Positive" in value in contrast to the "Inverting Amplifier". The result of this is that the output signal is "in-phase" with the input signal.

Feedback control of the non-inverting operational amplifier is achieved by applying a small part of the output voltage signal back to the inverting (–) input terminal via a R_f – R_2 voltage divider network, again producing negative feedback.

This closed-loop configuration produces a non-inverting amplifier circuit with very good stability, a very high input impedance, R_{in} approaching infinity, as no current flows into the positive input terminal, (ideal conditions) and a low output impedance.

$$A_{(v)} = 1 + \frac{R_F}{R_2}$$

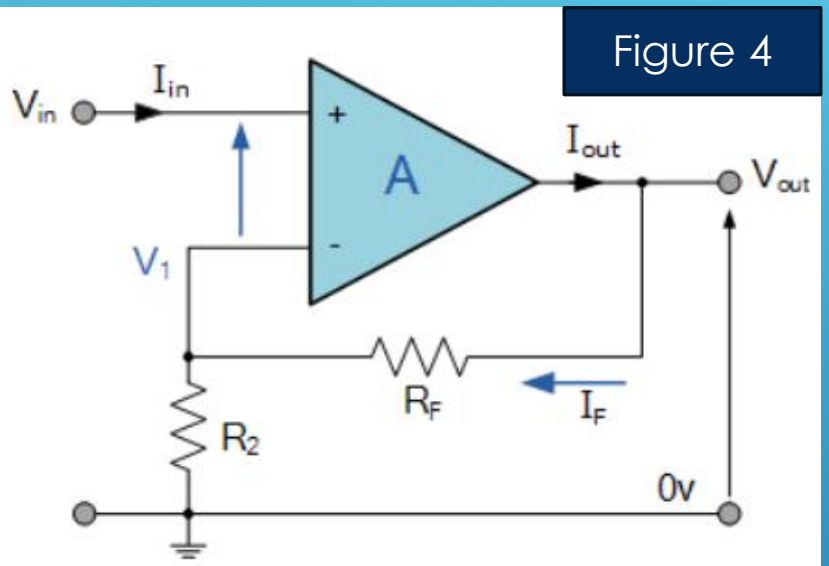
Figure 4



INTRODUCTION - WHAT IS THE UNITY GAIN BUFFER (OPERATIONAL AMPLIFIER)?

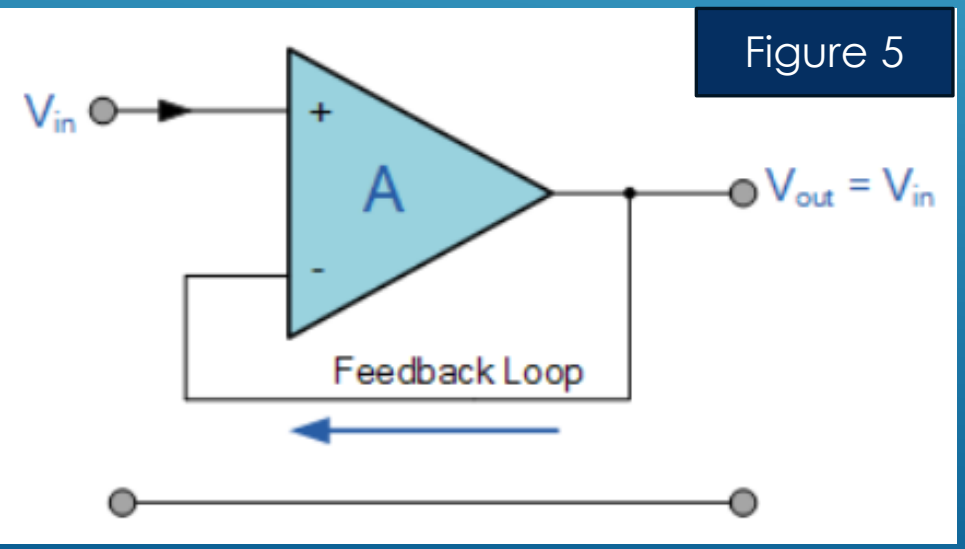
(PREPARATION TO P21)

Figure 4



If we made the feedback resistor, R_f equal to zero, and resistor R_2 equal to infinity, then the resulting circuit would have a fixed gain of "1" (unity) as all the output voltage is fed back to the inverting input terminal (negative feedback).
This configuration would produce a special type of the non-inverting amplifier circuit called a Voltage Follower, also known as a "unity gain buffer".

Figure 5



The advantage of the unity gain voltage follower configuration is that it can be used when impedance matching or circuit isolation is more important than voltage or current amplification as it maintains the input signal voltage at its output terminal.
Also, the input impedance of the voltage follower circuit is extremely high, typically above $1M\Omega$ as it is equal to that of the operational amplifiers input resistance times its gain (R_{in} x A_O).
The op-amps output impedance is very low since an ideal op-amp condition is assumed so is unaffected by changes in load.

INTRODUCTION - WHAT IS THE PRECISION HALF-WAVE RECTIFIER?

(THE "SUPERDIODE" BLOCK) – PART 1

(PREPARATION TO P21) [4]

The "Superdiode" precision half-wave rectifier, shown in Figure 6, consists of a diode placed in the negative-feedback path of an Op-Amp, with R being the rectifier load resistance.

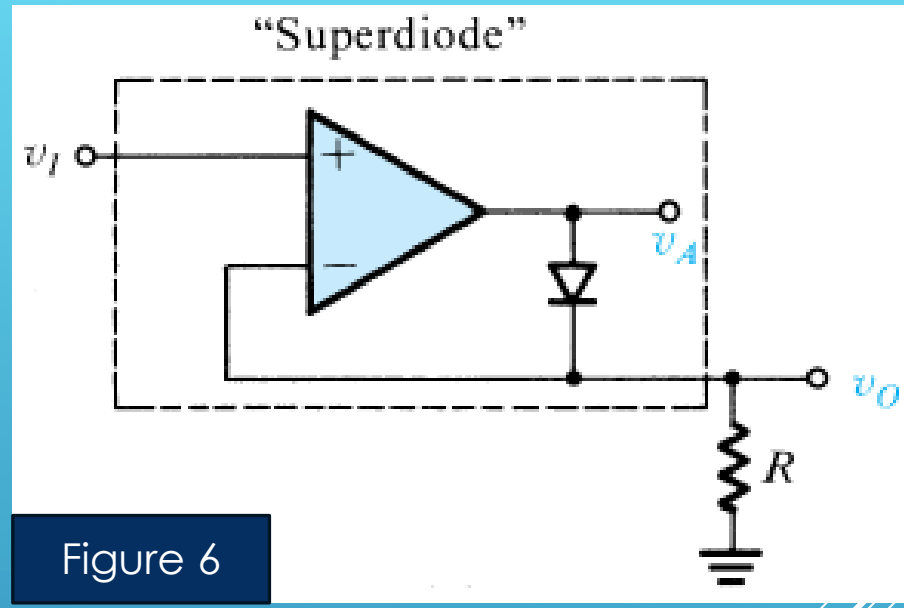
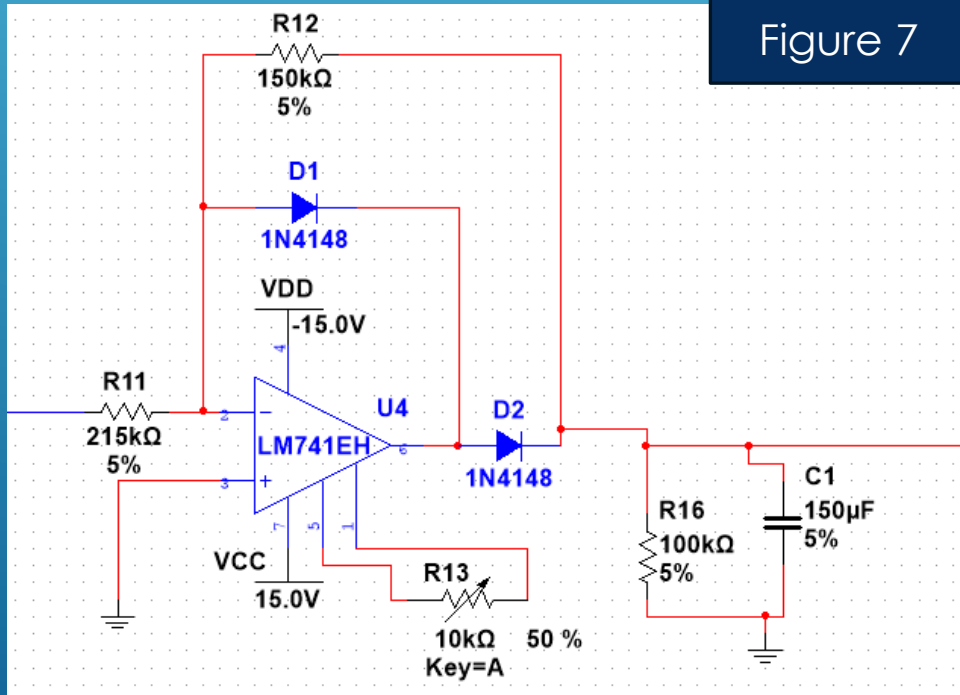


Figure 6

Disadvantages of this circuit:
When V_1 goes negative and $V_0 = 0$ the op amp may be damaged unless it is equipped with what is called "overvoltage protection". When V_1 is negative, the op amp will be saturated, which causes time delay that can slow down circuit operation.

We use the improved version, shown in Figure 7, of the precision half-wave rectifier, which includes diode D_1 in order to keep the feedback loop closed around the op amp during the off times of the rectifier diode D_2, thus preventing the op amp from saturating.

Figure 7



INTRODUCTION - WHAT IS THE PRECISION HALF-WAVE RECTIFIER?

(THE "SUPERDIODE" BLOCK) – PART 2

(PREPARATION TO P21) [4]

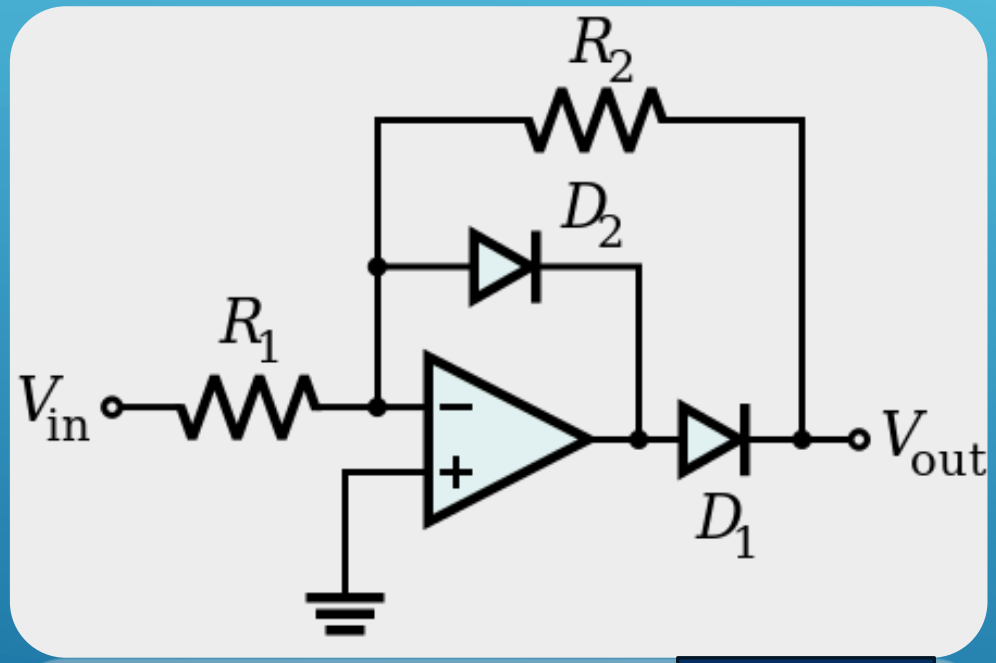


Figure 8

We use the improved version, shown in Figure 7, of the precision half-wave rectifier

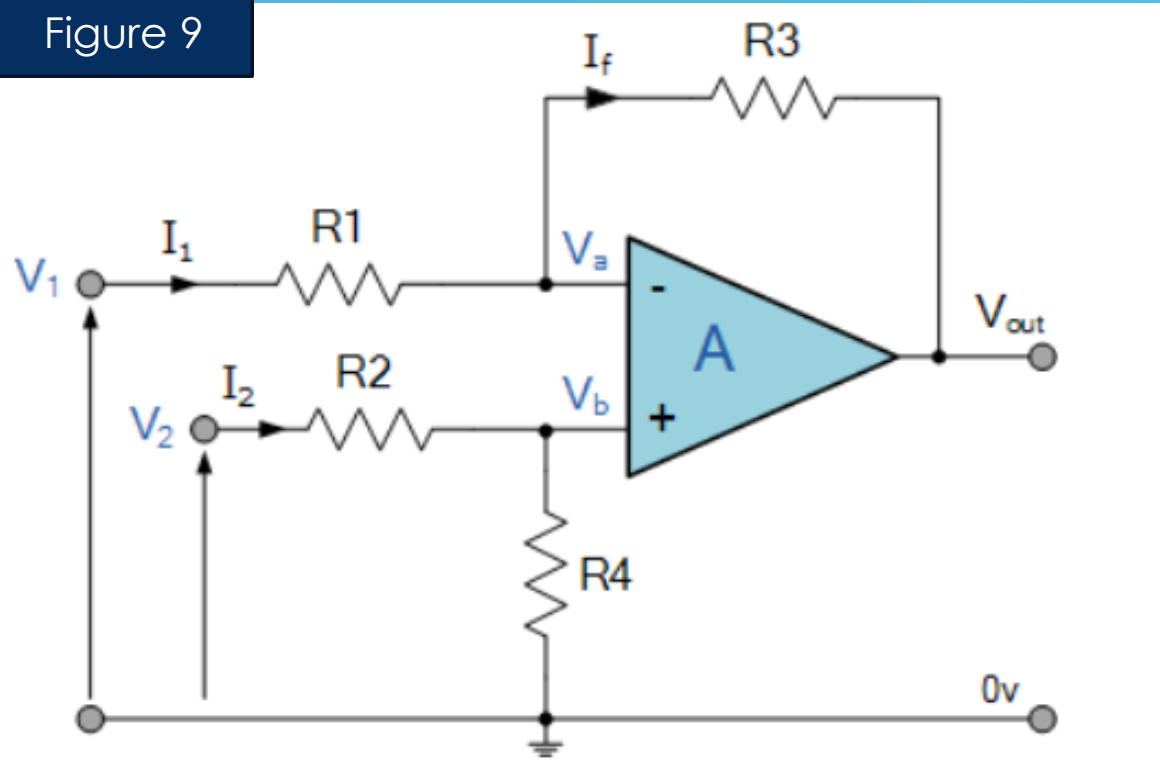
The circuit in figure 8 has the benefit that the op-amp never goes into saturation, but its output must change by two diode voltage drops (about 1.2 V) each time the input signal crosses zero. Hence, the slew rate of the operational amplifier and its frequency response (gain–bandwidth product) will limit high-frequency performance, especially for low signal levels, although an error of less than 1% at 100 kHz is possible.

https://en.wikipedia.org/wiki/Precision_rectifier

INTRODUCTION - WHAT IS THE DIFFERENTIAL AMPLIFIER ?

(PREPARATION TO P21)

Figure 9



The differential amplifier is a voltage subtractor circuit which produces an output voltage proportional to the voltage difference of two input signals applied to the inputs of the inverting and non-inverting terminals of an operational amplifier.

differential amplifiers amplify the difference between two voltages making this type of operational amplifier circuit a Subtractor unlike a summing amplifier which adds or sums together the input voltages. This type of operational amplifier circuit is commonly known as a Differential Amplifier configuration and is shown below:

When resistors, R1 = R2 and R3 = R4 the above transfer function for the differential amplifier can be simplified to the following expression:

$$V_{OUT} = \frac{R_3}{R_1}\left(V_2 - V_1\right)$$

https://www.electronics-tutorials.ws/opamp/opamp_lmth.5

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 1)

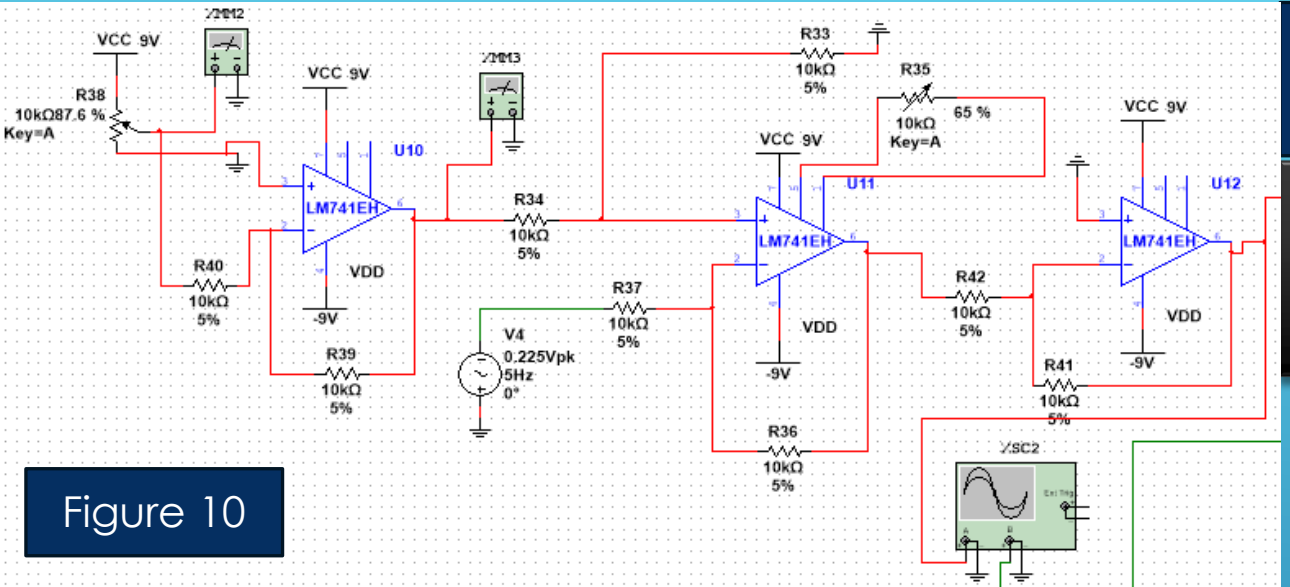
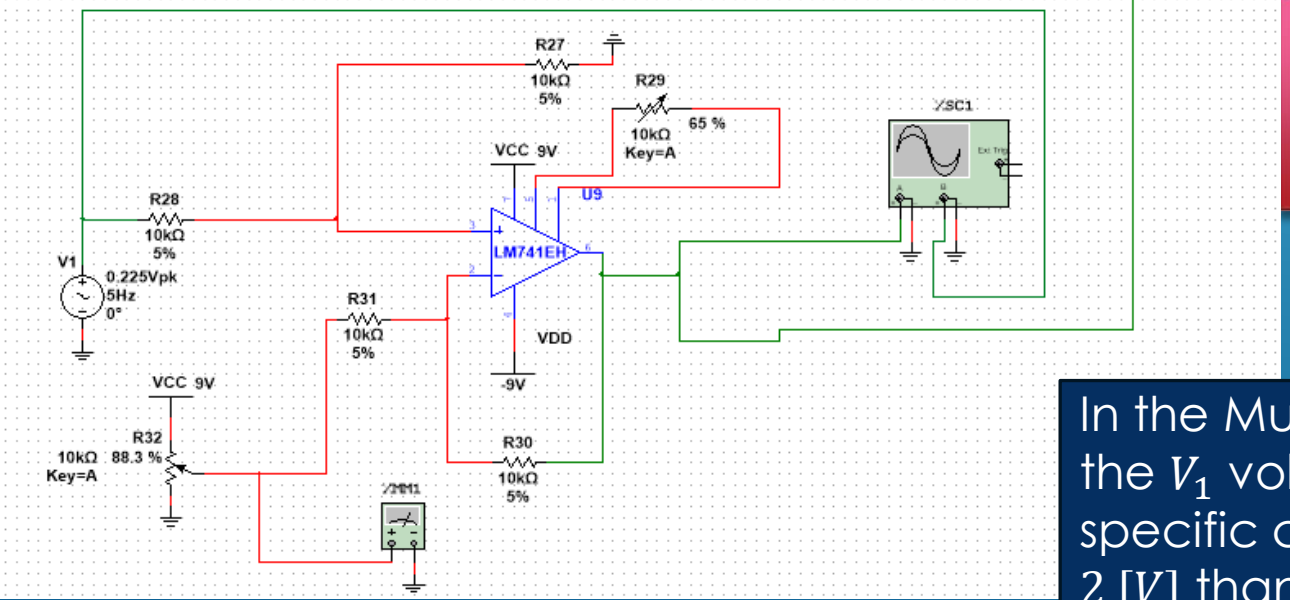


Figure 10



This block is relevant for taking care of the DC offset voltage we add to the input signals (to simulate noise)

This block will be the input of the instrumentation amplifier stage (**second stage**) if we want to input the system a differential AC voltage with DC offset

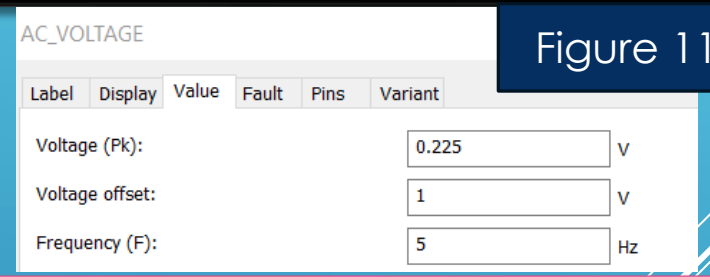


Figure 11

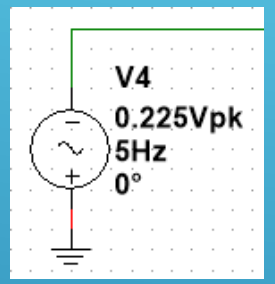
Because using that block demand calibration using the potentiometer we will use it **only** when trying to simulate differential AC voltage with DC offset

$$V_{OUT} = \frac{R_3}{R_1}\left(V_2 - V_1\right)$$

Figure 12

In the Multisim simulation we used a multimeter to show that the V_1 voltage (in the formula) is a constant 1 $[V]$ – in the specific case that the DC offset is 1 $[V]$ (if the DC offset will be 2 $[V]$ than need to manually adjust the potentiometer to produce (it is simply a voltage divider) 2 $[V]$ as V_1 voltage

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 2)



Negative polarized AC voltage level shifter subblock

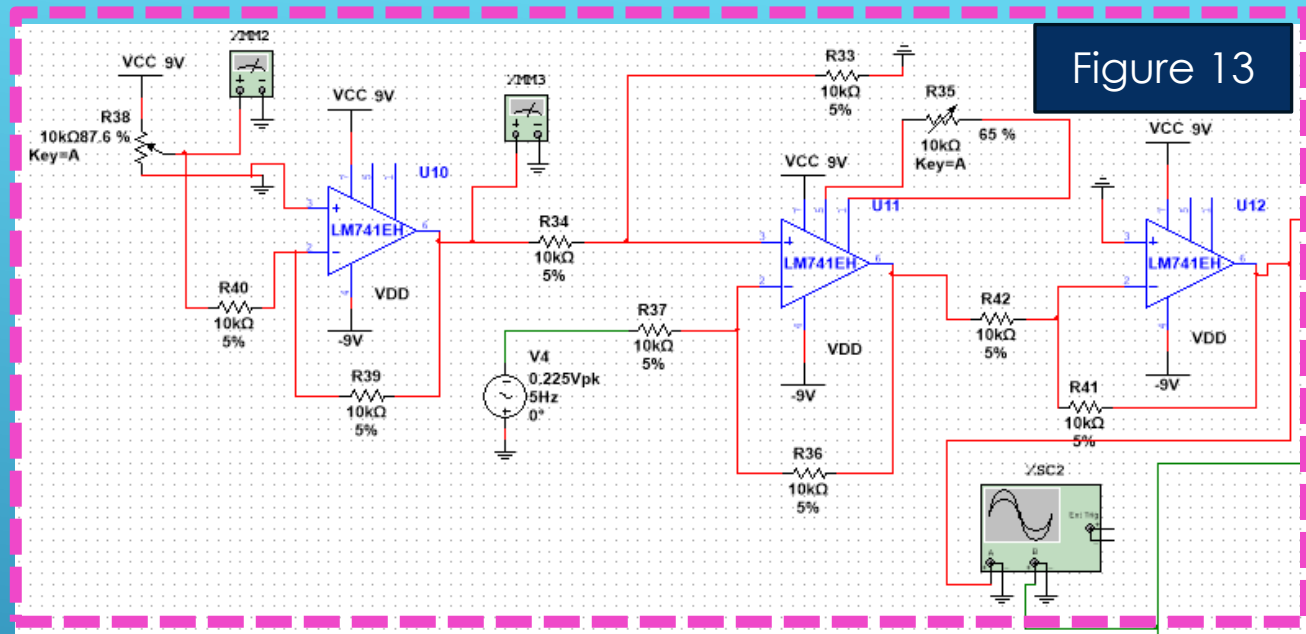
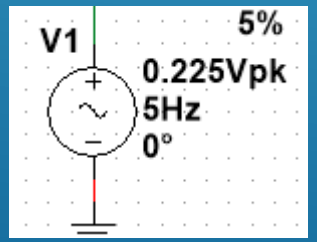


Figure 13



positive polarized AC voltage level shifter subblock

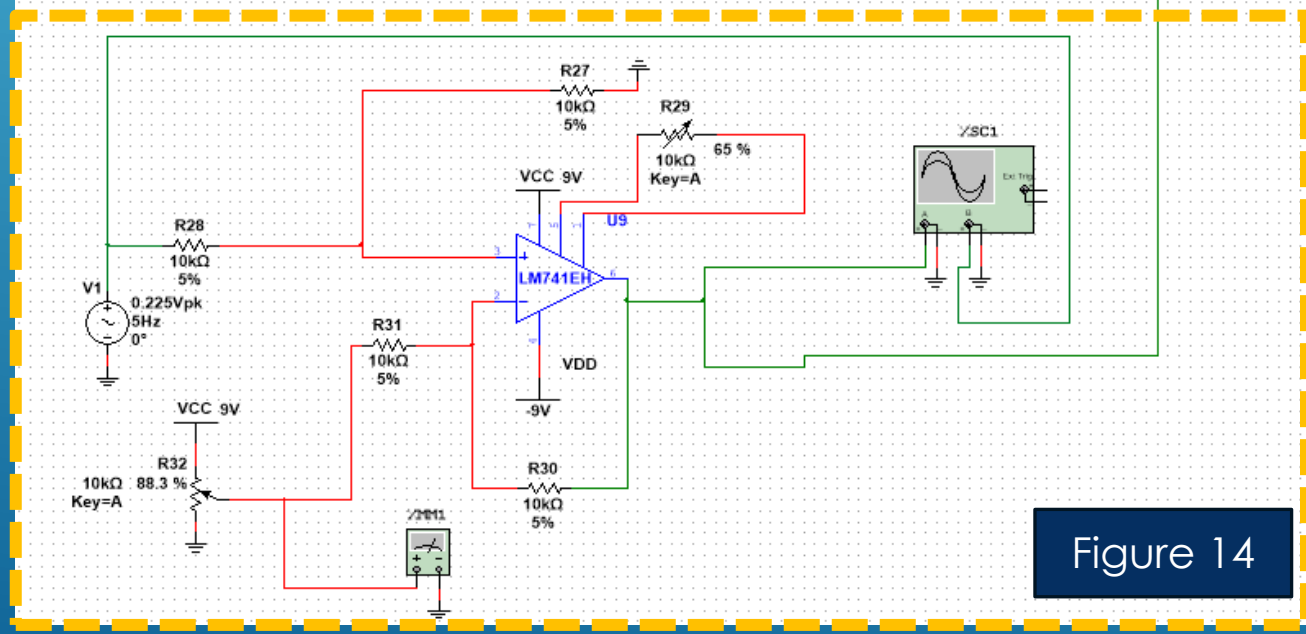


Figure 14

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 3)

The negative polarization input is Comprised of 2 invertive OP amps and a differential amplifier

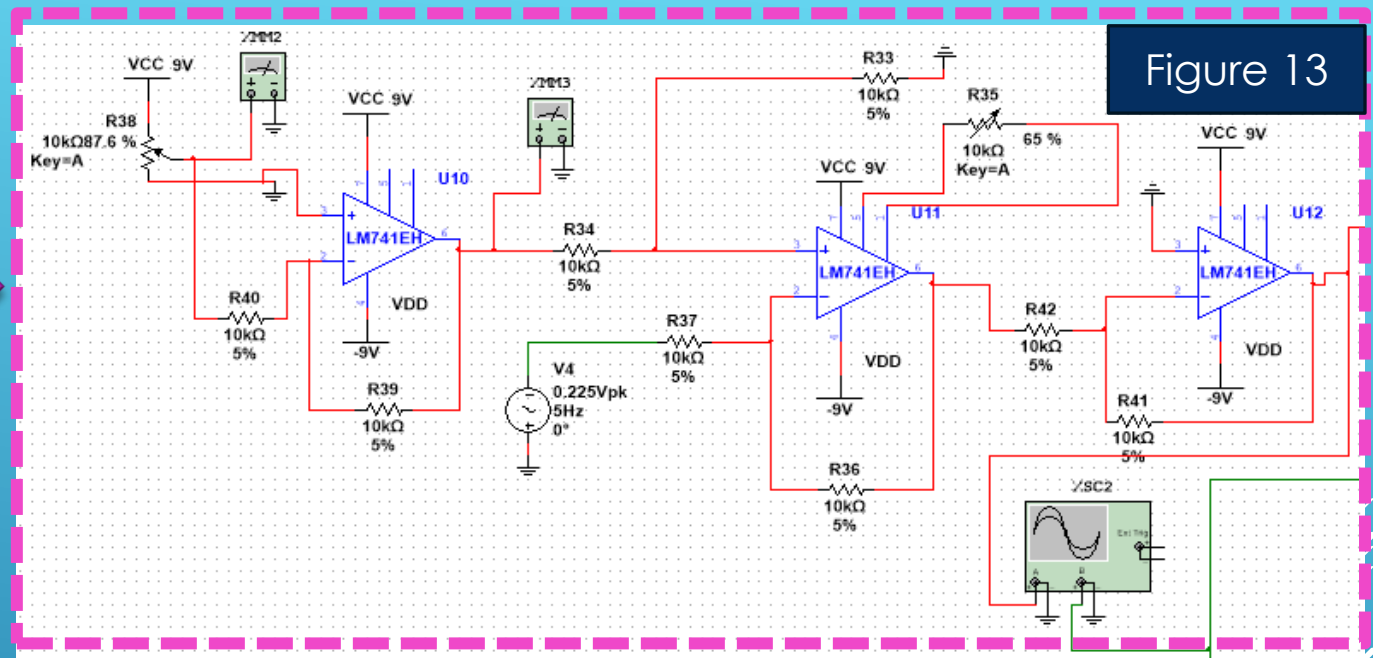


Figure 13

The positive polarization input is Comprised of only a differential amplifier

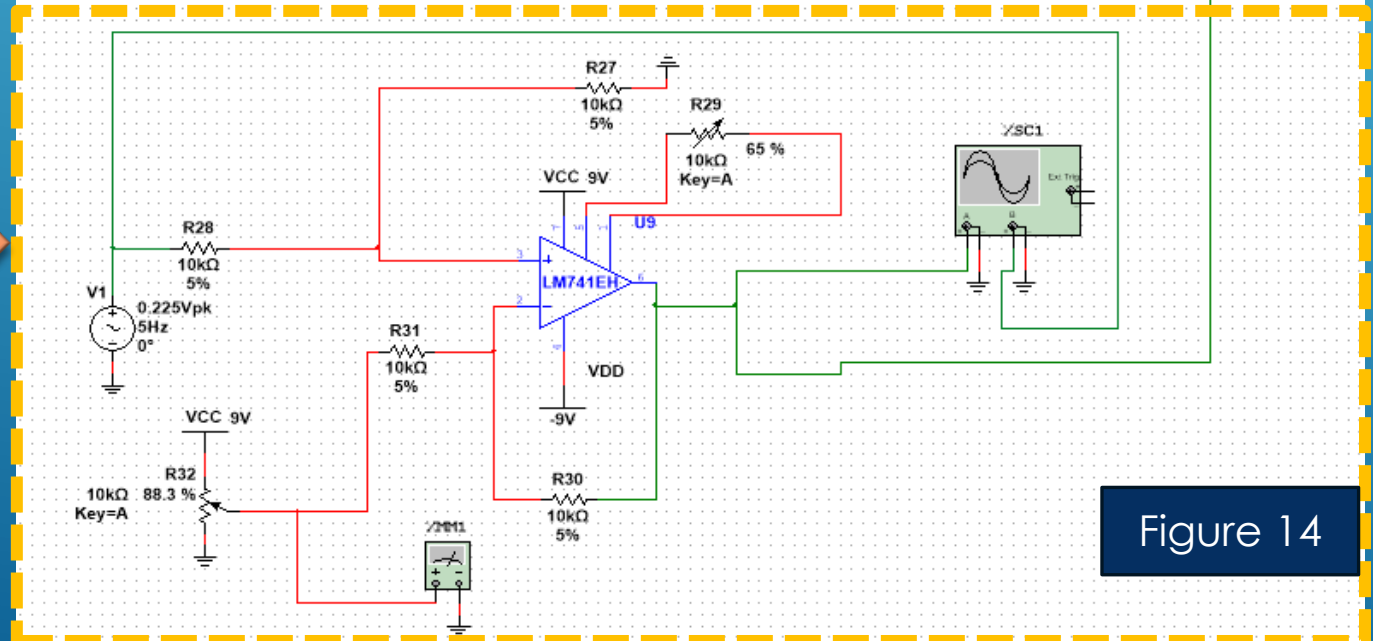


Figure 14

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 4)

Potentiometer as voltage divider

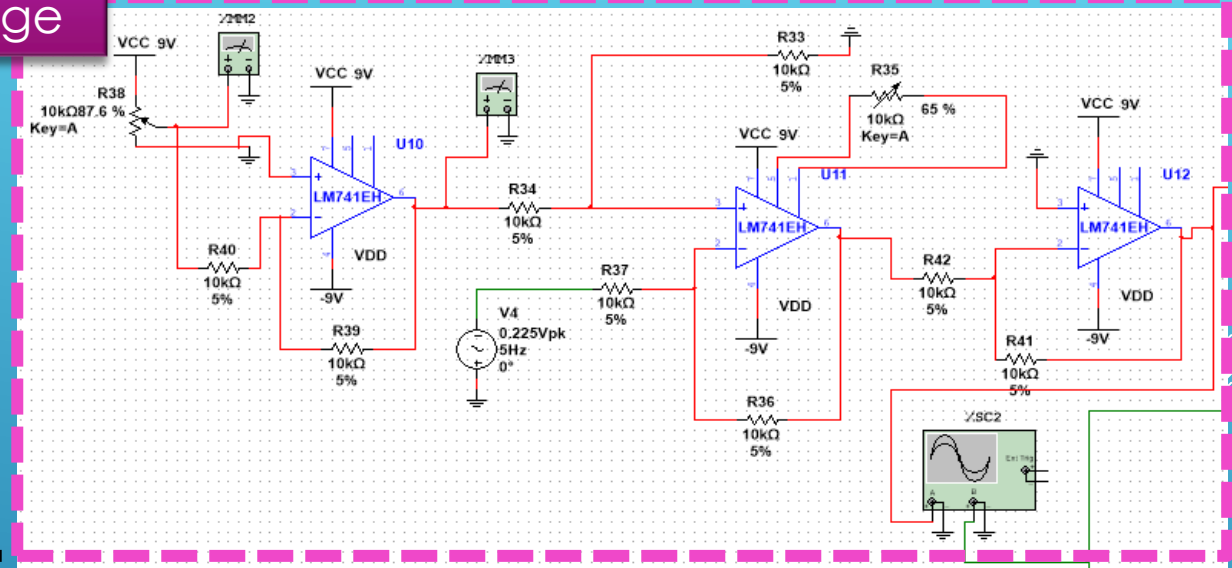
Inverter (741)

Two inputs of differential amplifier – first is the output of the inverter and the second is the AC voltage with DC offset

Inverter (741)

Output to one of the inputs of the instrumentation amplifier stage

Figure 13

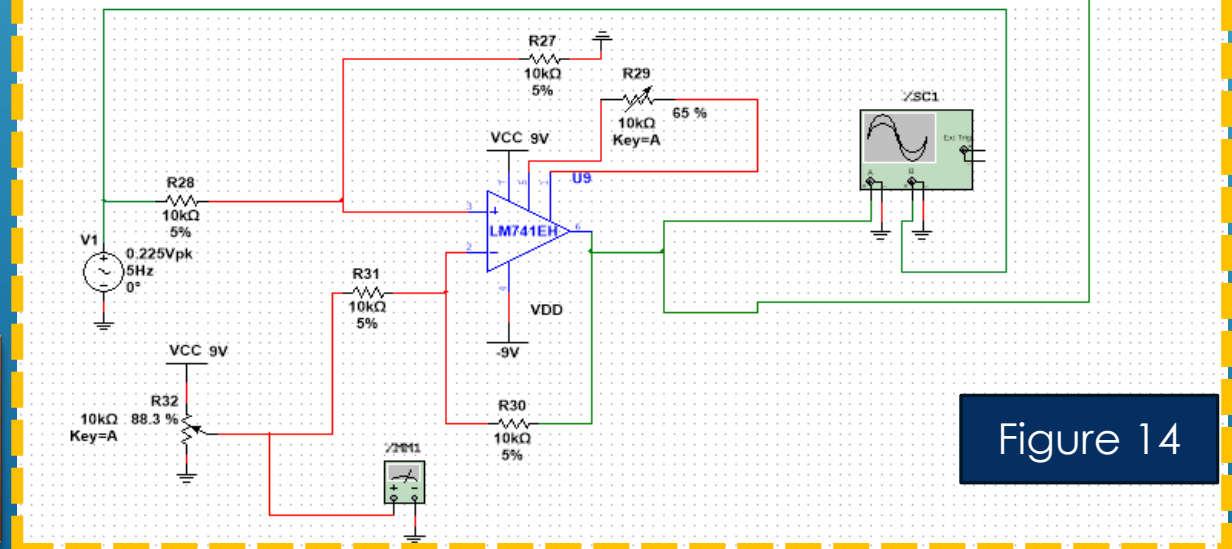


Potentiometer as voltage divider

Two inputs of differential amplifier – first is the output of the inverter and the second is AC voltage with DC offset

Output to one input of the instrumentation amplifier stage

Figure 14



P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 5)

The negative polarization input is Comprised of 2 invertive OP amps and a differential amplifier

The potentiometer output is adjusted to be 1v (in the case of DC offset of 1V but can be any other level we want)

Because we deal with the negative polarization input (AC voltage source) we need to invert the output of the potentiometer – we do so using an inverting OP amp

Than input to the differential amplifier (with equal resistors for gain of 1) for: $V^+ - V^-$

A closer look at the **negative** polarization AC voltage source (with DC offset) – **part 1**

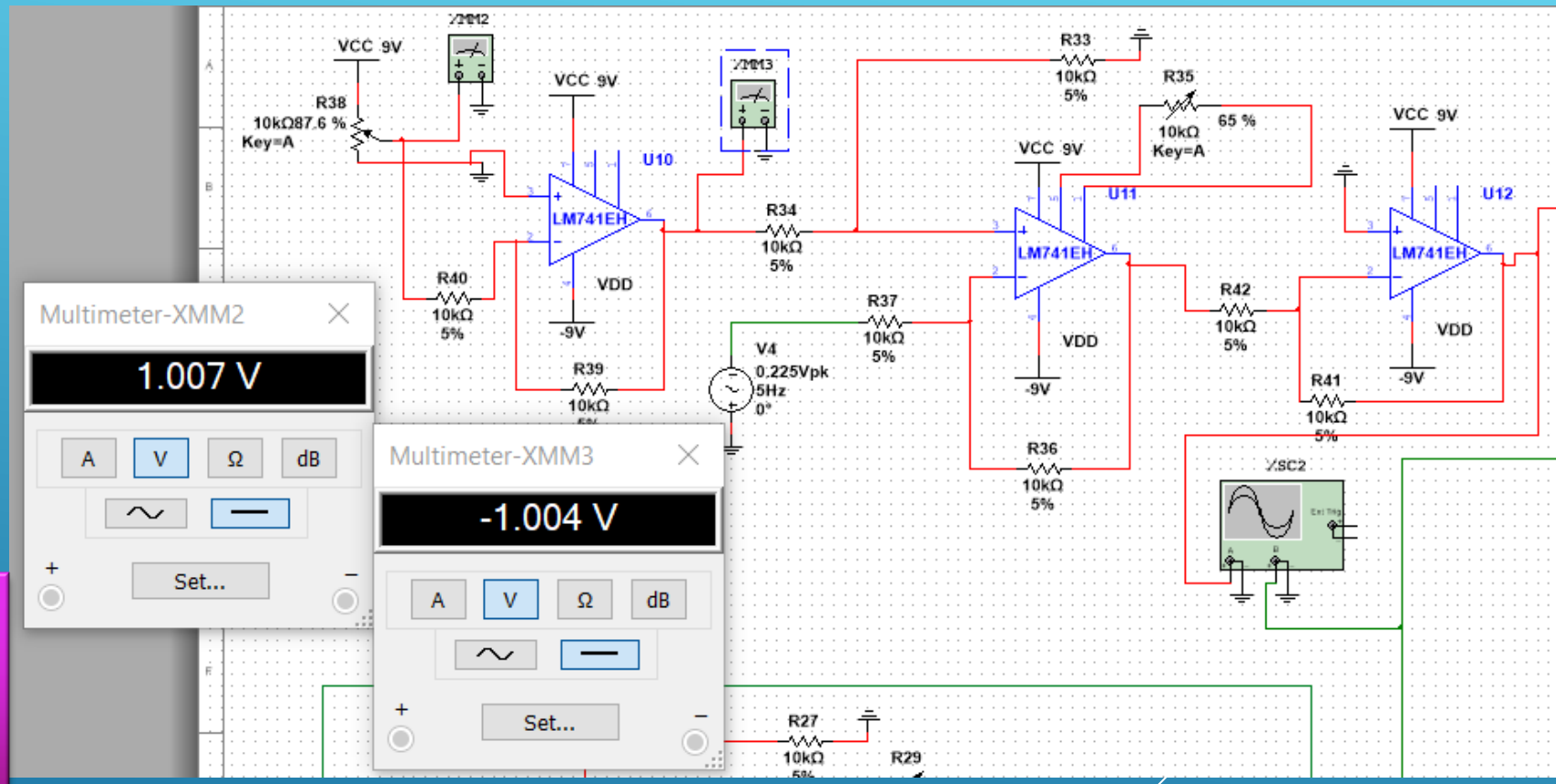
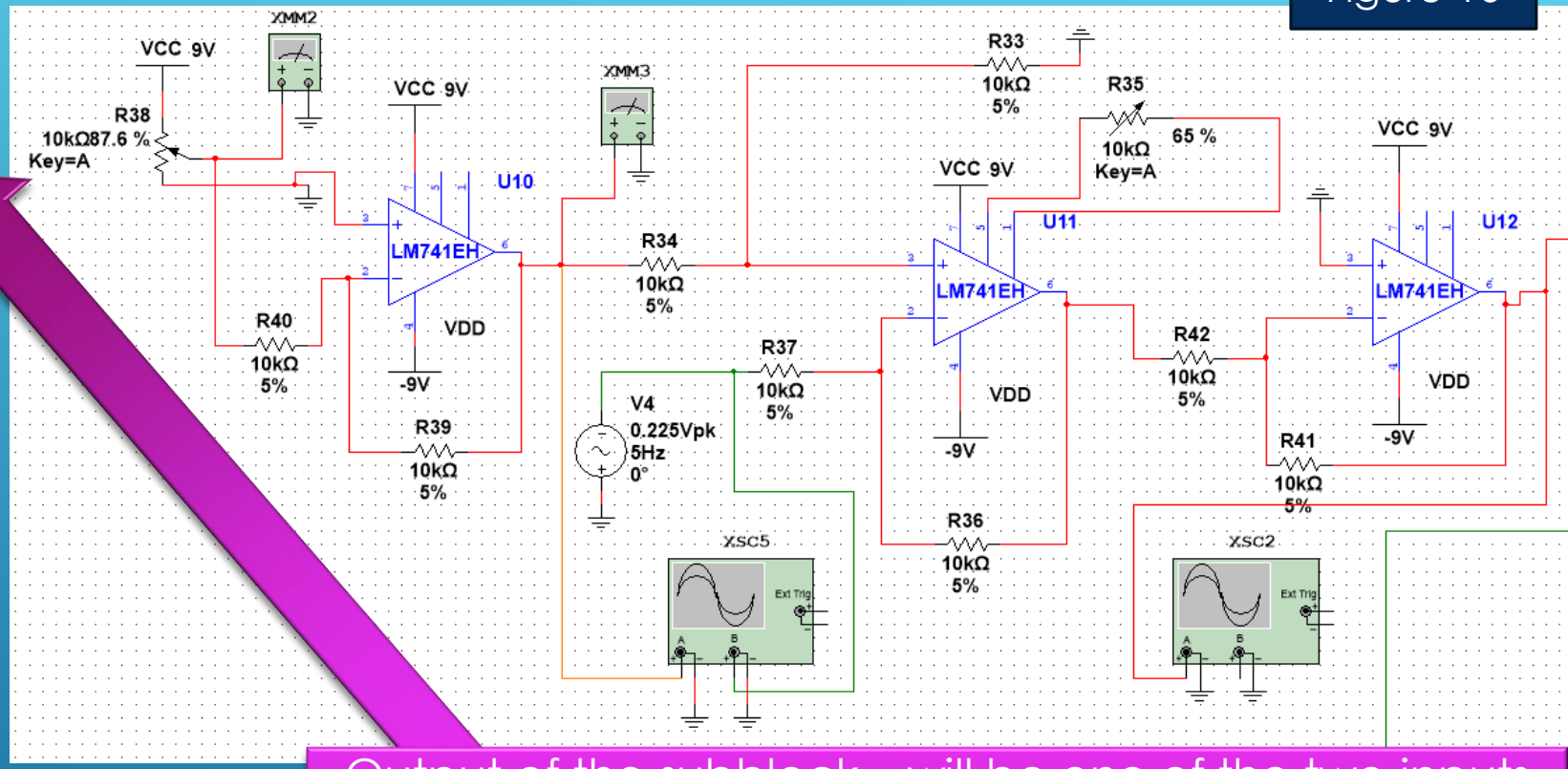


Figure 15

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 6)

A closer look at the **negative** polarization AC voltage source (with DC offset) – **part 2**

Figure 16



Output of the subblock – will be one of the two inputs of the instrumentational amplifier stage

The two inputs signals of the differential amplifier in that subblock – the **green** is the AC voltage source (with DC offset of 1 [v]) and the **orange** is the output of the potentiometer to balance the DC offset

Figure 17

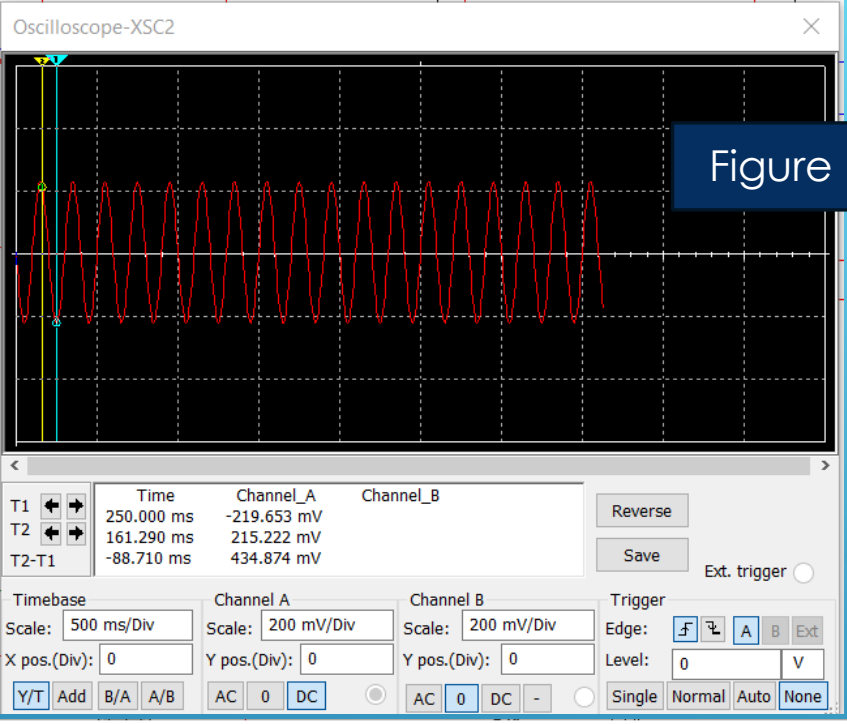
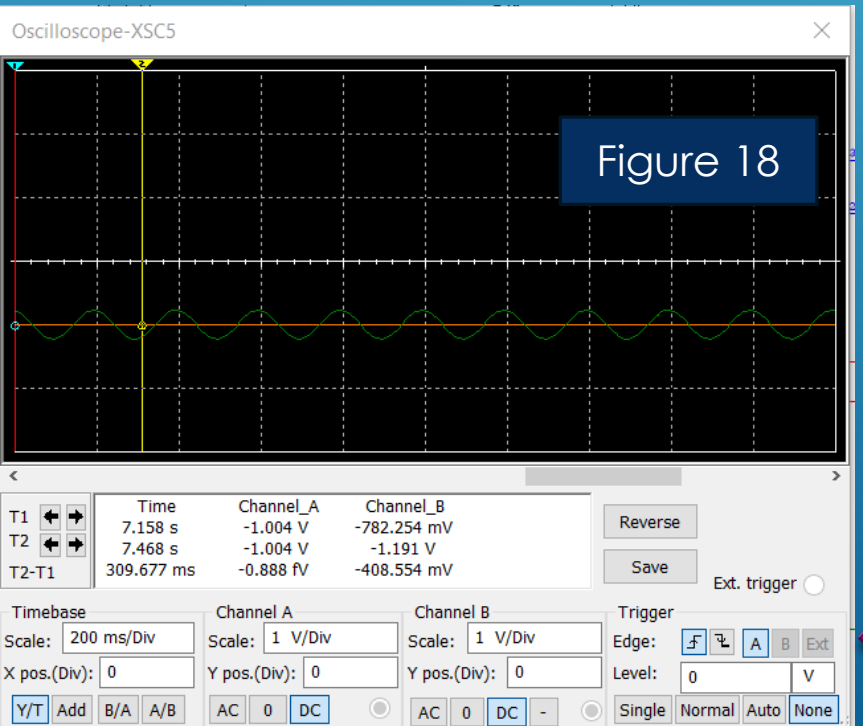
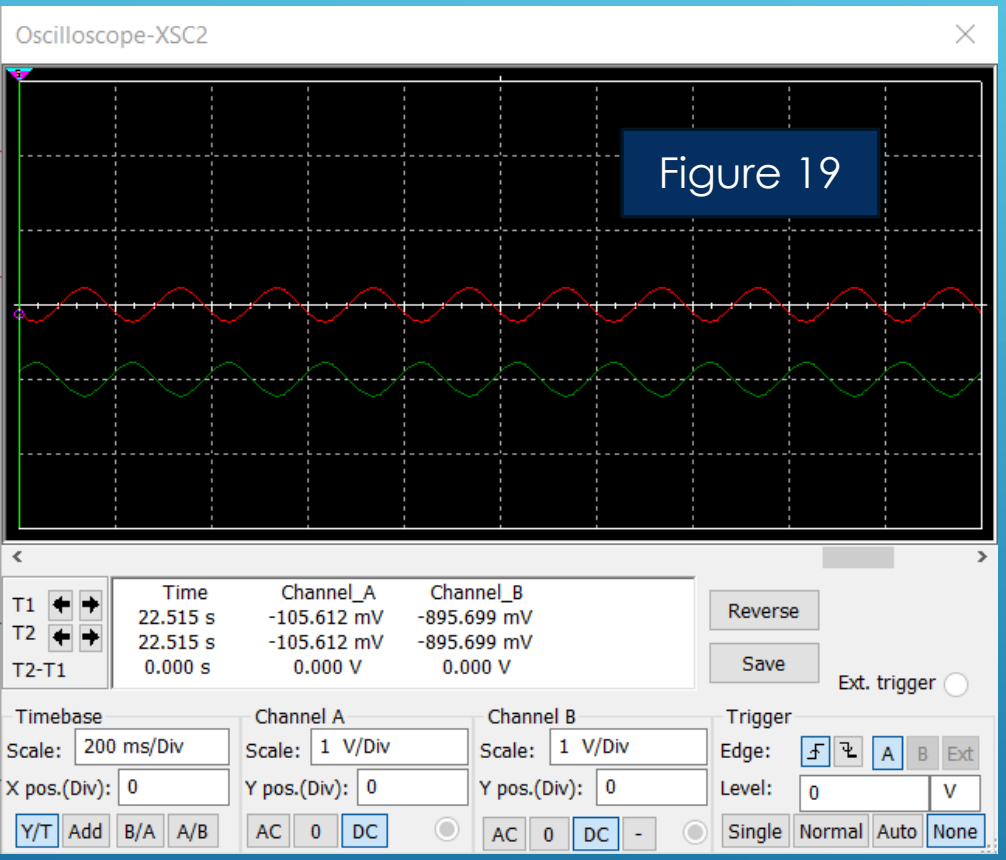


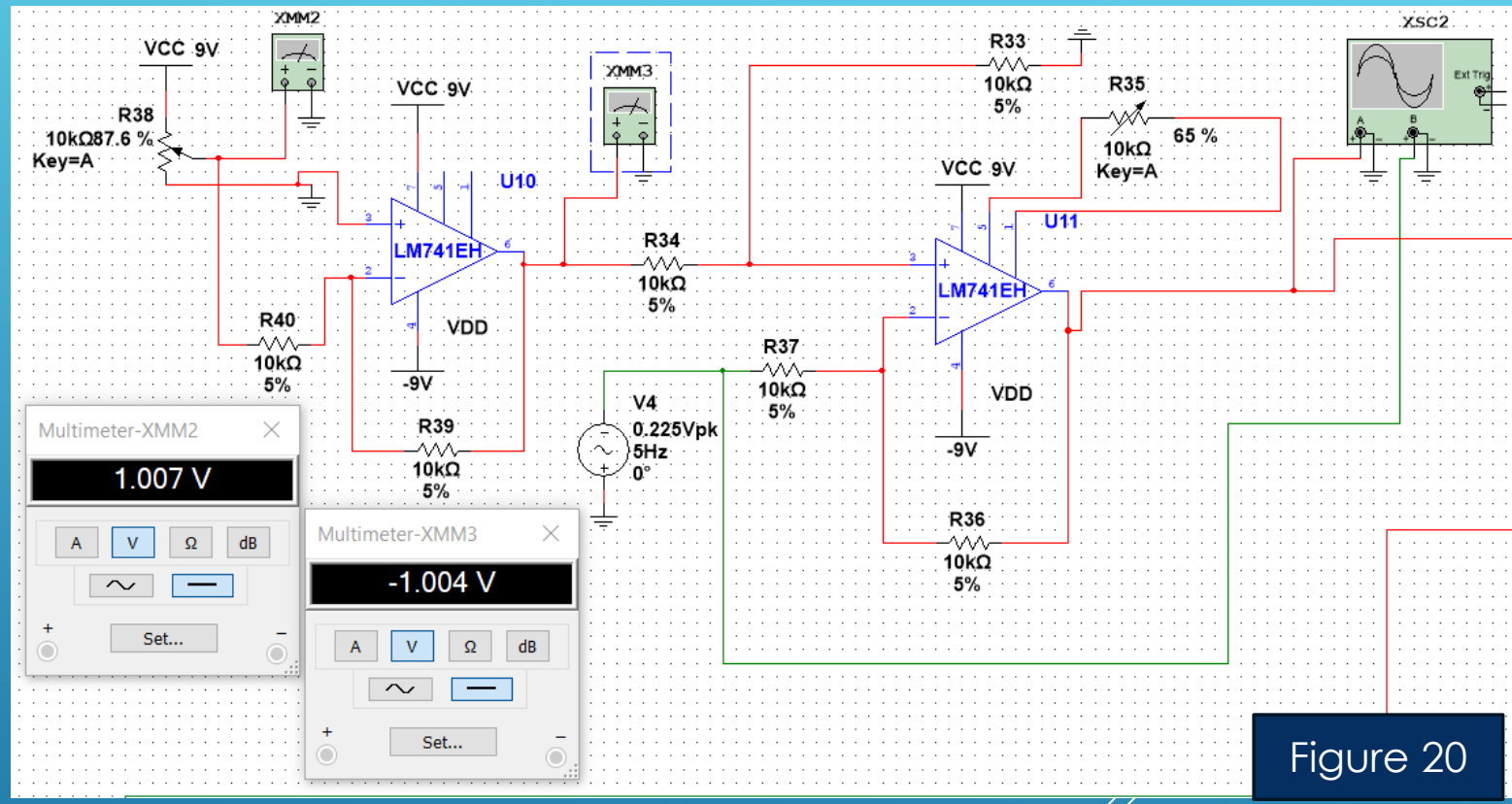
Figure 18



P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 7)

A closer look at the **negative** polarization AC voltage source (with DC offset) – **part 3**





The green signal is the input signal (negative polarization) (with DC offset) of 1V and the output signal of the level shifter is the red signal (without the DC offset-balanced)

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 8)

A closer look at the **negative** polarization AC voltage source (with DC offset) –in TinkerCad-**part 4**

Figure 22

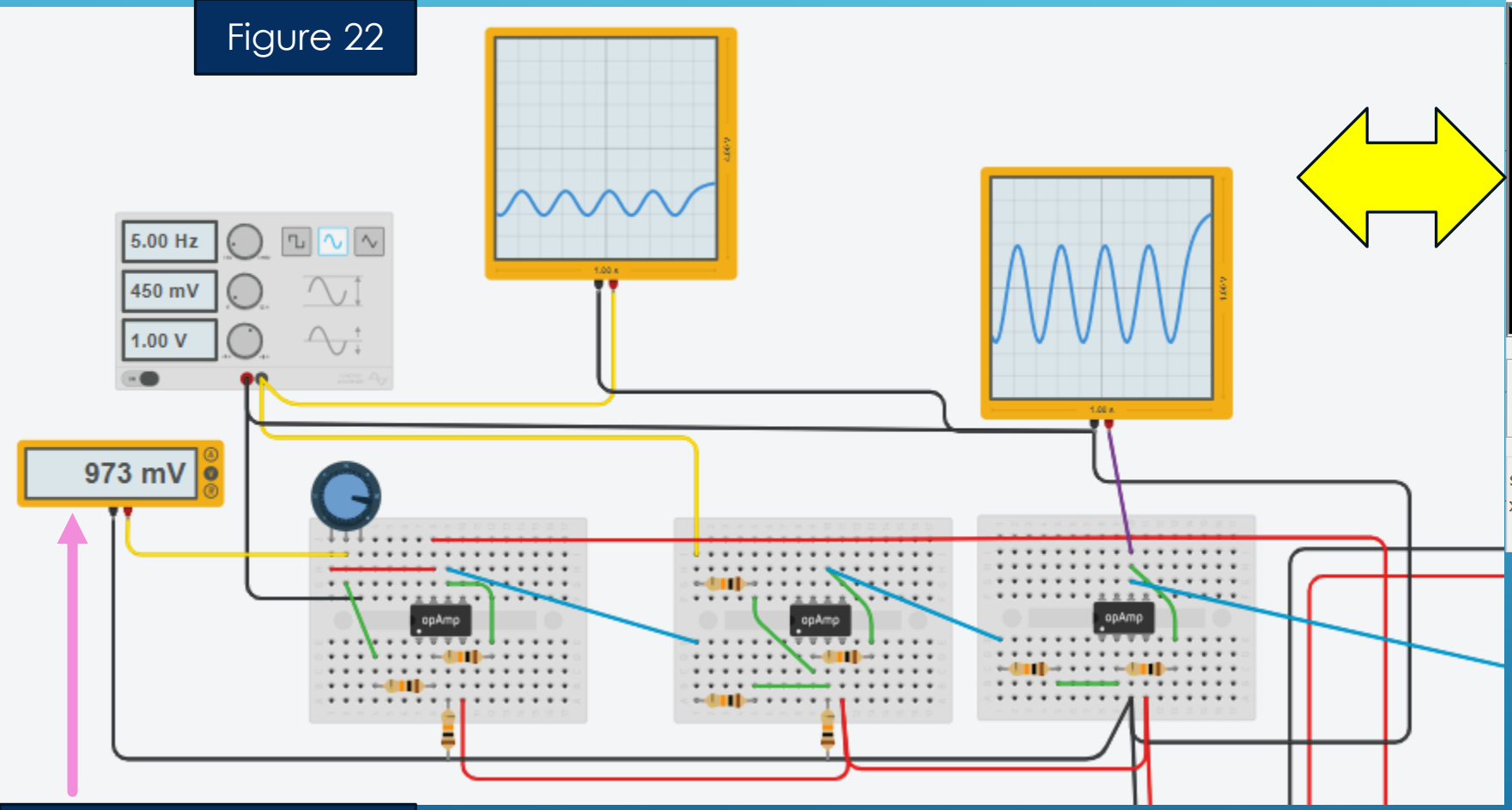
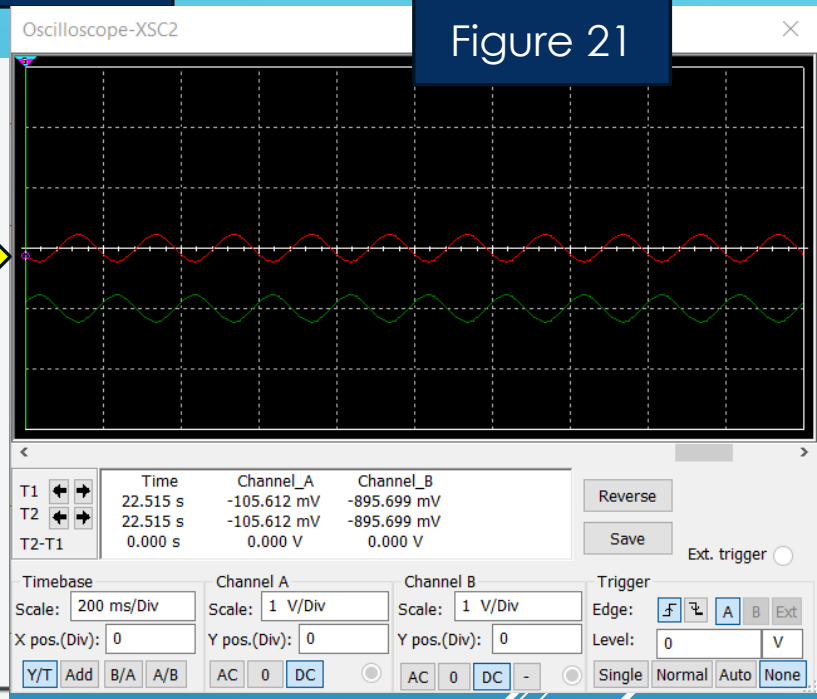


Figure 21



Failed to reach 1[V] but good enough

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 8)

A closer look at the **positive** polarization AC voltage source (with DC offset) – **part 1**

Figure 23

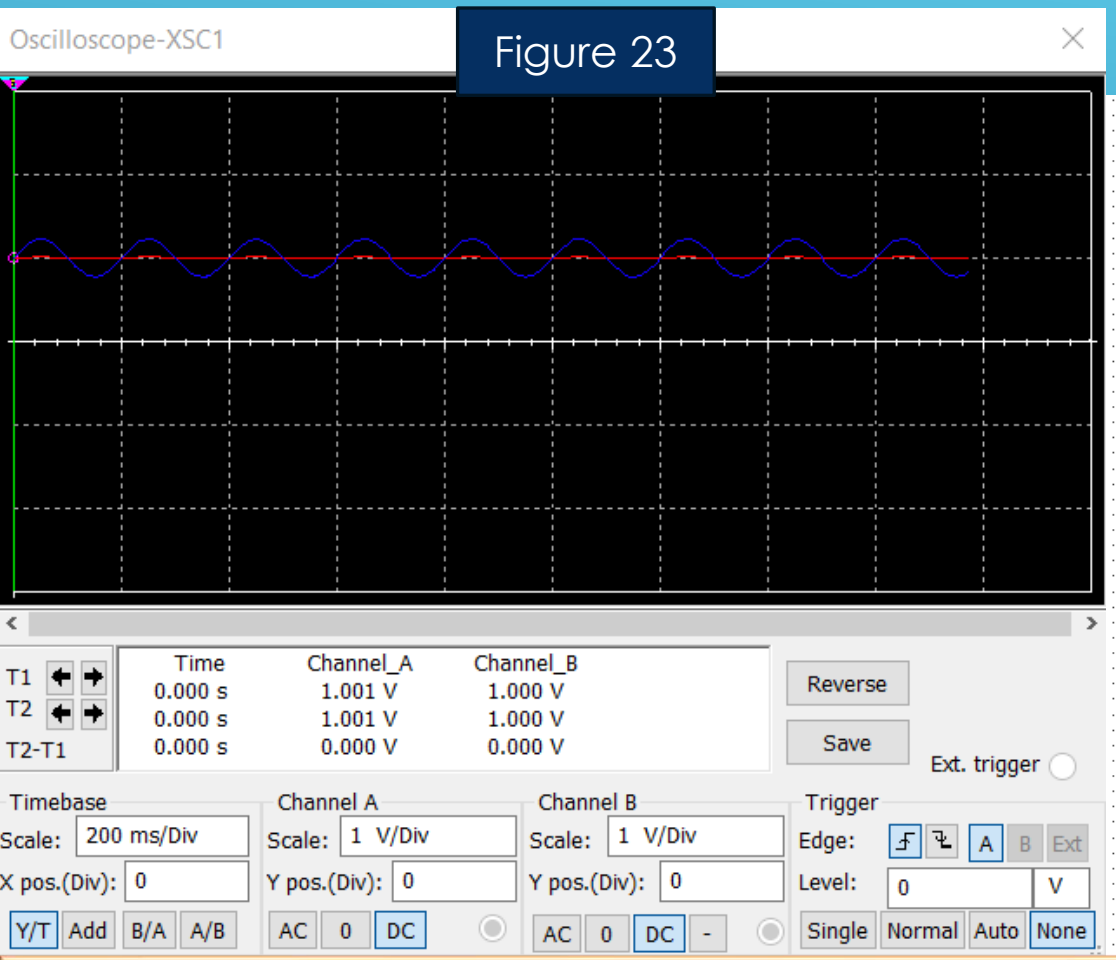
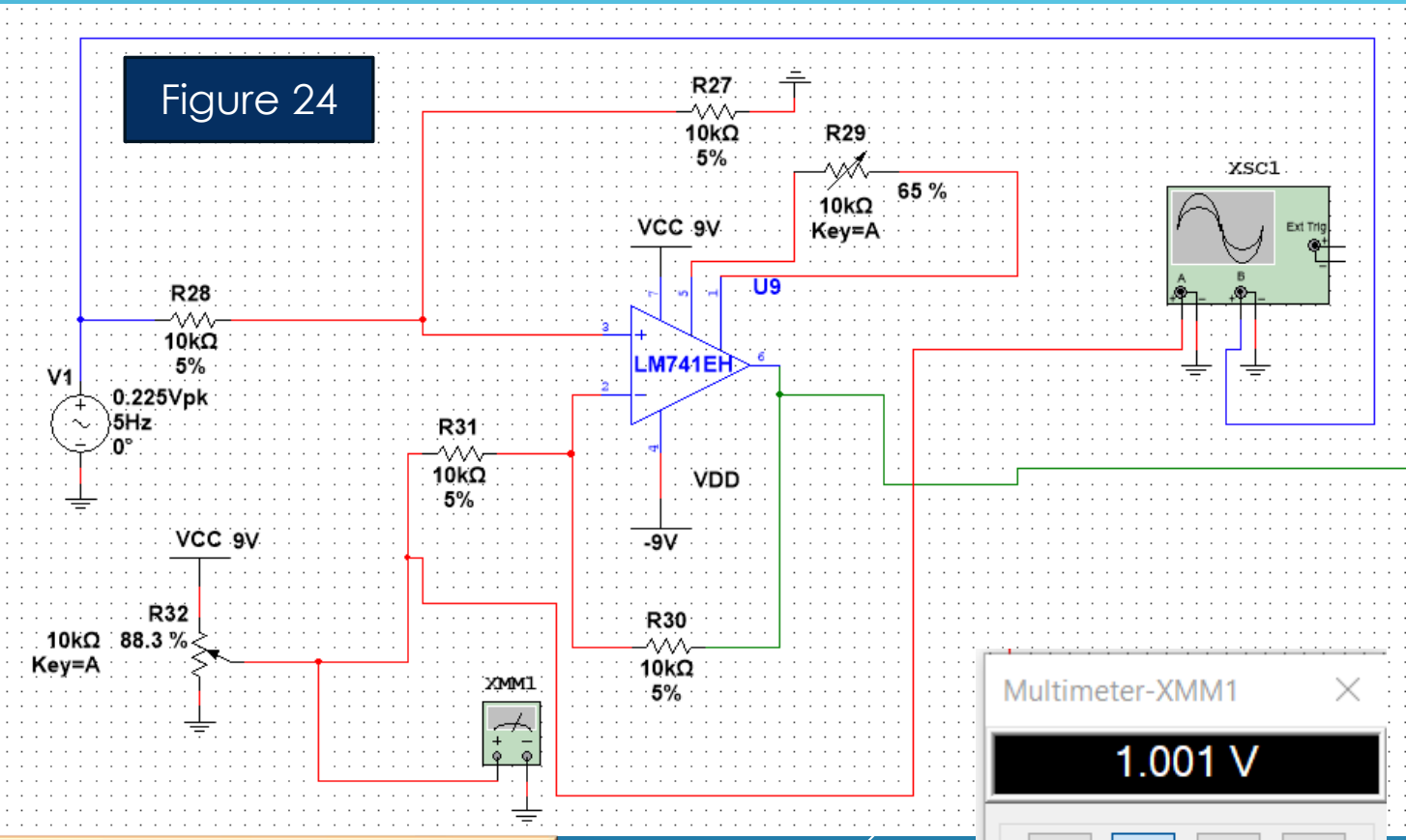


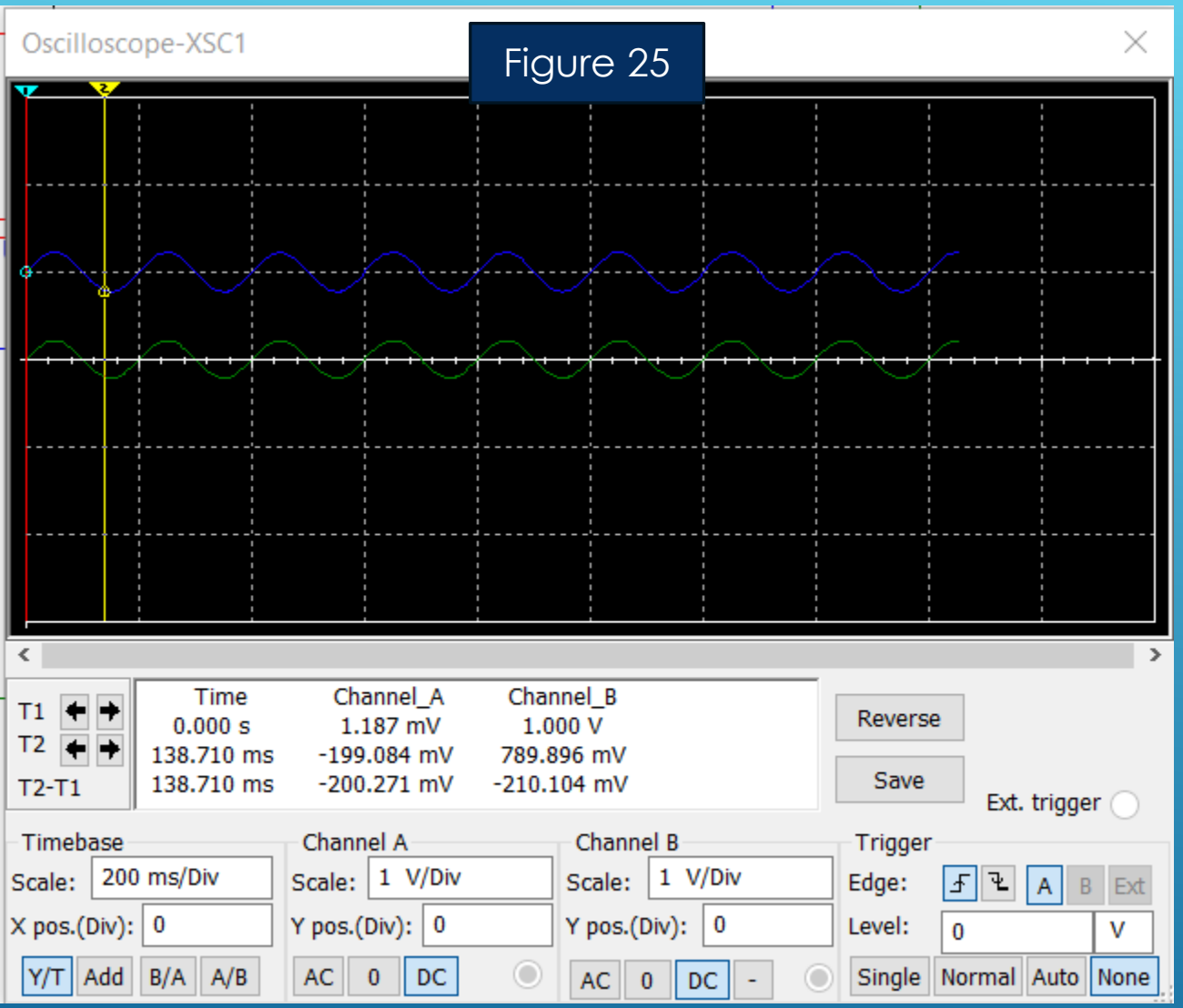
Figure 24



The two inputs signals of the differential amplifier in that subblock – the **blue** is the AC voltage source (with DC offset of 1 [v]) and the **red** is the output of the potentiometer to balance the DC offset

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 9)

Figure 25



A closer look at the **positive** polarization AC voltage source (with DC offset) – **part 2**

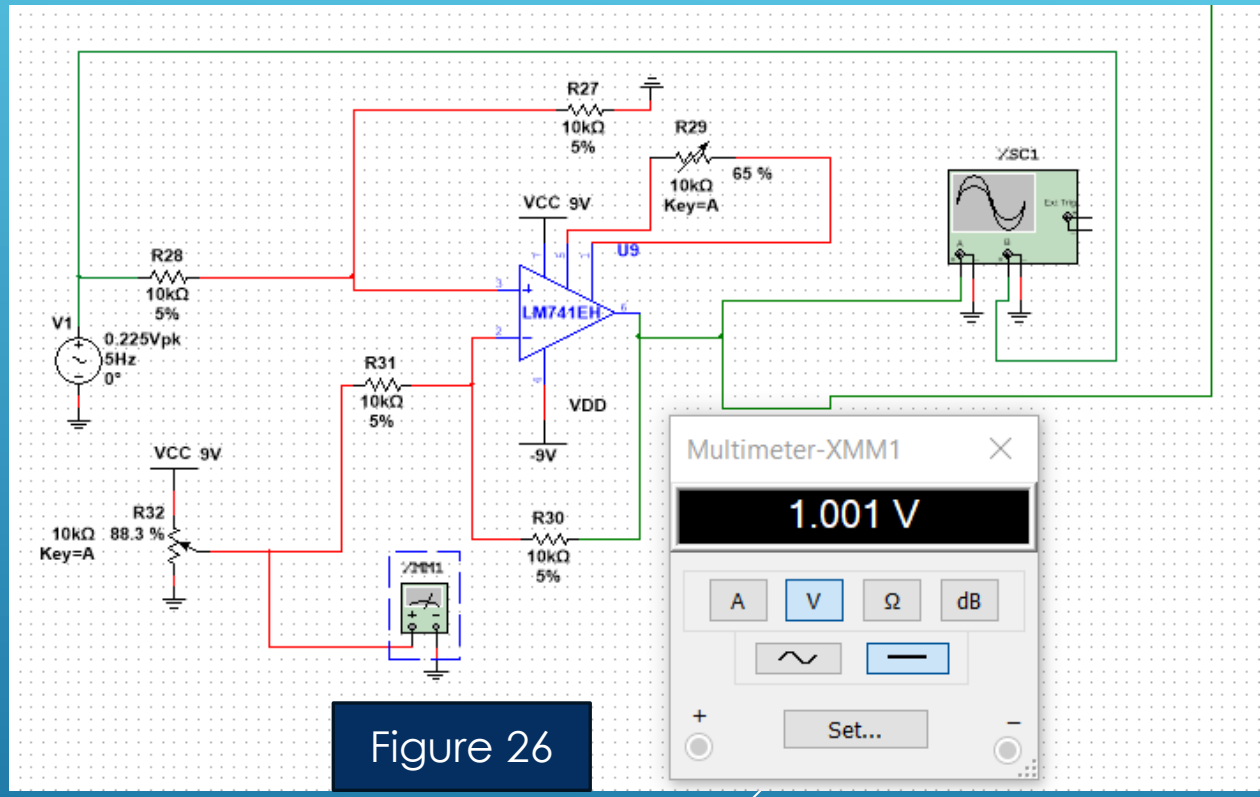


Figure 26

Output of the subblock – will be one of the two inputs of the instrumentational amplifier stage (in **green**) and the AC voltage source (with DC offset of 1[V]) in **blue**.

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) (PART 10)

A closer look at the **positive** polarization AC voltage source (with DC offset) – **part 3 (in TinkerCad)**

The scopes show the input (dc offset of 1) and output as a balanced signal without offset

Figure 27

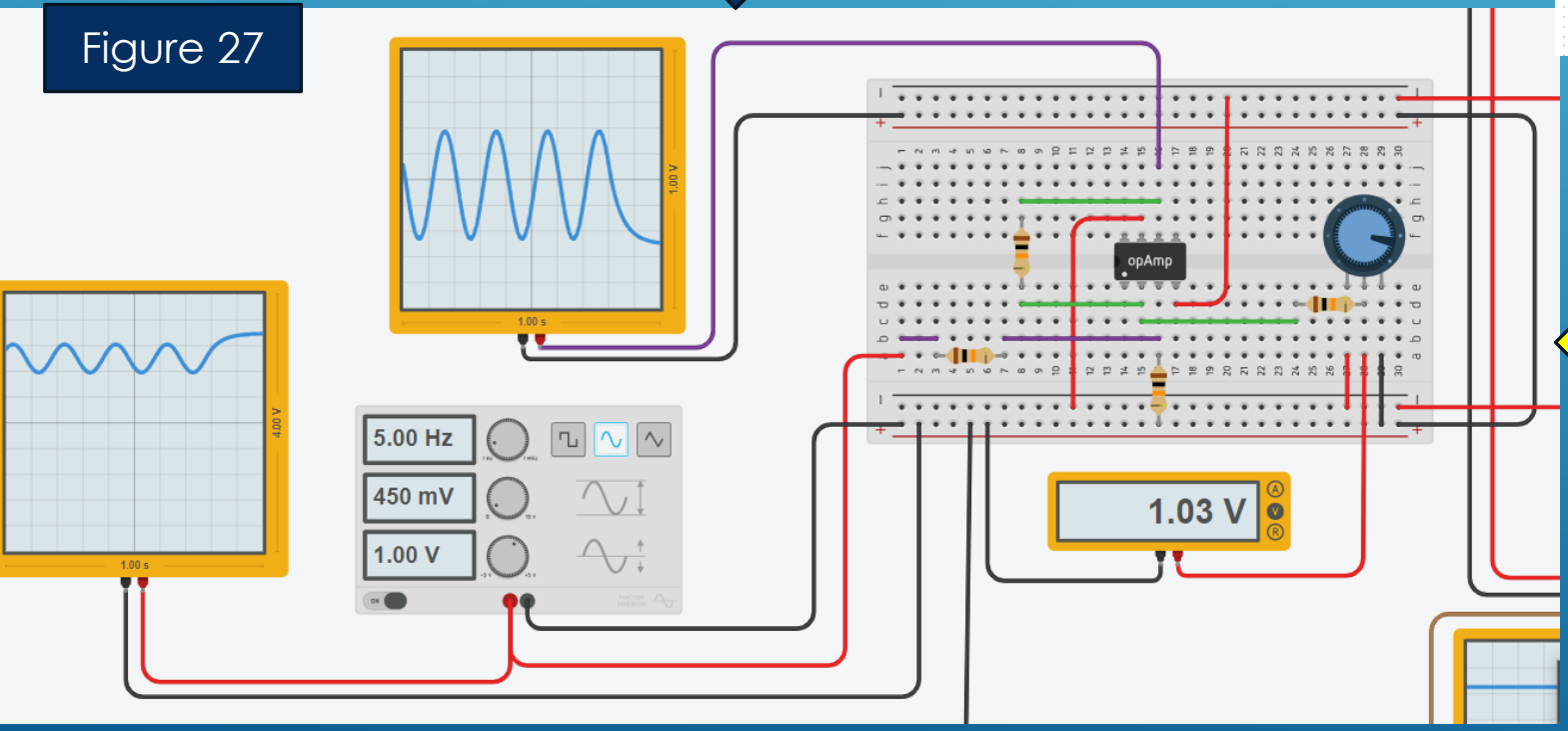


Figure 28

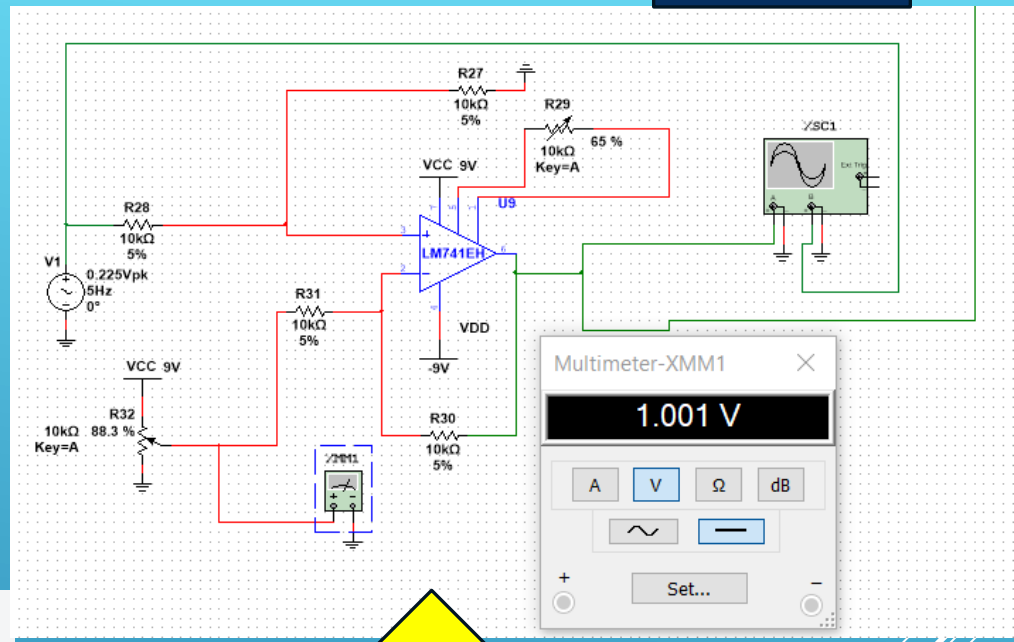
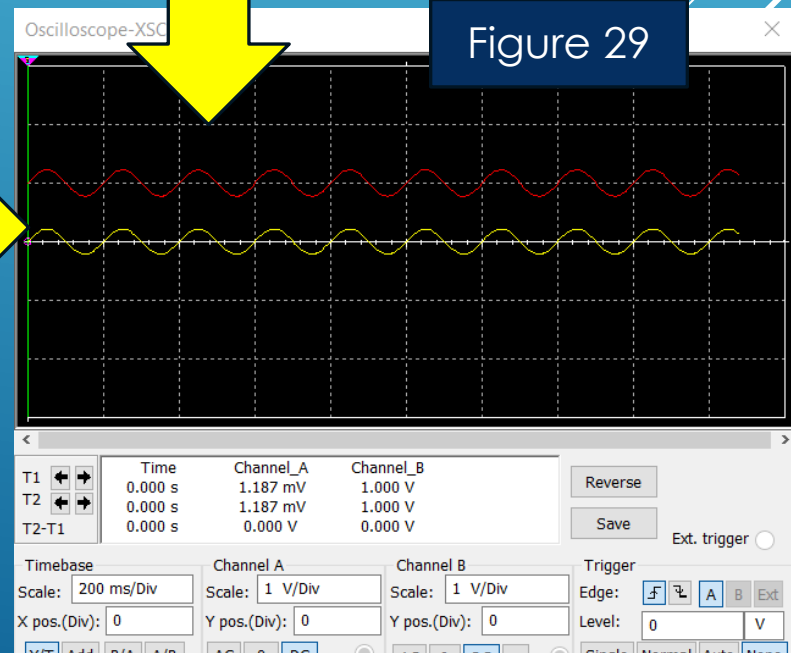
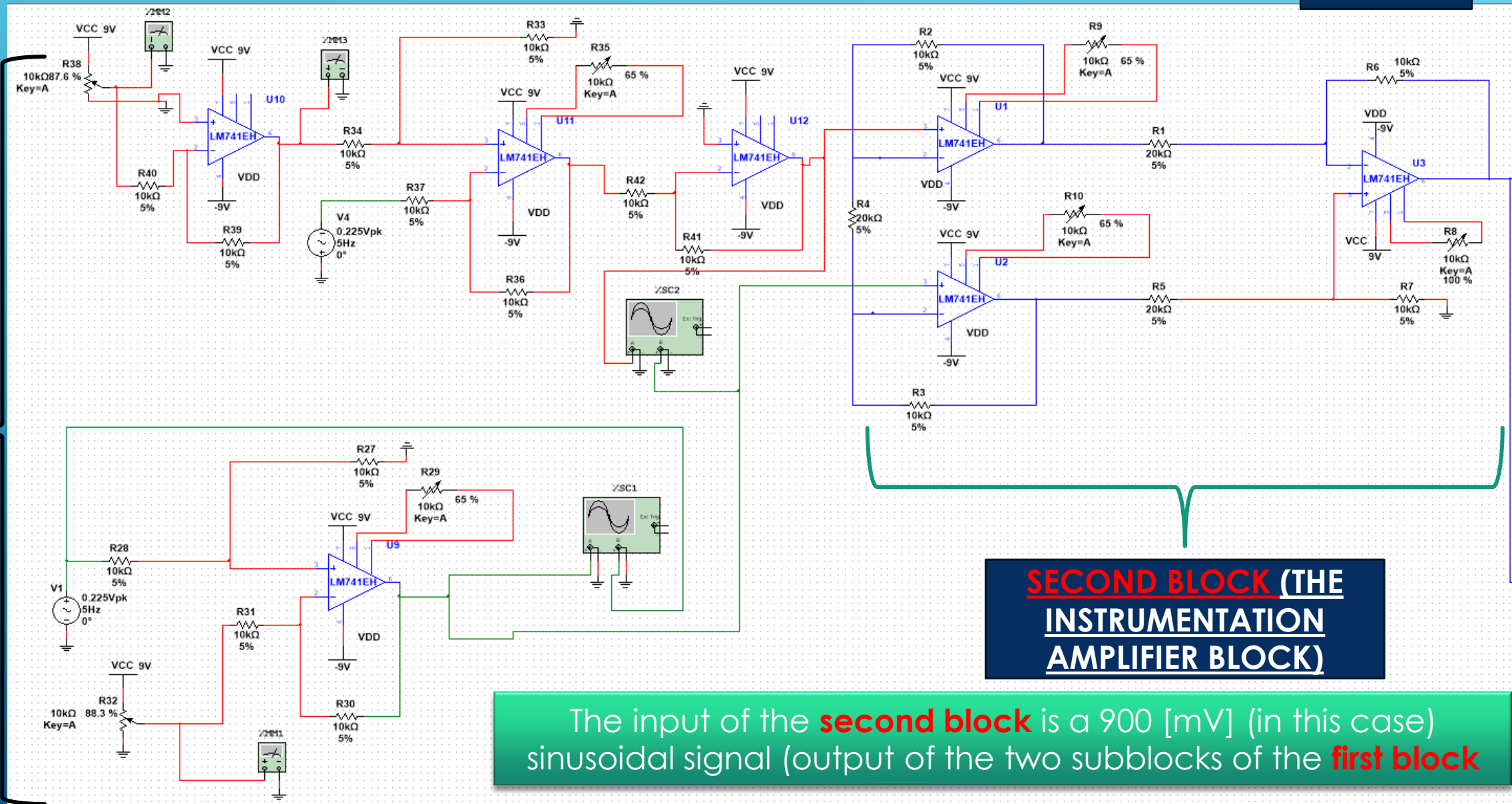


Figure 29



P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) AND THE CONNECTION TO THE SECOND BLOCK (THE INSTRUMENTATION AMPLIFIER BLOCK) – PART 1

Figure 30



FIRST BLOCK (THE LEVEL SHIFTER BLOCK)

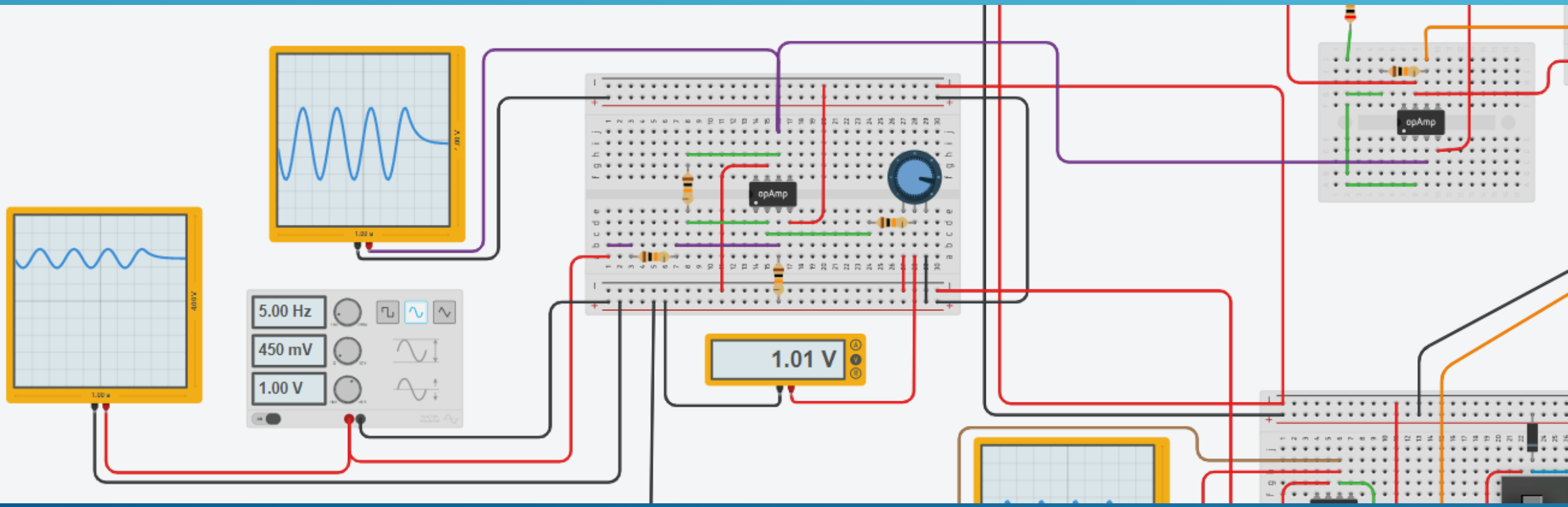
SECOND BLOCK (THE INSTRUMENTATION AMPLIFIER BLOCK)

The input of the **second block** is a 900 [mV] (in this case) sinusoidal signal (output of the two subblocks of the **first block**

P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) AND THE CONNECTION TO THE SECOND BLOCK (THE INSTRUMENTATION AMPLIFIER BLOCK) – PART 2 (TINKERCAD SIMULATION)

A closer look at the **positive** polarization AC voltage source (with DC offset) connection to the next stage – **part 3 (in TinkerCad)**

Figure 31



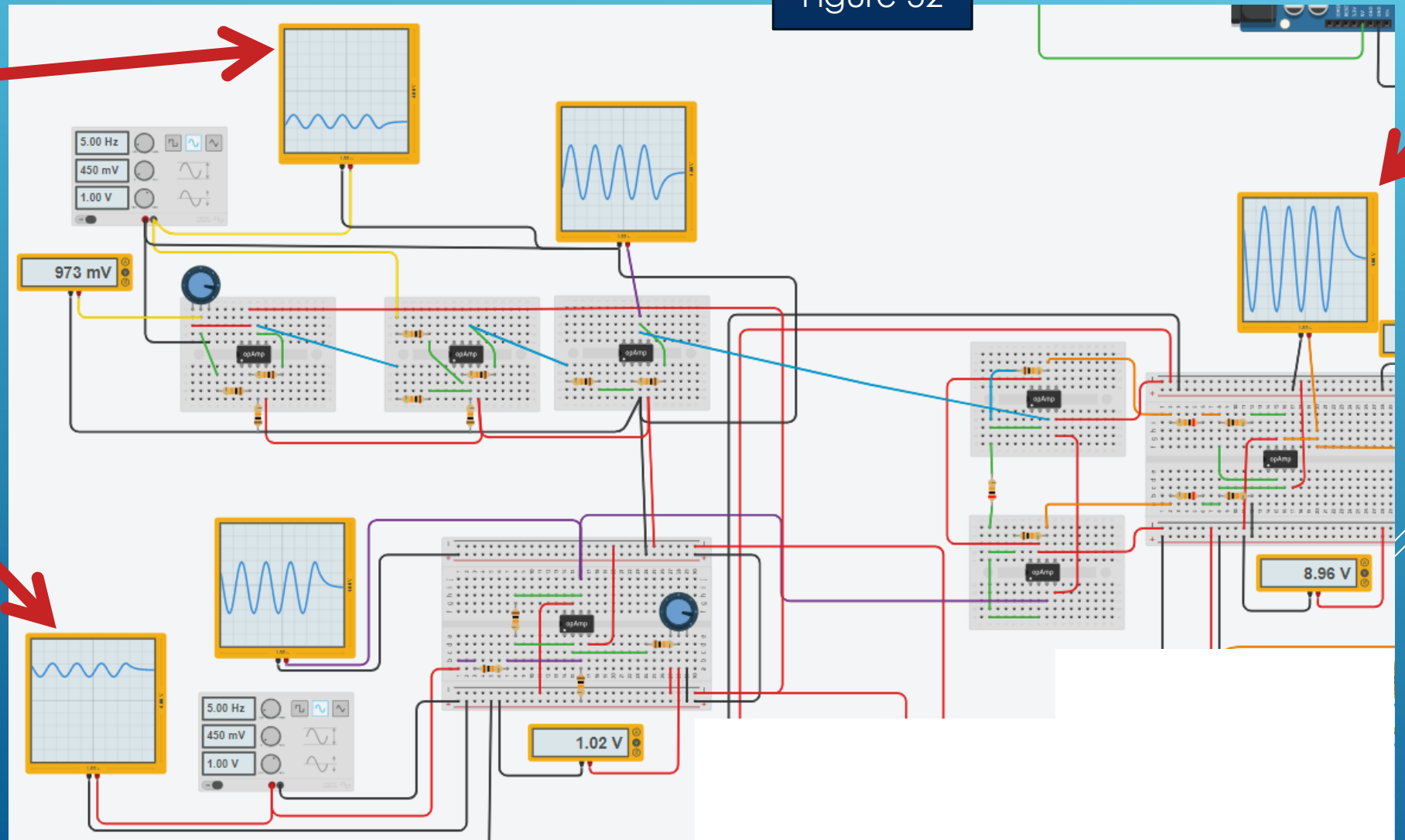
P21 - THE LEVEL SHIFTER BLOCK (FIRST BLOCK) AND THE CONNECTION TO THE SECOND BLOCK (THE INSTRUMENTATION AMPLIFIER BLOCK) – PART 3 (TINKERCAD SIMULATION)

Figure 32

negative polarized input with the offset of 1v

Positive polarized input with the offset of 1v

The full signal after second stage



P21+P23- THE INSTRUMENTATION AMPLIFIER BLOCK (SECOND BLOCK) (PART 1)

The instrumentation amplifier, shown in Figure 4, consists of two stages in cascade. The first stage is formed by op amps U1 and U2 and their associated resistors, and the second stage is the difference amplifier formed by op amp U3 and its four associated resistors.

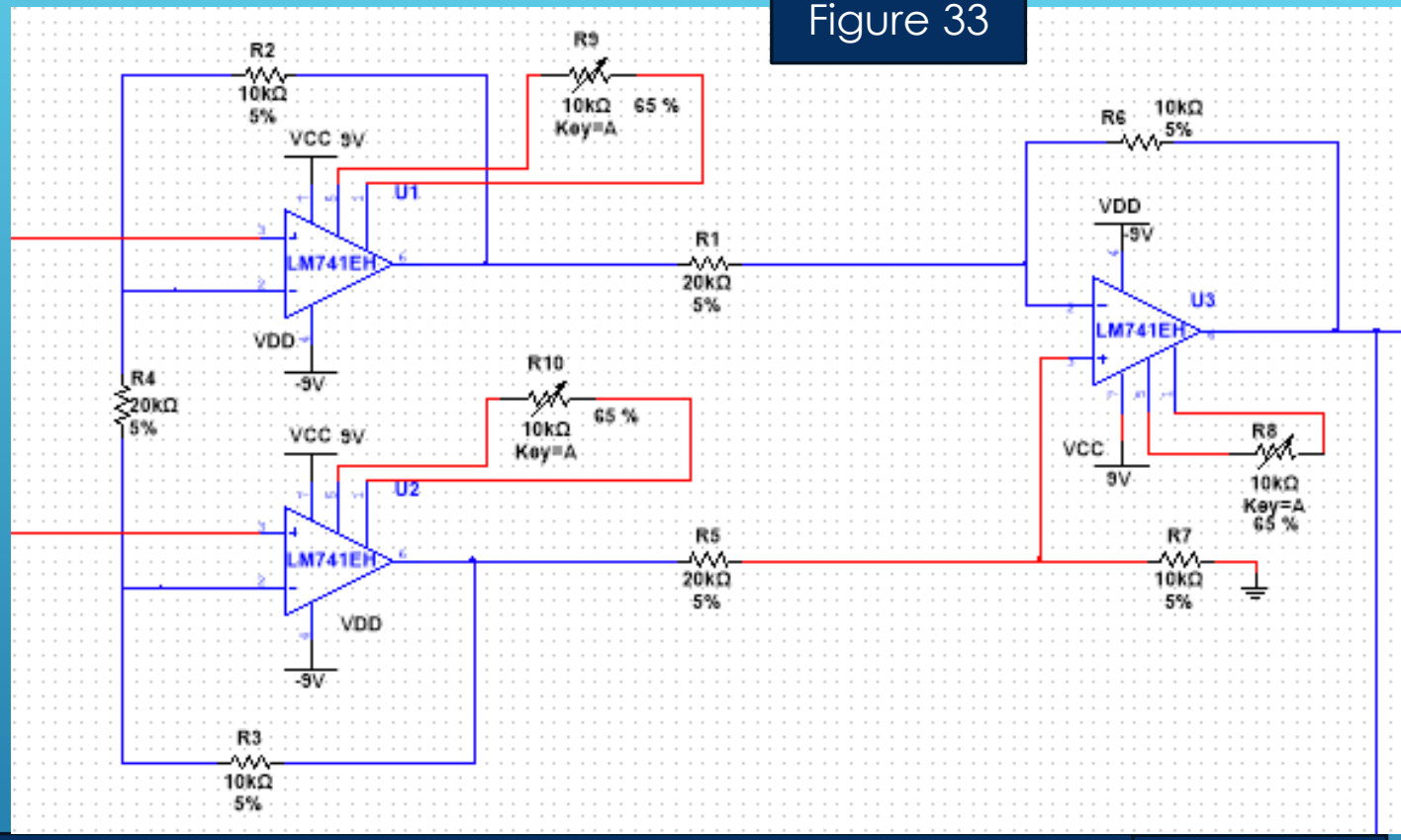
The overall differential voltage-gain is given by: $A_d = \frac{R_6}{R_1}\left(1 + \frac{2R_2}{R_4}\right) = \frac{10k}{20k}\left(1 + \frac{20k}{20k}\right) = 1$

$(R_2 = R_3 \;\; \& \;\; R_7 = R_6 \;\; \& \;\; R_5 = R_1)$

Both input terminals are connected to:

- Produce zero common mode output by U3.
- Improve the situation at input U3:
  The difference signal has been amplified by $\left(1 + \frac{2R_2}{R_4}\right)$ while the common-mode voltage remained unchanged.

Figure 33



the block's inputs are the outputs of the level shifter stage (one for each of the two inputs)

Figure 4

[2] Adel S. Sedra & Kenneth C. Smith, Microelectronic Circuits, 7th Edition, New York: Oxford University Press, July 12, 2009, pp. 82-85.

P21 - THE INSTRUMENTATION AMPLIFIER BLOCK (SECOND BLOCK) (PART 2)

Figure 34

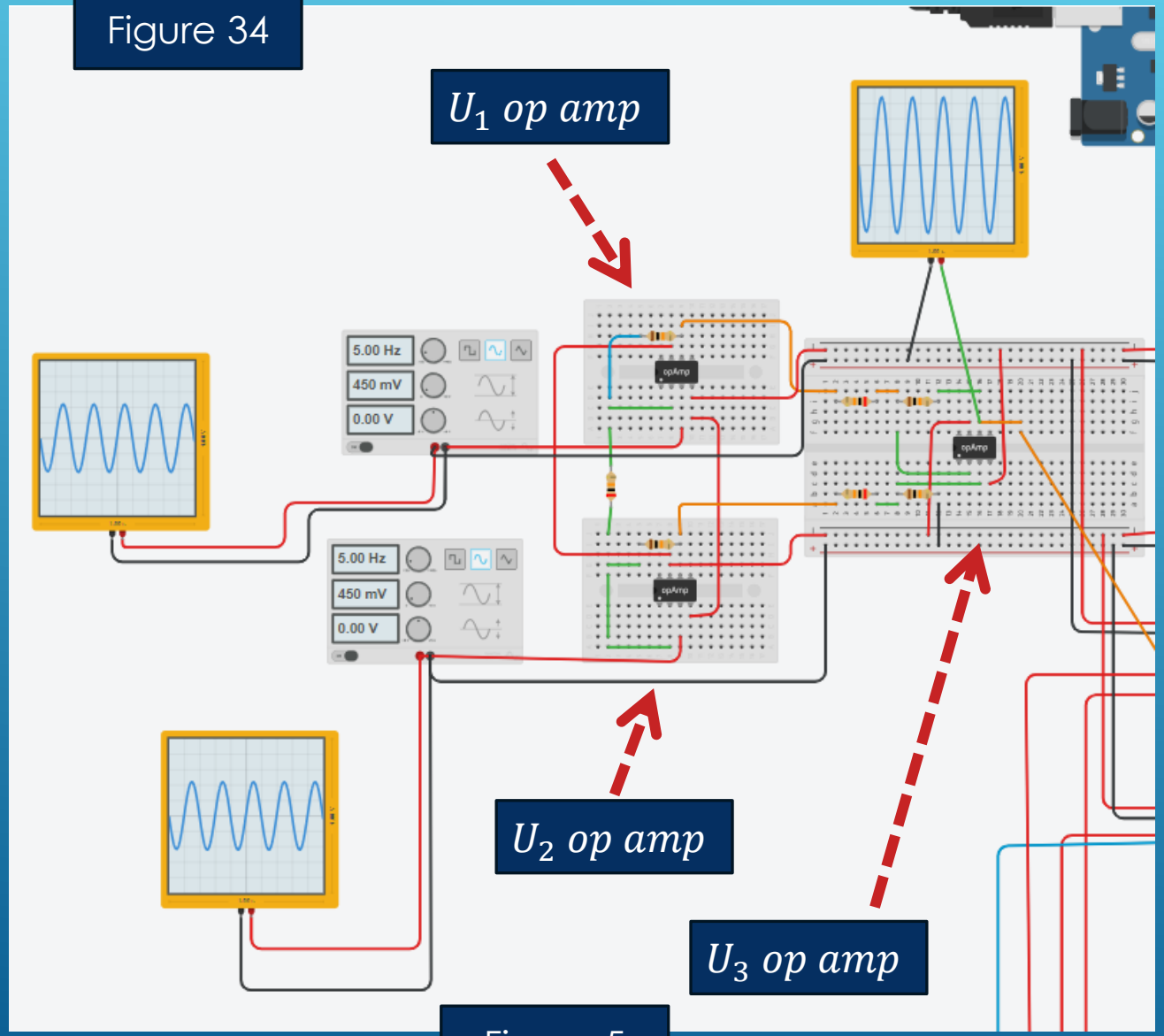


Figure 5

A picture of the TinkerCad simulation is presented here – we see the U_1 , U_2 and U_3 OP amps are clearly seen – and the scopes also.

The two left scopes show the V_2 and V_1 inputs (created in the function generators) – the output of the stage is seen in the right scope.

We create two signals – both in frequency of 5 Hz and amplitude of 450 mV (peak to peak) – but in opposite polarity – creating a 900 mV sinus signal in the output of the instrumentation amplifier (it has gain of 1)

Pay attention that this picture need to demonstrate the behavior of that block only (without the level shifter) – other pictures in the next parts of the presentation will present the full story of the TinkerCad simulation!

P21 - THE INSTRUMENTATION AMPLIFIER BLOCK (SECOND BLOCK) (PART 3)

Figure 7

V_1 input of instrumentation amplifier stage – In Tinkercad

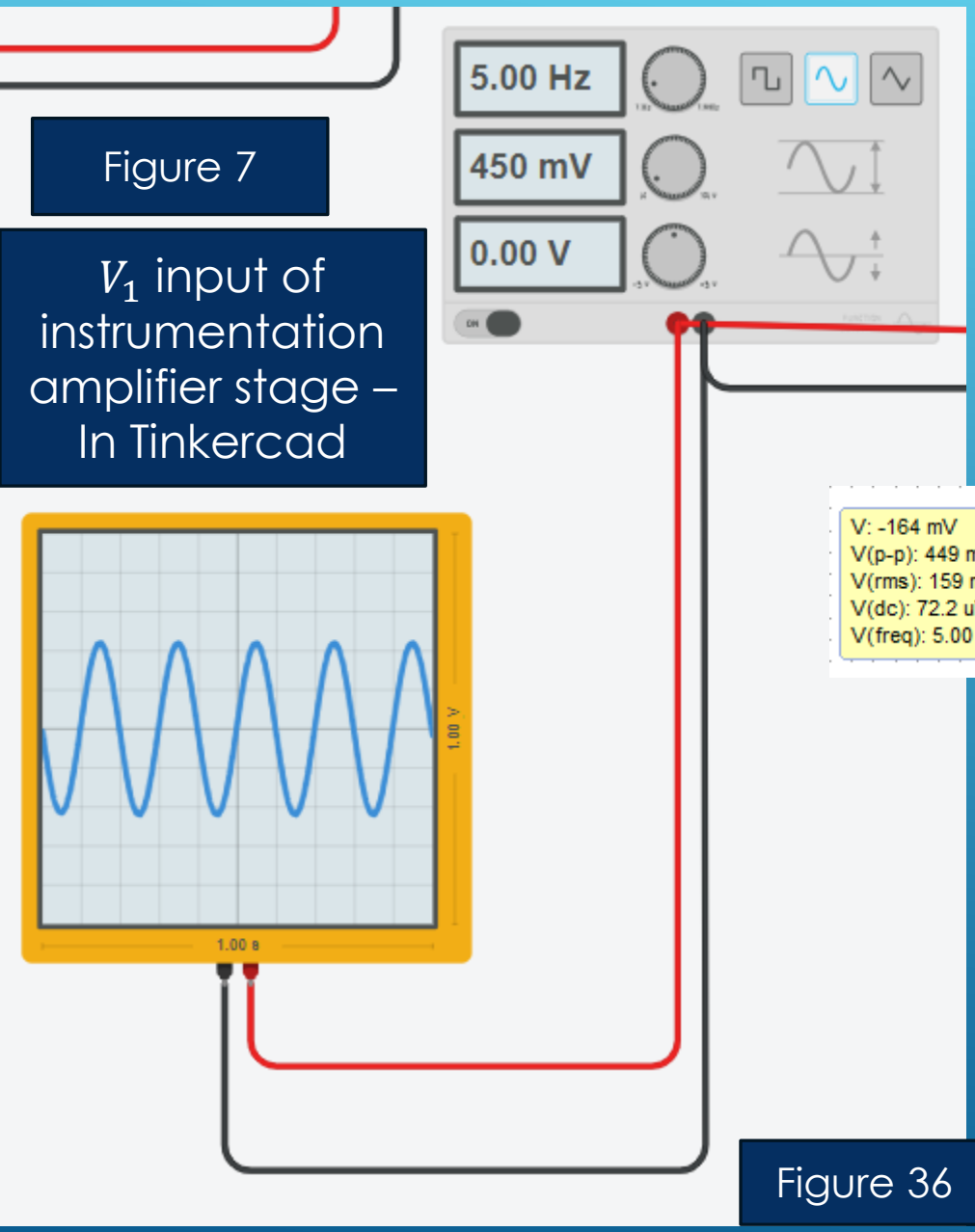


Figure 36

Figure 35

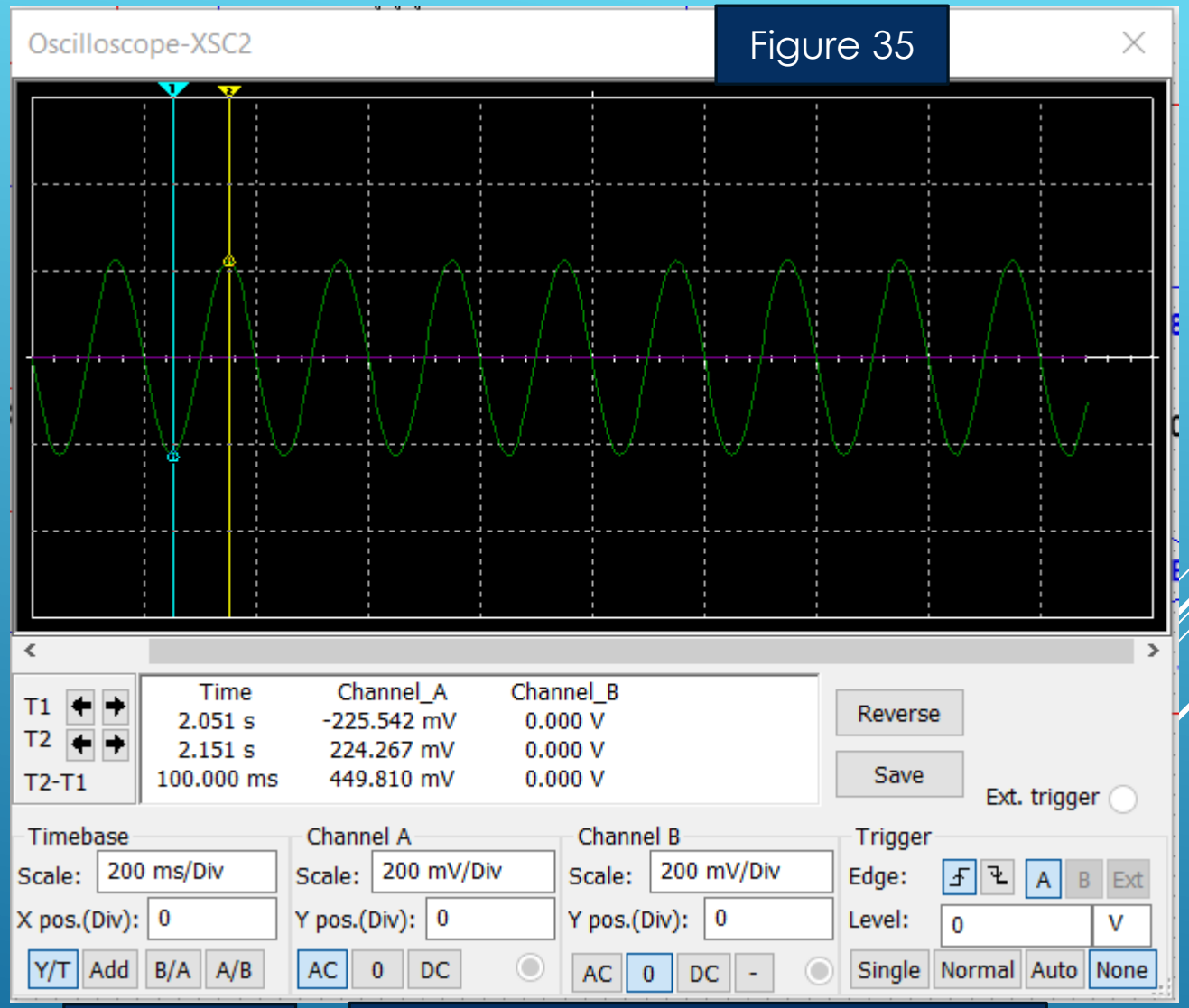
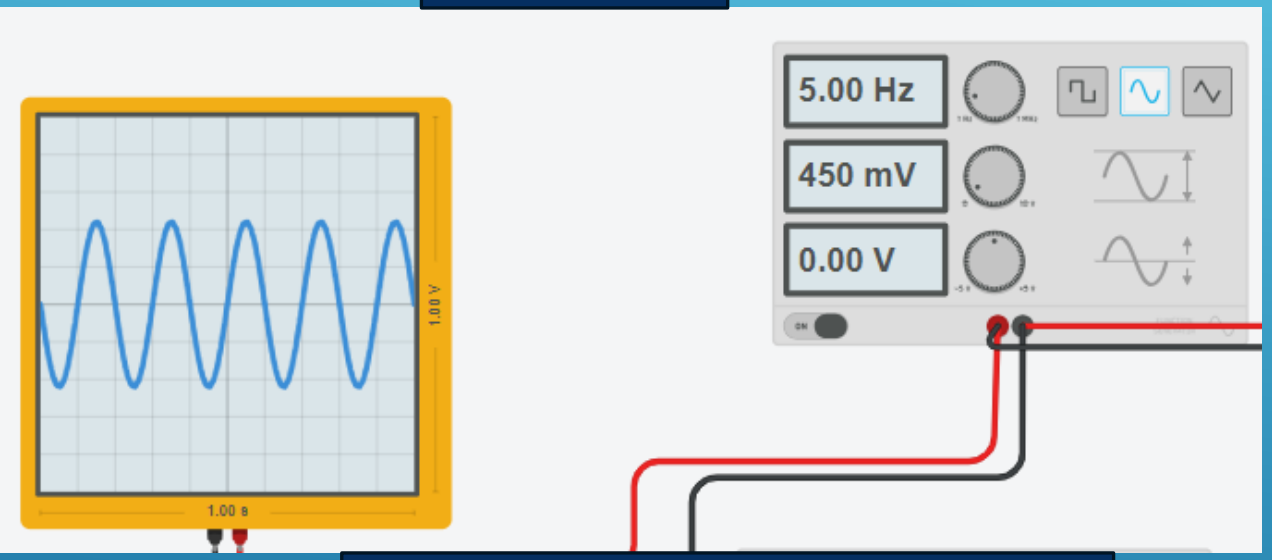


Figure 6

V_1 input of instrumentation amplifier stage – In Multisim

P21 - THE INSTRUMENTATION AMPLIFIER BLOCK (SECOND BLOCK) (PART 4)

Figure 38



V_2 input of instrumentation amplifier stage – In Tinkercad

Figure 37

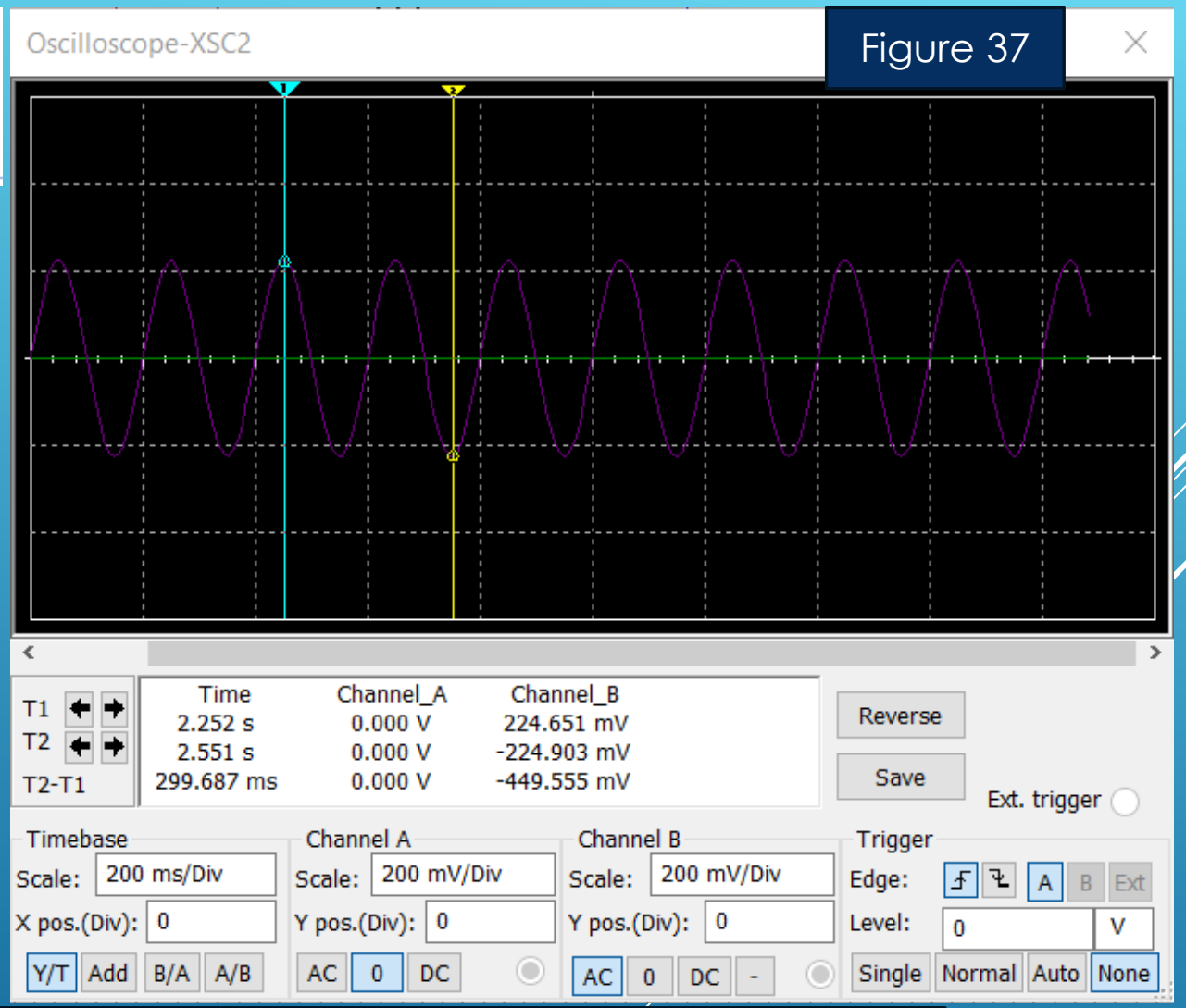


Figure 8

V_2 input of instrumentation amplifier stage – In Multisim

P21 - THE INSTRUMENTATION AMPLIFIER BLOCK (SECOND BLOCK) (PART 5)

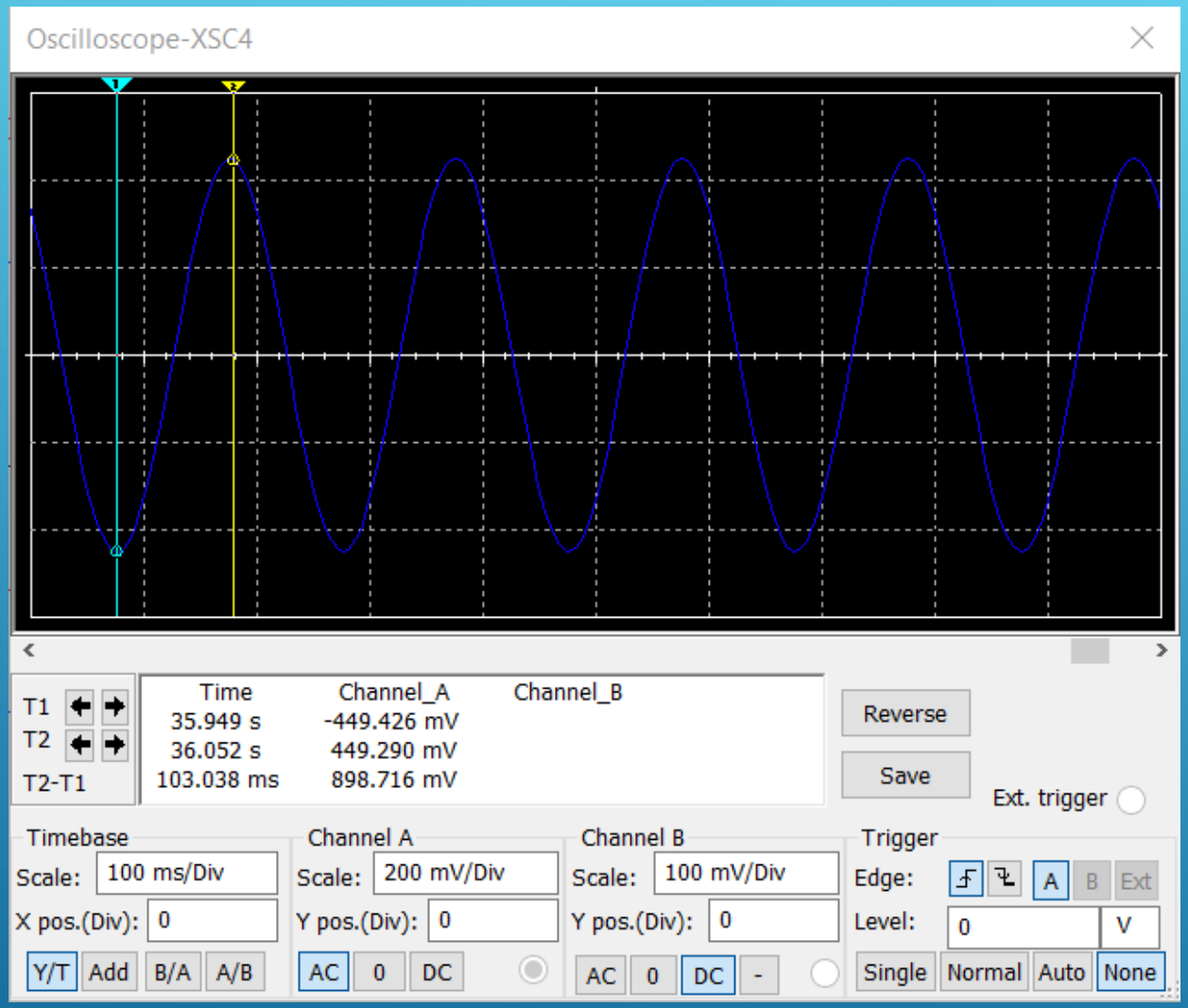
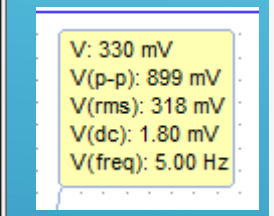


Figure 40

Output of instrumentation amplifier stage – In Multisim



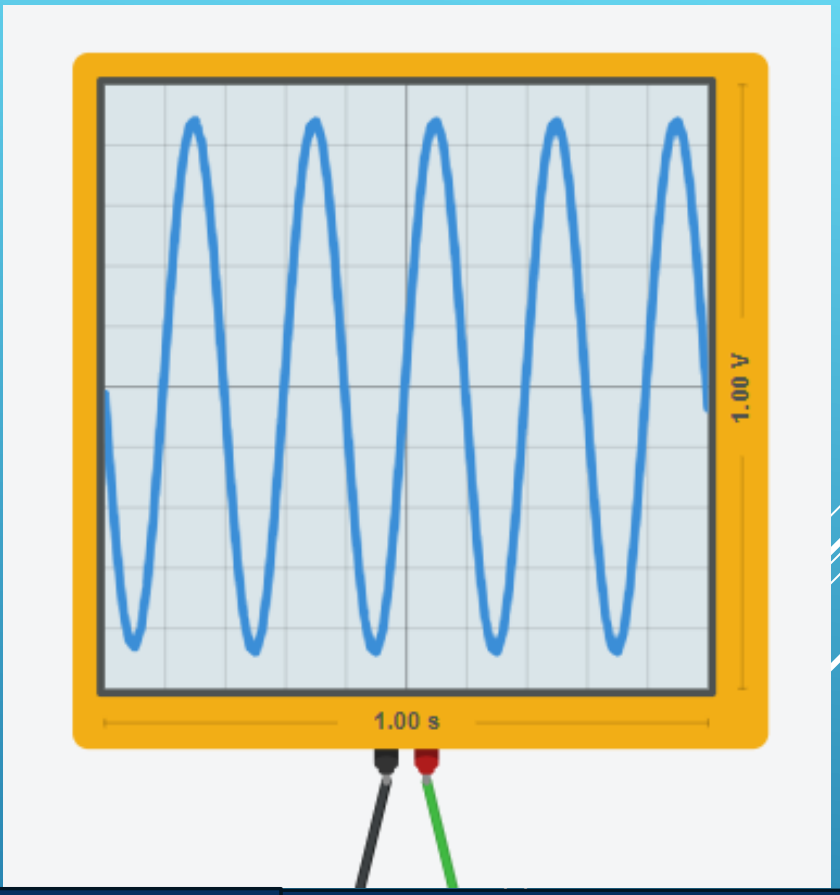


Figure 39

Output of instrumentation amplifier stage – In Tinkercad

P21+P23- THE "SUPERDIODE" BLOCK (THIRD BLOCK) (PART 1)

The improved "Superdiode" rectifier converts AC to DC while not losing 0.7V on the diodes.

In this case, when the input is greater than zero, D_2 is off and D_1 is on, so the output is zero because the other end of R_{11} is connected to the virtual ground and there is no current through R_{11}.

When the input is less than zero, D_2 is on and D_1 is off, so the output is like the input with an amplification of $-\frac{R_{11}}{R_{12}} = -\frac{215k}{150k} = -\frac{43}{30} \approx -1.433$

We use a capacitor filter in order to convert the pulsating direct current into pure direct current.

In the next slide we will present a Multisim and TinkerCad simulation of the Precision Half-Wave Rectifier output with and without connecting to the capacitor filter…

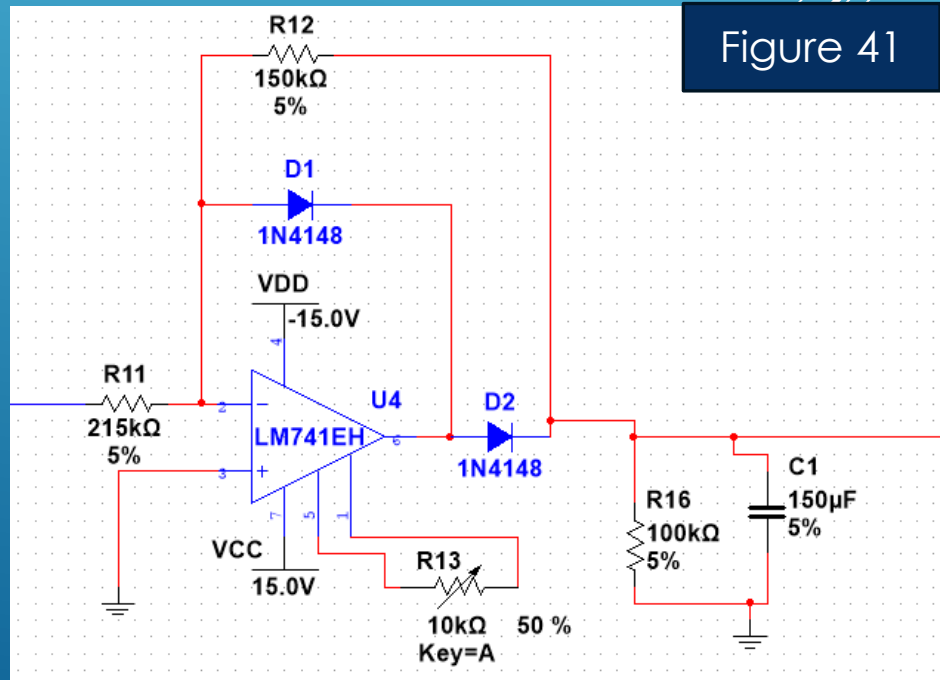
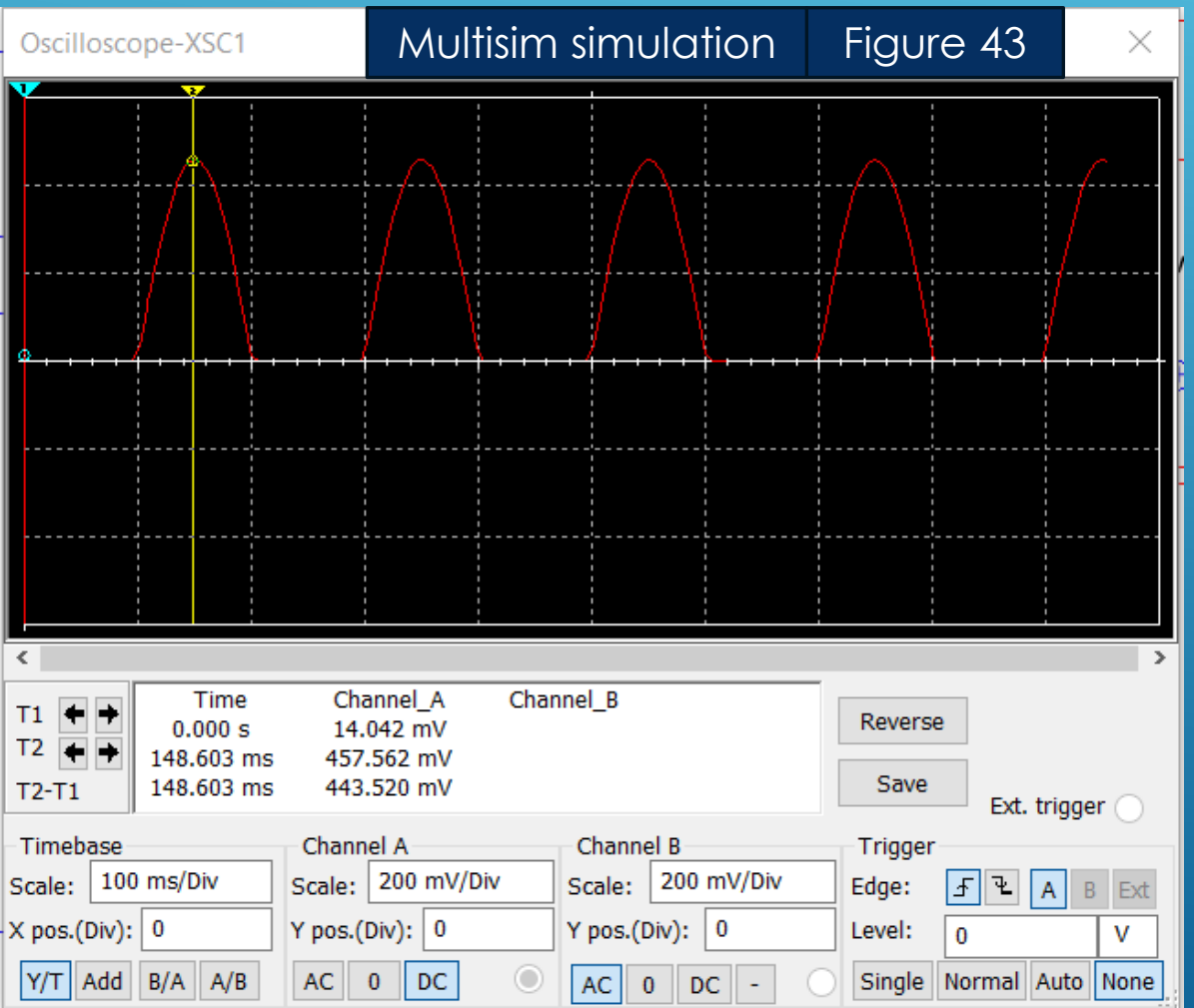


Figure 41

P21 - THE "SUPERDIODE" BLOCK (THIRD BLOCK) (PART 2)

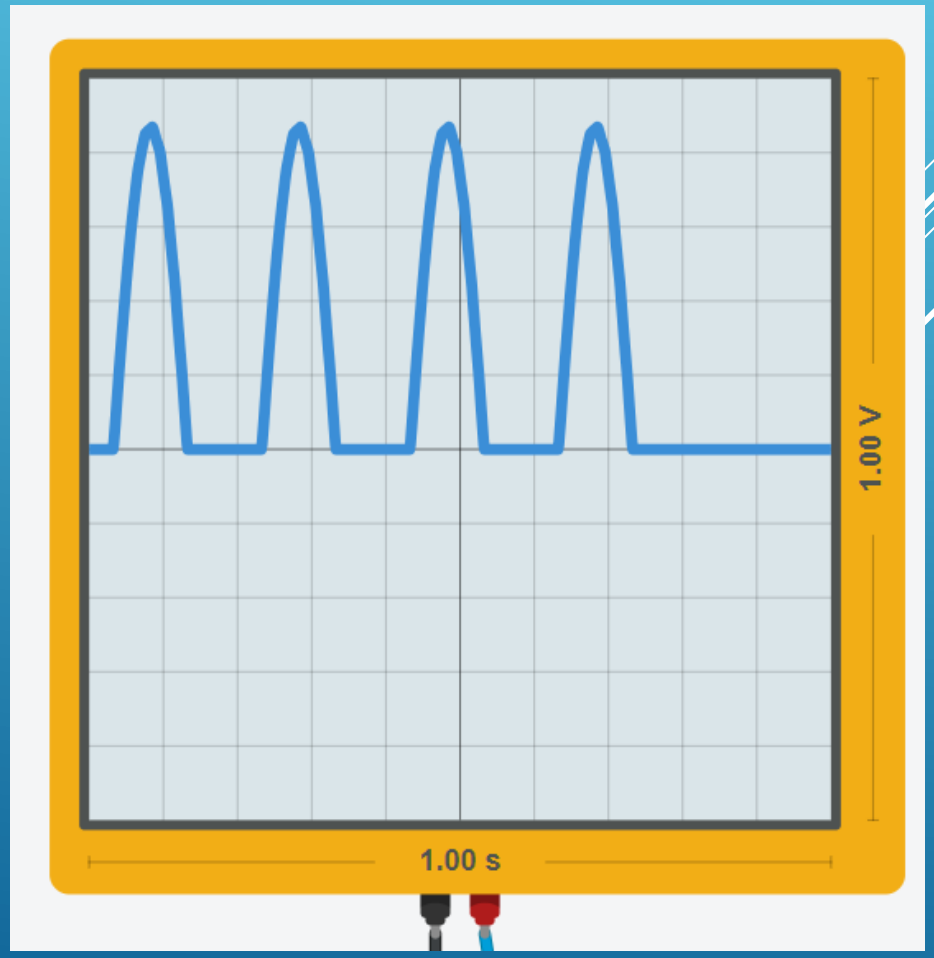
In figure 9 and 10 we can see that when we don't use the capacitor and resistor filter, we will see a DC wave with ripples at the output of the "Superdiode".

Multisim simulation | Figure 43



In the next slide – the effect of using the capacitor-resistor filter – no ripples

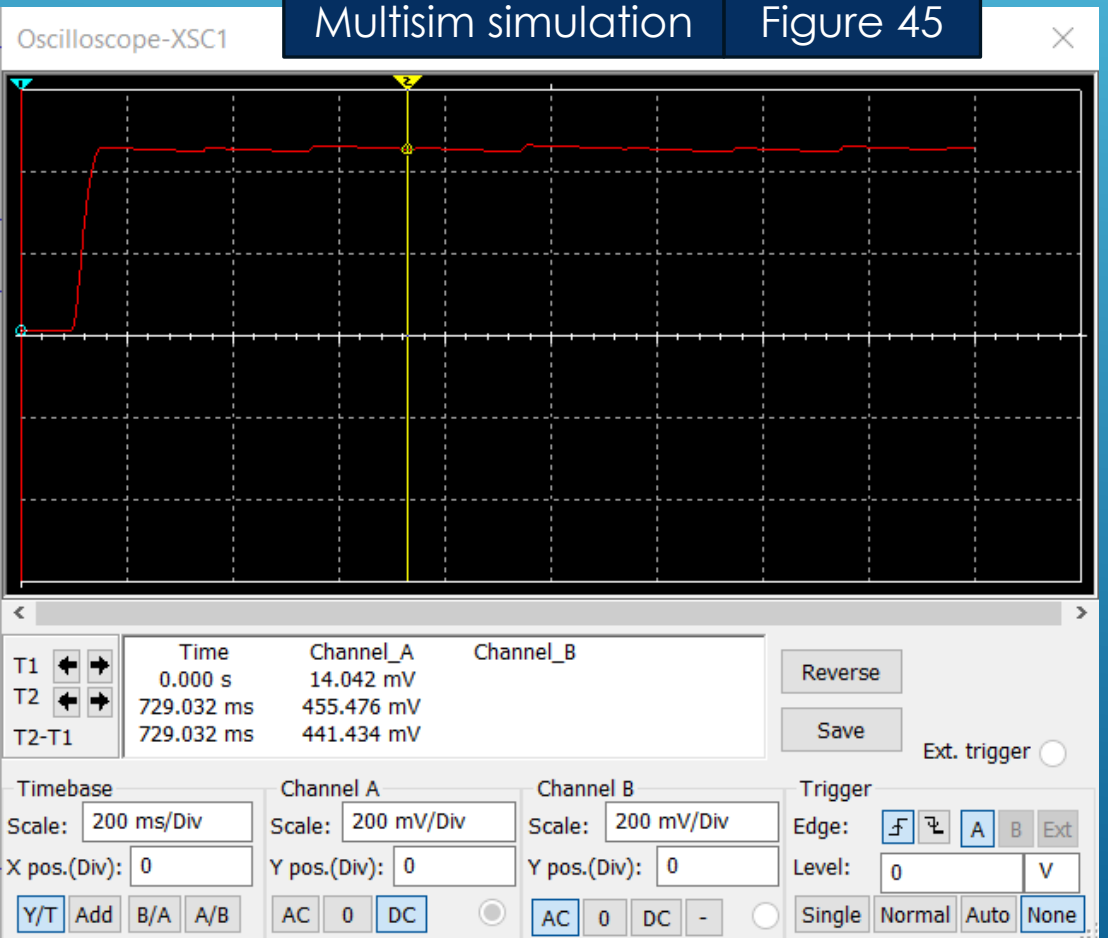
TinkerCad simulation | Figure 42



P21 - THE "SUPERDIODE" BLOCK (THIRD BLOCK) (PART 3)

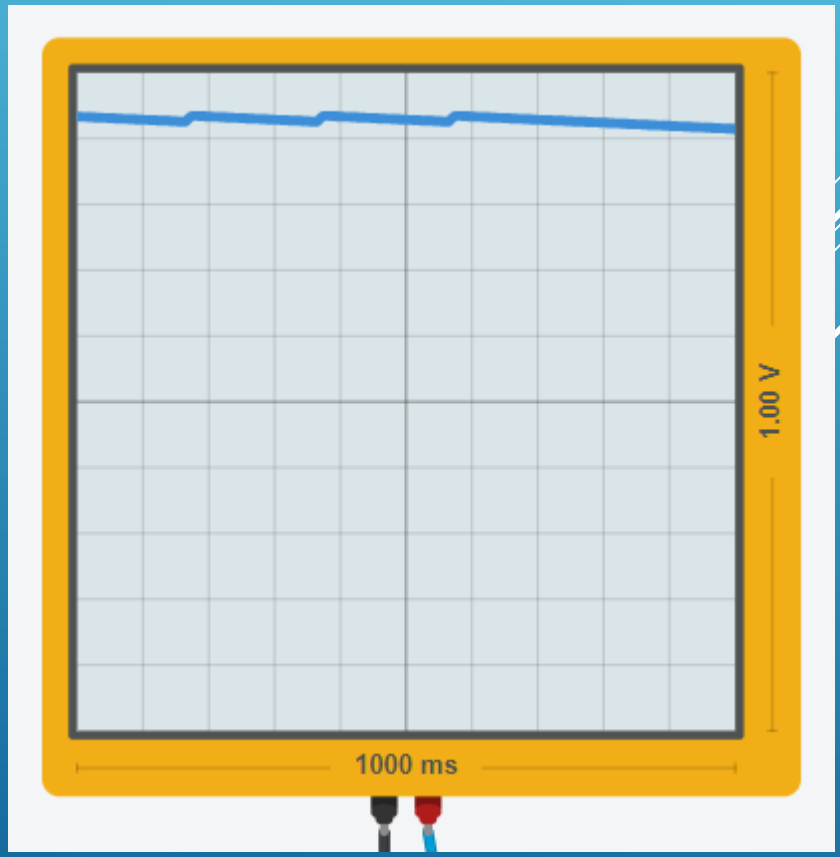
In Figure 11 and 12 we can see that when we use a capacitor and resistor filter we receive a suppressed DC wave signal at the output of the "Superdiode", as expected.

Multisim simulation | Figure 45



So we can see that the capacitor-resistor filter gives us a DC waveform – almost ready to be sampled by the Arduino

TinkerCad simulation | Figure 44

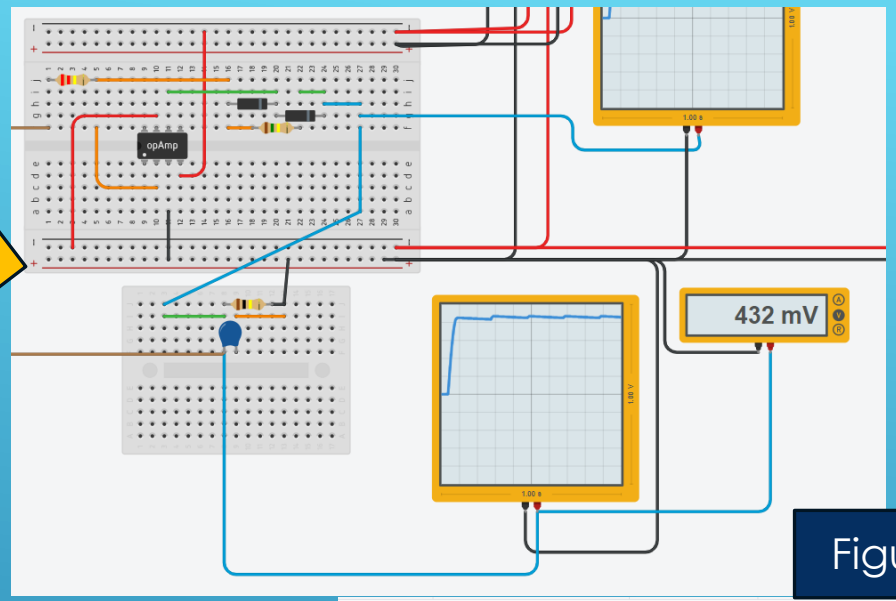


P21 - THE "SUPERDIODE" BLOCK (THIRD BLOCK) (PART 4)

"Classical approach" (without Arduino) – "superdiode block"

Figure 46

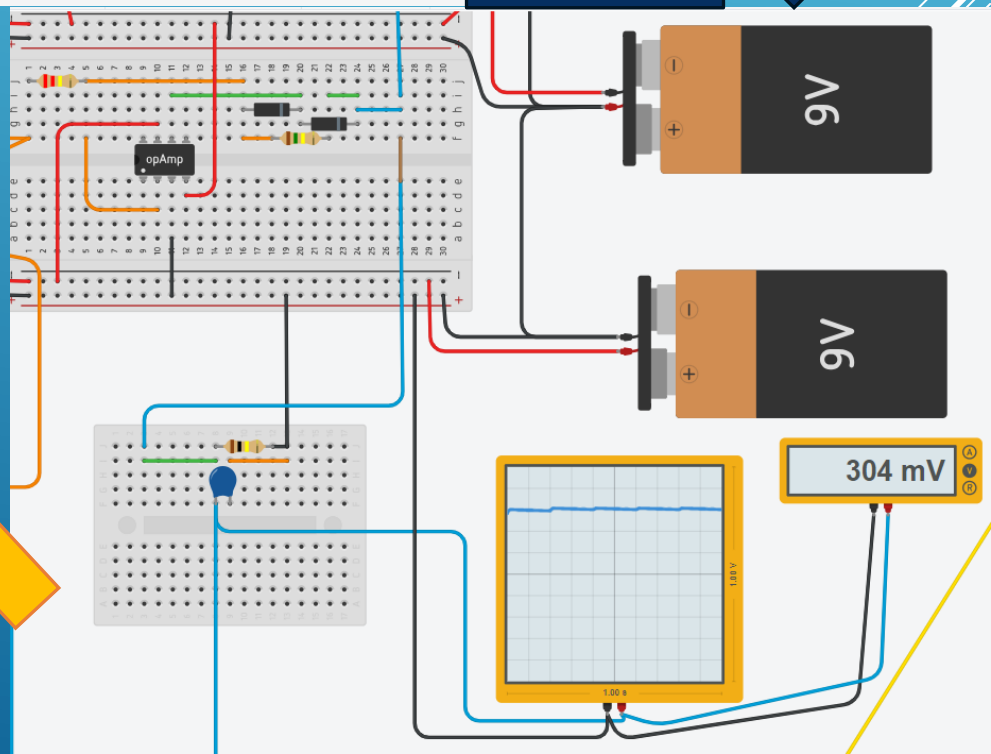
Upper graph



"advanced approach" (with Arduino) – "superdiode block"

Figure 47

lower graph



P21+P23- THE "SUPERDIODE" BLOCK (THIRD BLOCK) (PART 5)

Capacitor calculation:

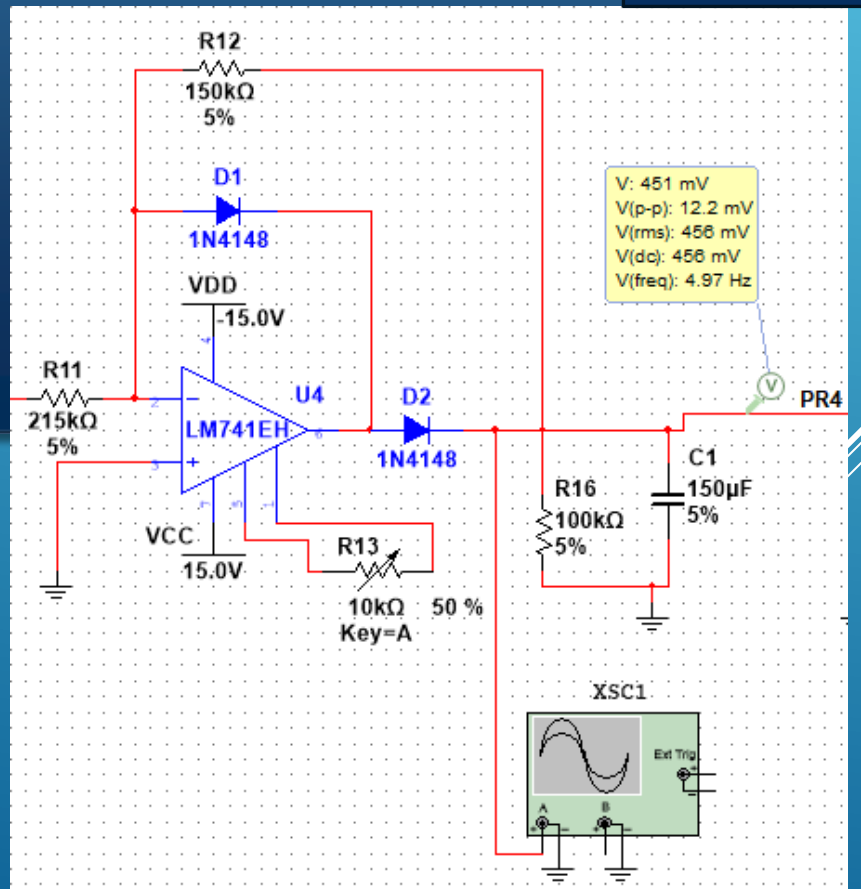
In order to keep V_{ripple} small we must select a capacitance C so that $CR \gg T$.

The ripple voltage V_{ripple} can be expressed in terms of the frequency $f = \frac{1}{T}$ as:

$$V_{ripple} = \frac{V_{peak}}{f \cdot C \cdot R}$$

$$C = \frac{V_{peak}}{f \cdot R_T \cdot V_{ripple}} = \frac{3.535[V]}{5[Hz] \cdot 60[k\Omega] \cdot 76.5[mV]} = 150[\mu F]$$

Figure 48



P21 - THE UNITY-GAIN BUFFER BLOCK (FOURTH BLOCK) (PART 1)

We placed the diode to clamp voltages higher than 5V and prevent damage to the Arduino digital pin A0.

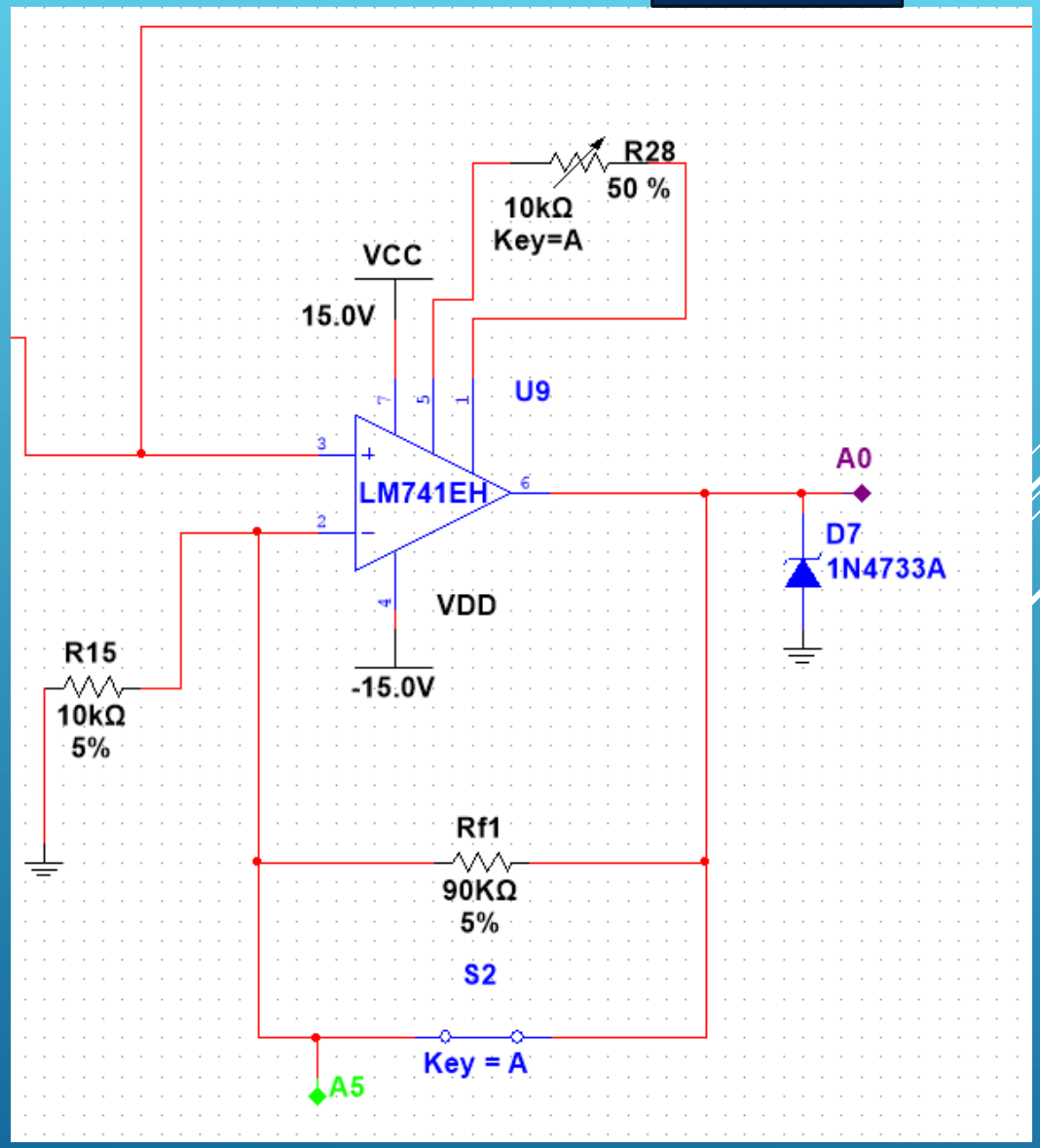
When the switch is closed, we receive an op-amp based unity-gain buffer.

Unity gain buffer is one of the simplest uses of an operational amplifier, where the output voltage is exactly same as the input voltage applied to the circuit. In other words, the gain of a unity gain buffer circuit is 1:

$$A_{CL} = 1$$

In practice, the output voltage of a unity gain buffer will not be exactly equal to the input voltage applied and there will be a slight difference. This difference is due to the high internal voltage gain of the op-amp.

Figure 49



P21 - THE UNITY-GAIN BUFFER BLOCK (FOURTH BLOCK) (PART 2)

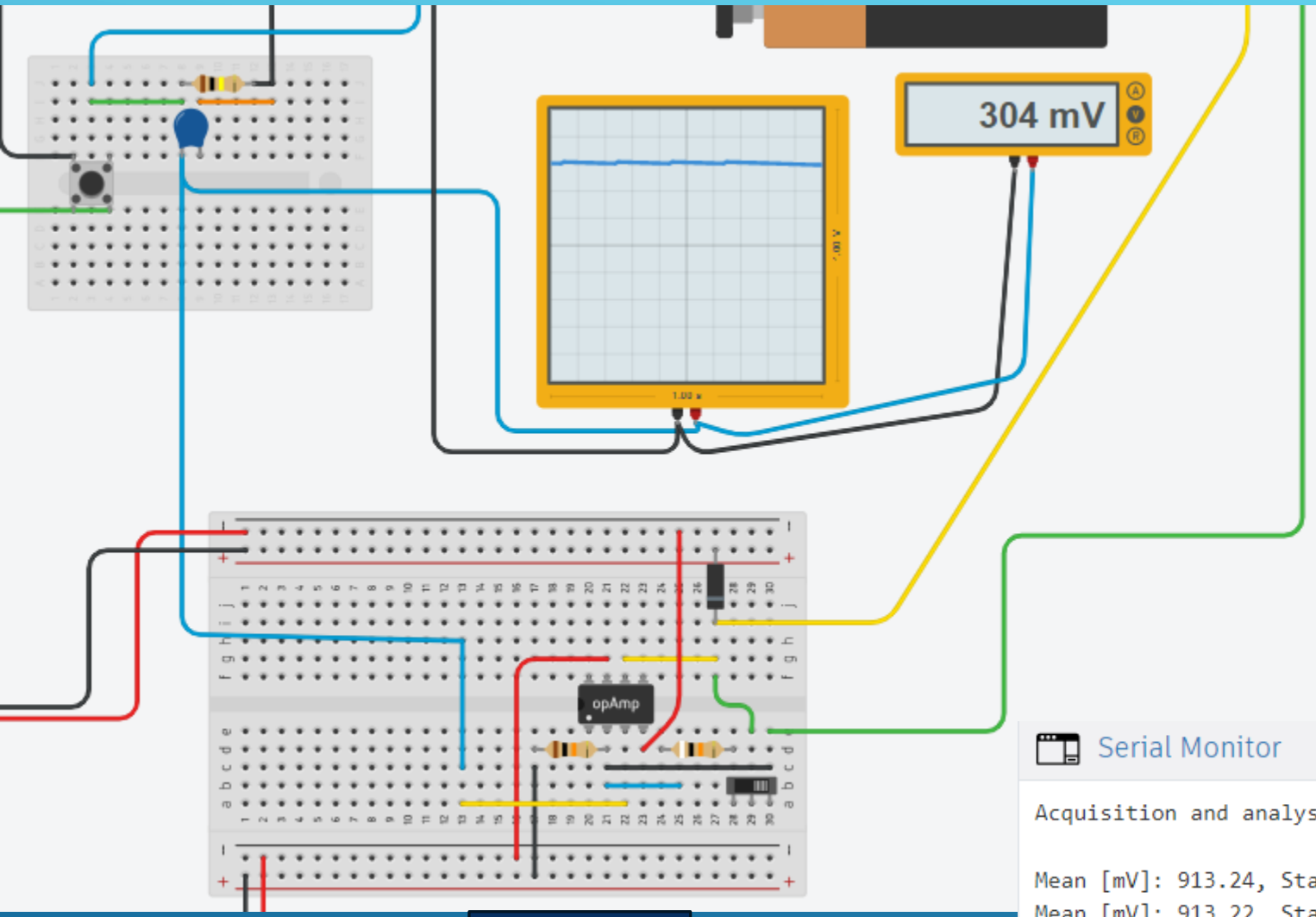
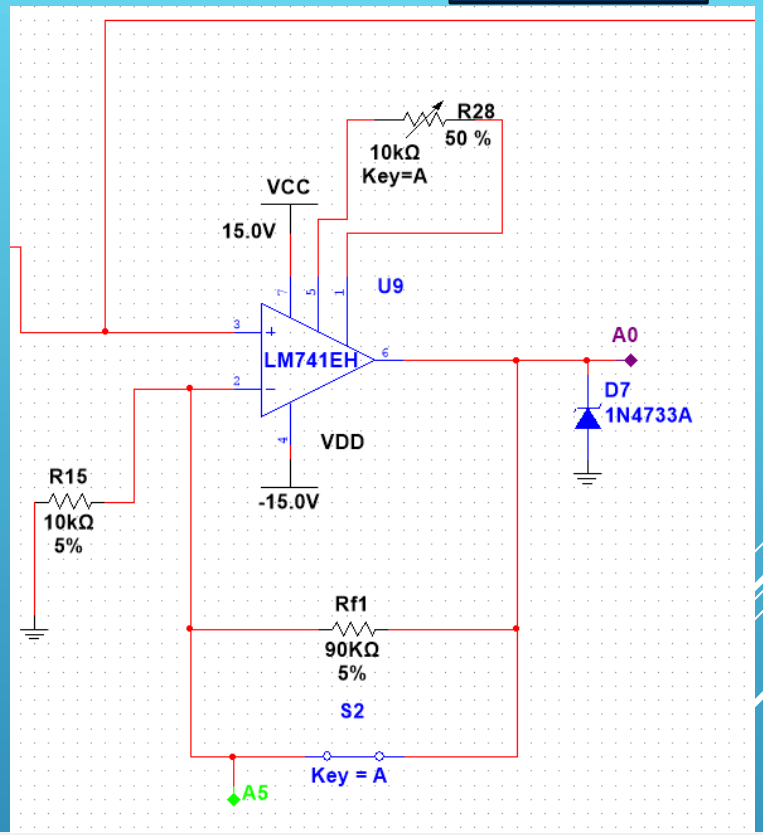


Figure 51

Figure 50



Serial Monitor

```
Acquisition and analysis are going on now

Mean [mV]: 913.24, Standard Deviation [mV]: 6.35
Mean [mV]: 913.22, Standard Deviation [mV]: 6.37
Mean [mV]: 913.21, Standard Deviation [mV]: 6.38
Mean [mV]: 913.21, Standard Deviation [mV]: 6.39
Mean [mV]: 913.21, Standard Deviation [mV]: 6.39
Acquisition and analysis finished

Mean [mV]: 913.21, Standard Deviation [mV]: 6.39
```

Figure 52

P21 - MICRO-CONTROLLER: ARDUINO UNO R3 (FIFTH BLOCK) (PART 1)

The Arduino Uno is an open-source microcontroller board based on the Microchip Atmega328P microcontroller.

The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.[12]

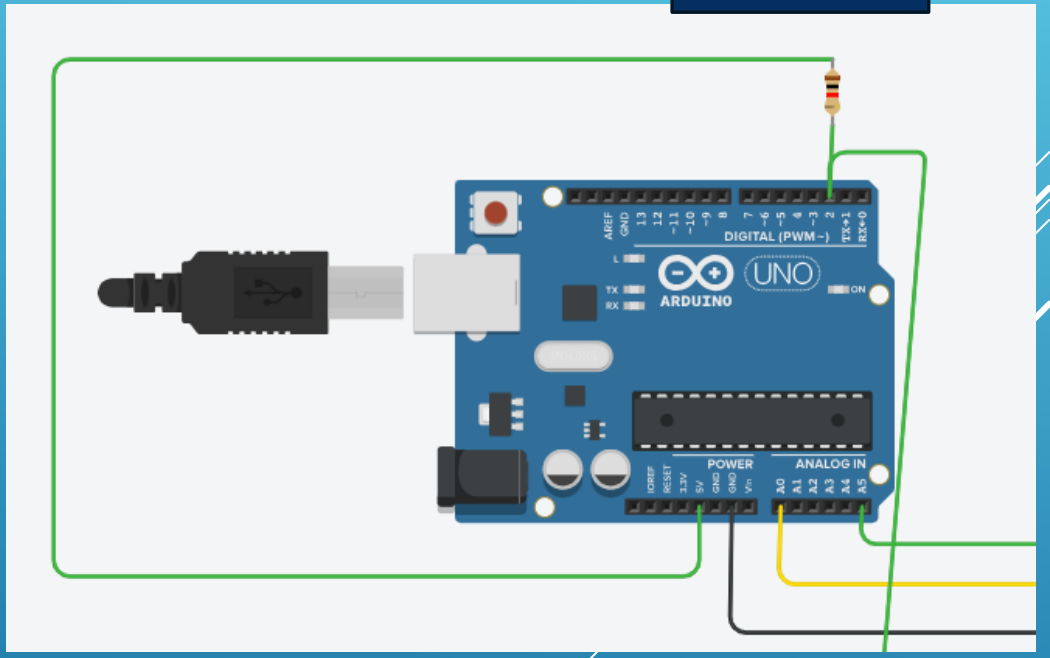
In our project, we used digital pin A5 as the Analog input that controls relay for the voltage permitted ranges.
And also - Analog pin A0 that gets amplified voltage from non-inverting amplifier block that fits to the "Full Scale" of the A/D.

The resolution of the A/D is: $Res = \frac{V_{ref}}{2^n} = \frac{5}{2^{10}} \approx 4.9[\text{mV}]$ per unit

The purpose of the amplifier is to make the whole range of the A/D available, we get the wanted result by mapping the MAX value of each range to the FSR voltage of the A/D which is 5V:
$0.5V \rightarrow 5V$ Theoretically achieved when:

$$0.5 \cdot A_v = 5 \rightarrow \frac{5}{0.5} = 1 + \frac{R_f}{10K} \rightarrow R_f = 90\ [k\Omega]$$

Figure 53



P21 - MICRO-CONTROLLER: ARDUINO UNO R3 (FIFTH BLOCK) (PART 2)

First of all the "microcontroller" reads the voltage on the A0 pin.
The voltage from A0 let the microcontroller do the initial selection.

Here the FSR is $5[V]$ while the we already know that the resolution is about $4.9[mV]$ so we get:

$$RAW = [\frac{V_{A0}}{4.9[mV]\ per\ unit}] \quad \rightarrow \quad [\frac{500[mV]}{4.9[mV]\ per\ unit}] = 102$$

The value of 102 represent the digital value of 500[mV] of measured voltage,
and will help us in determining the required range.
For the first range we check values between: 0-102
For the second range we check values between: 103-1023
Initially, Switch S1 is closed,
If the "RAW" value is between 0-102 and Switch S1 is Closed, the microcontroller will Open the switch S1, Else if the "RAW" value is greater than 102 and the Switch S1 is Open the microcontroller will Close the Switch S1.

P21 - MICRO-CONTROLLER: ARDUINO UNO R3 (FIFTH BLOCK) (PART 3)

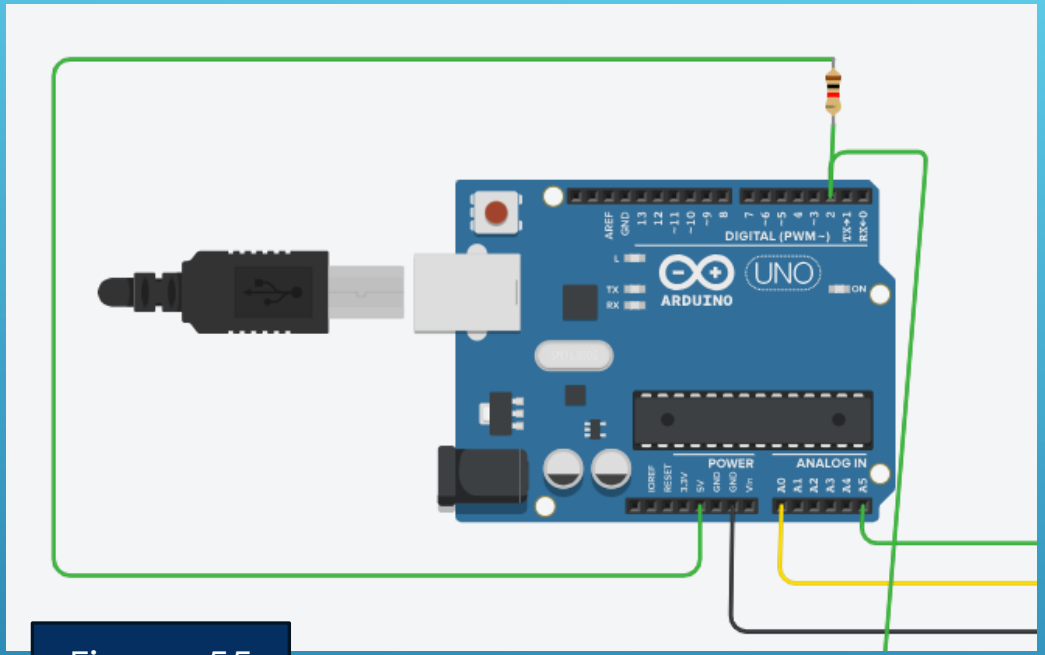


Figure 55

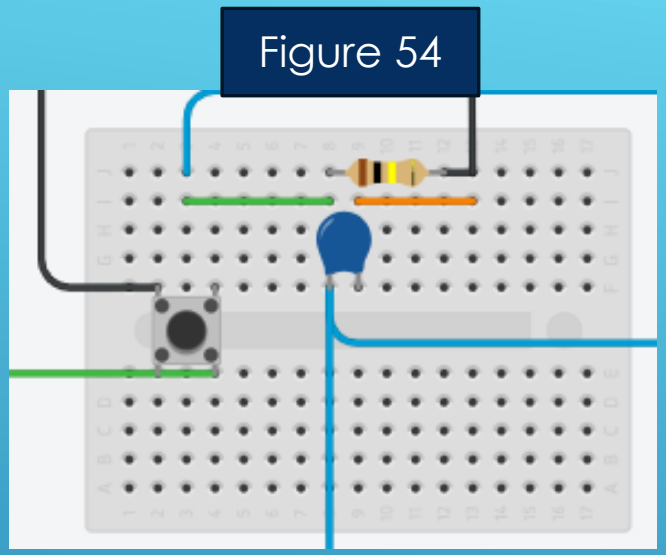


Figure 54

In addition to the acquisition of the voltage from the A0 and A5 analog inputs – we had to add a push button – to instruct the Arduino UNO when to sample the voltage using a timer and an hardware interrupt

```
22    //set Button as External Interrupt with change_mode function
23    attachInterrupt(digitalPinToInterrupt(button_pin), myISR,FALLING);
```

P21 - THE NON-INVERTING OPERATIONAL AMPLIFIER BLOCK (SIXTH AND SEVENTH BLOCK)

(PART 1)

Pay attention – that block is relevant only to the "classic" approach we did in Multisim – Pay attention that in the Arduino approach ("advanced approach") the measurement ends in the fourth block (unity gain buffer with output to A5 ana A0 of the Arduino)

Figure 56

Fifth block-amplify the signal by sqrt(2) in order to get the RMS value

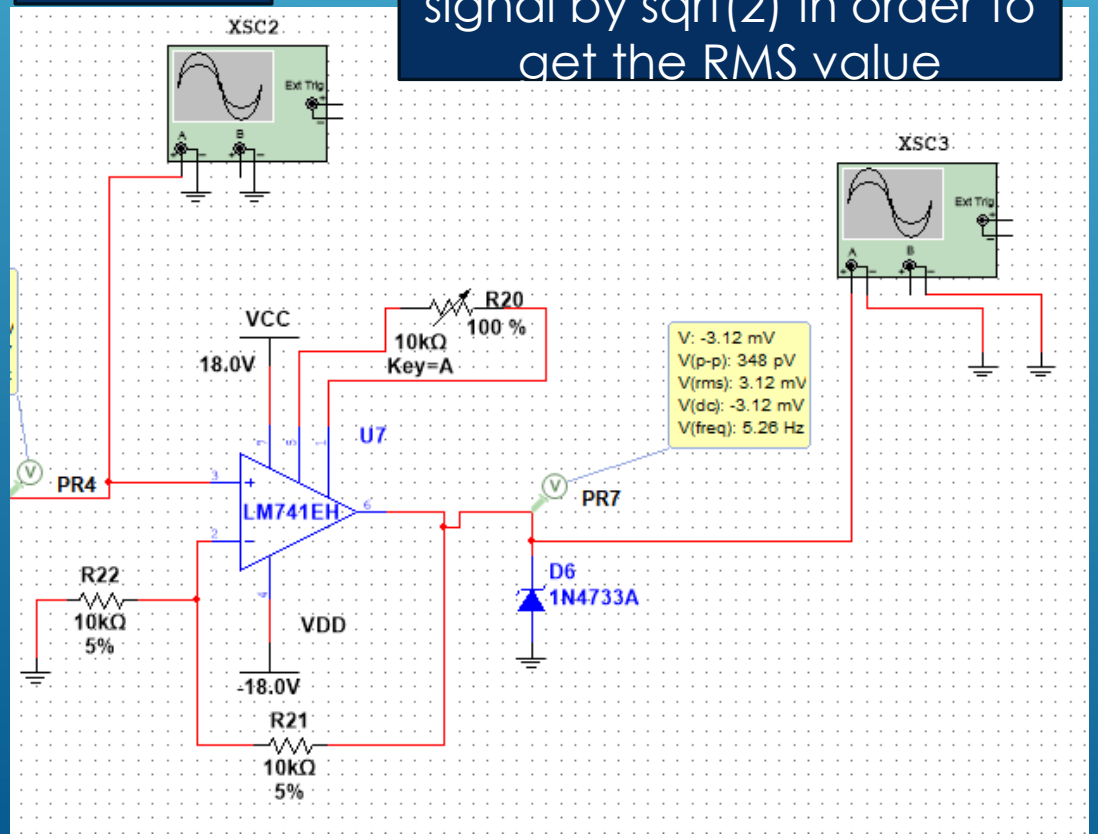
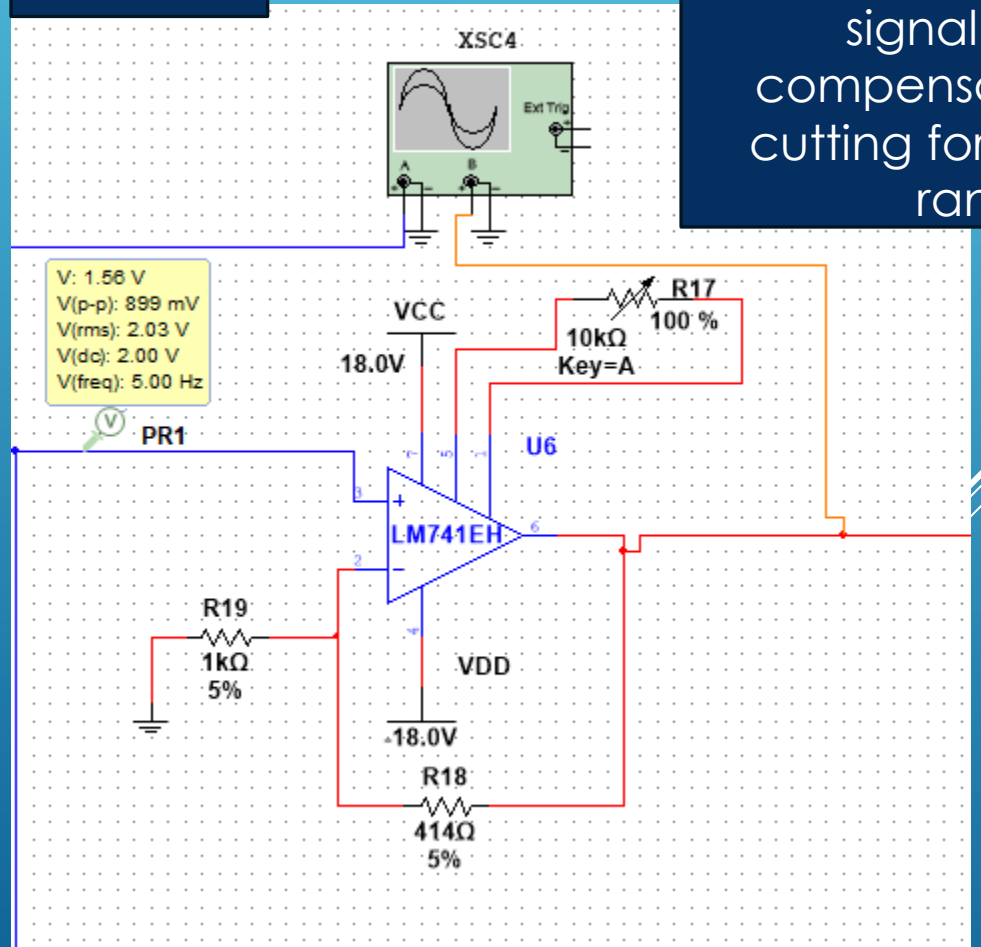


Figure 57

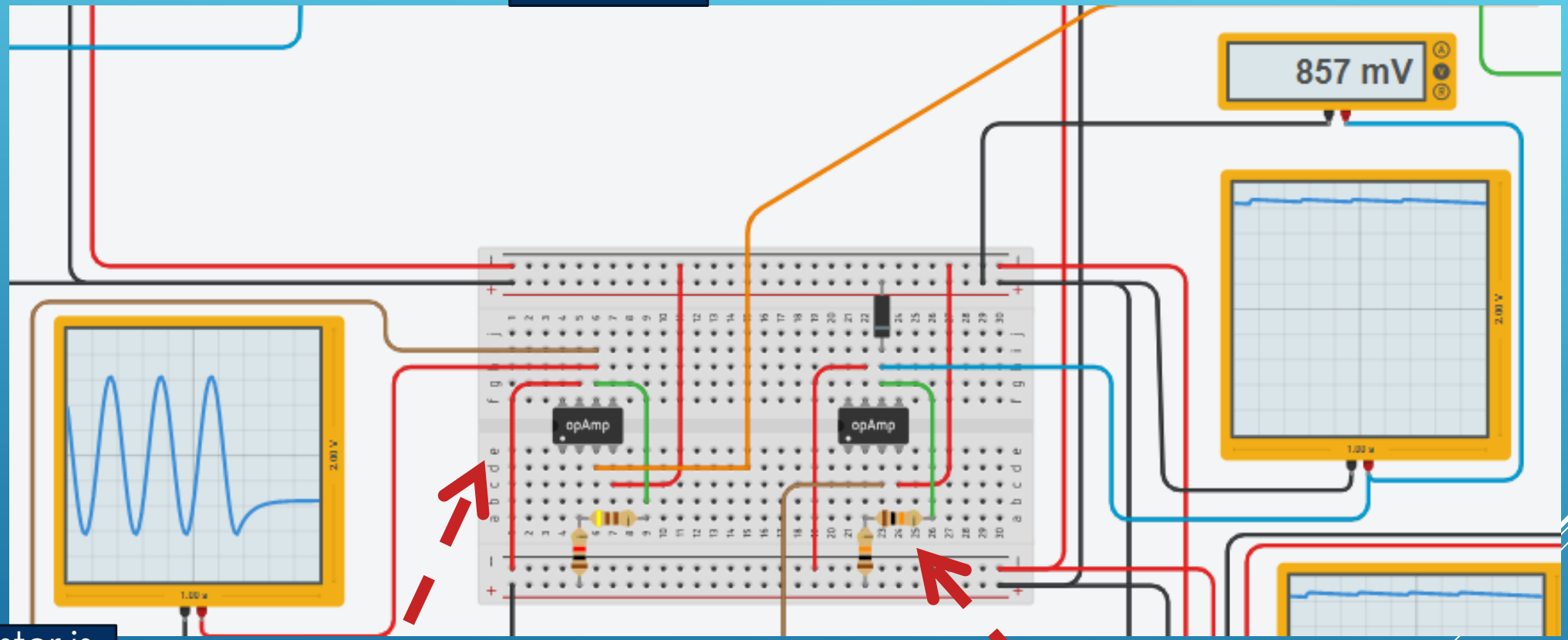
Sixth block – amplify the signal by 2 – compensate on the cutting for the entire range



P21 - THE NON-INVERTING OPERATIONAL AMPLIFIER BLOCK (SIXTH AND SEVENTH BLOCK)

(PART 2)

Figure 58



One resistor is 1k and the other is 414 ohm for the sqrt(2) amplification

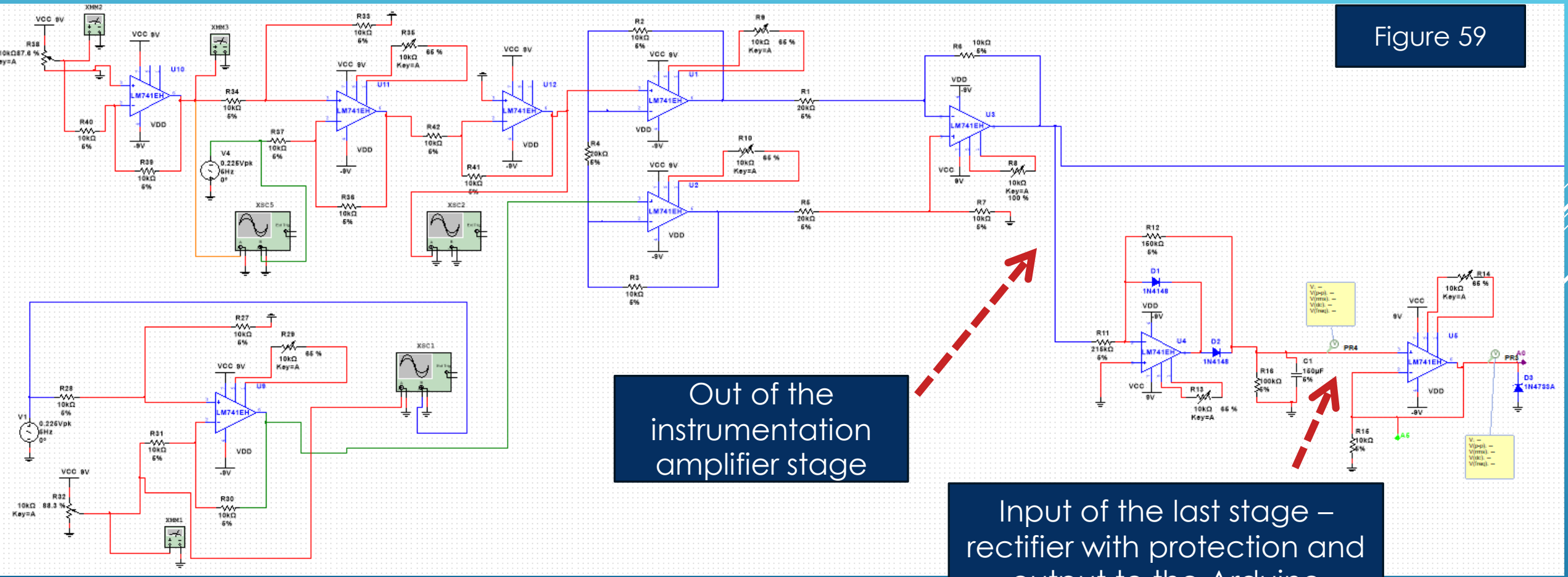
sixth block-amplify the signal by sqrt(2) in order to get the RMS value

Resistors are equal (10k ohm)

seventh block – amplify the signal by 2 – compensate on the cutting for the entire range

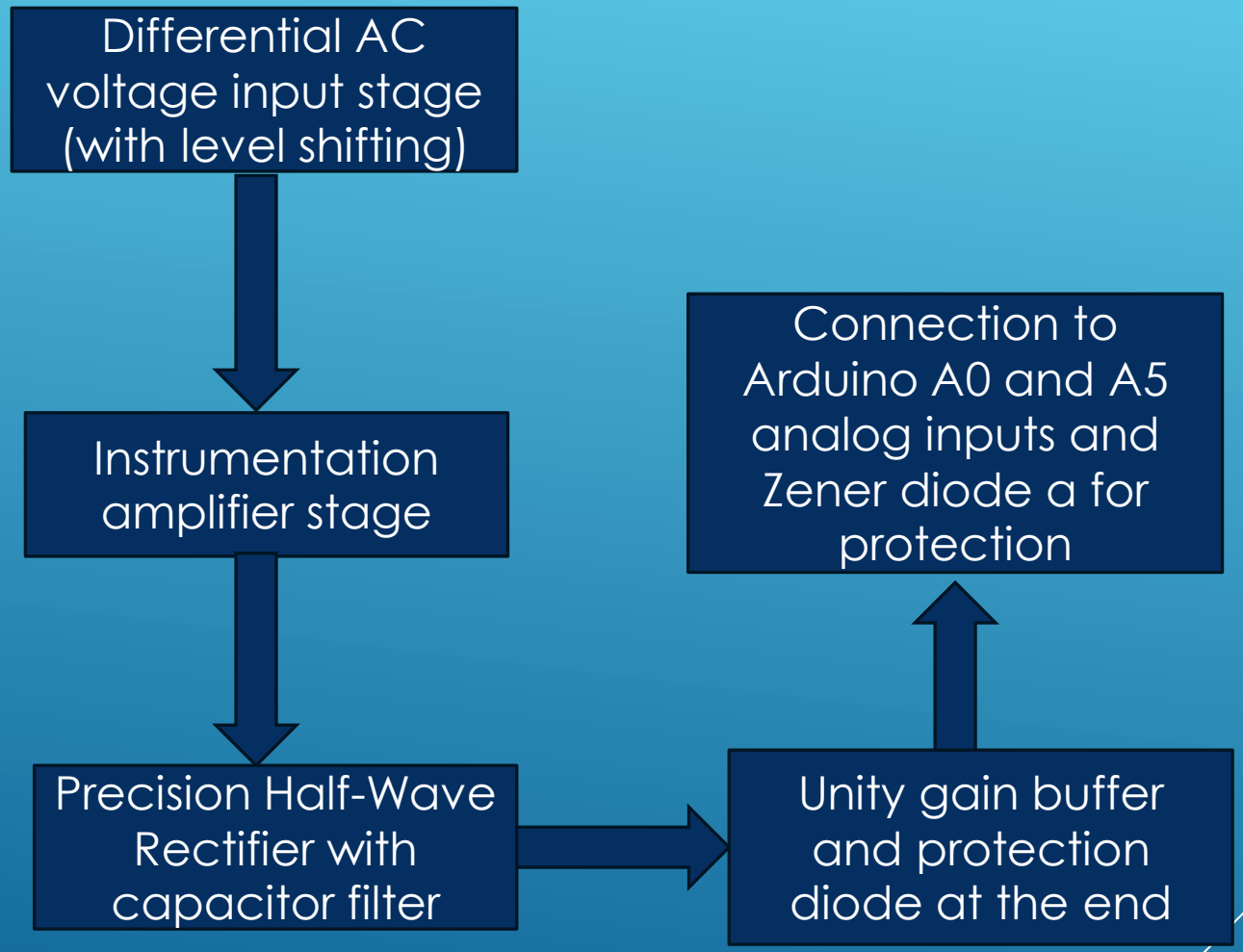
P22 - DRAW THE FULL CIRCUIT OF THE DACDCA – PART 1

"advanced approach" – using the Arduino to evaluate the Differential AC voltage



P22 - DRAW THE FULL CIRCUIT OF THE DACDCA – PART 2

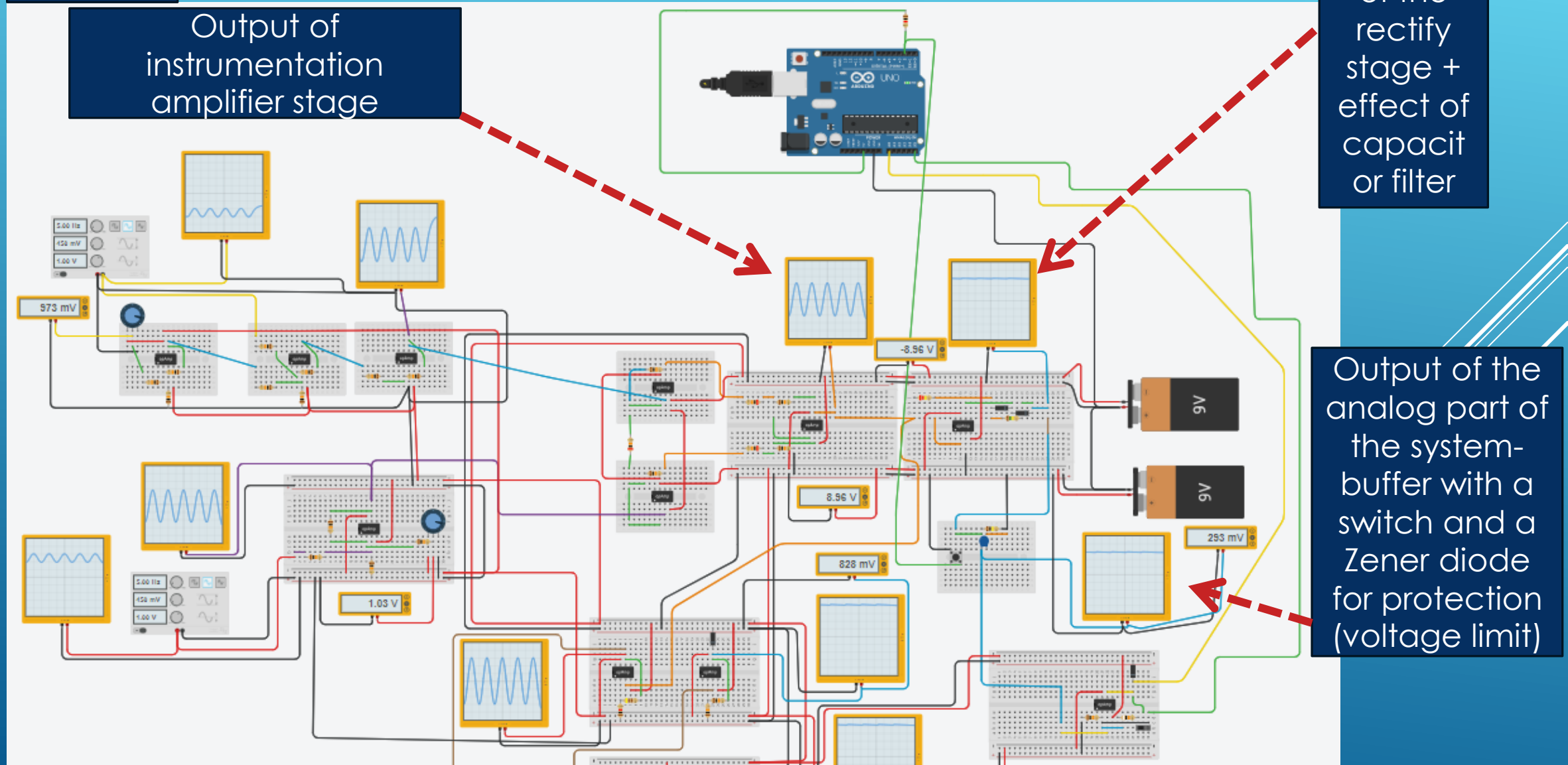
"advanced approach" – using the Arduino to evaluate the Differential AC voltage



P22 - DRAW THE FULL CIRCUIT OF THE DACDCA – PART 3

"advanced approach" – using the Arduino to evaluate the Differential AC voltage – TinkerCad simulation

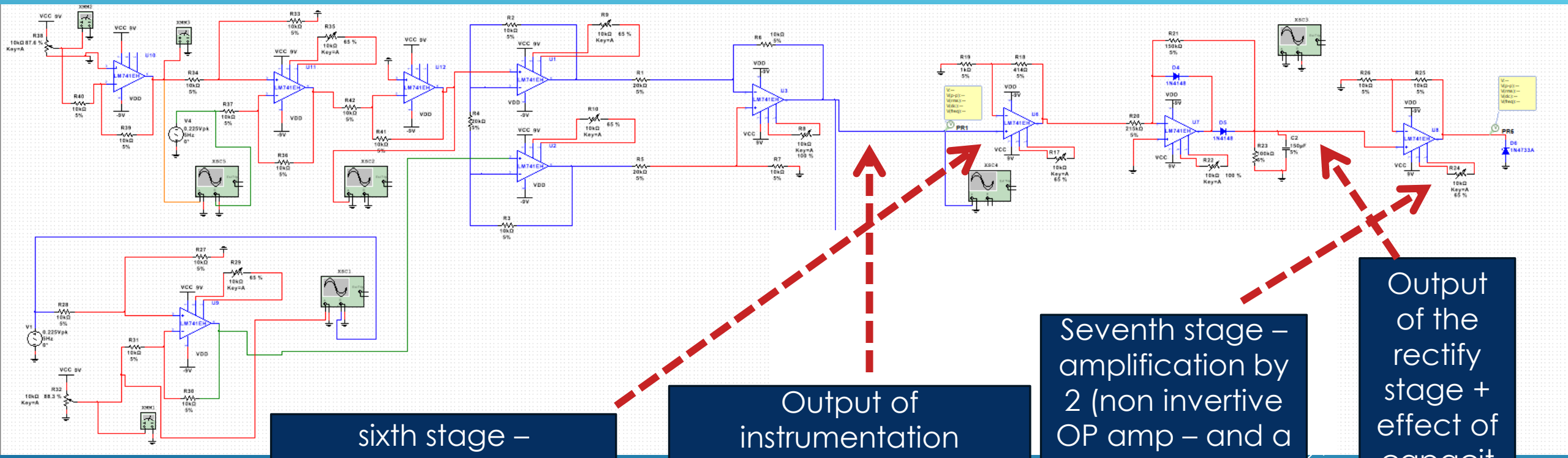
Figure 60



P22 - DRAW THE FULL CIRCUIT OF THE DACDCA – PART 4

"classical approach" – without using the Arduino to evaluate the Differential AC voltage

Figure 61



sixth stage – amplification by sqrt(2) (non invertive OP amp

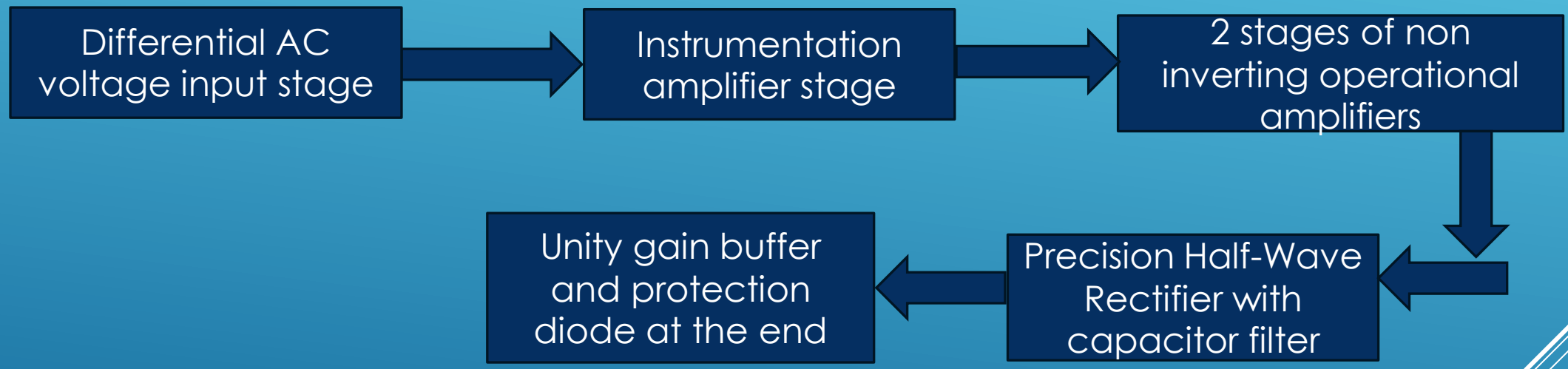
Output of instrumentation amplifier stage

Seventh stage – amplification by 2 (non invertive OP amp – and a protection Zener diode

Output of the rectify stage + effect of capacit or filter

P22 - DRAW THE FULL CIRCUIT OF THE DACDCA – PART 3

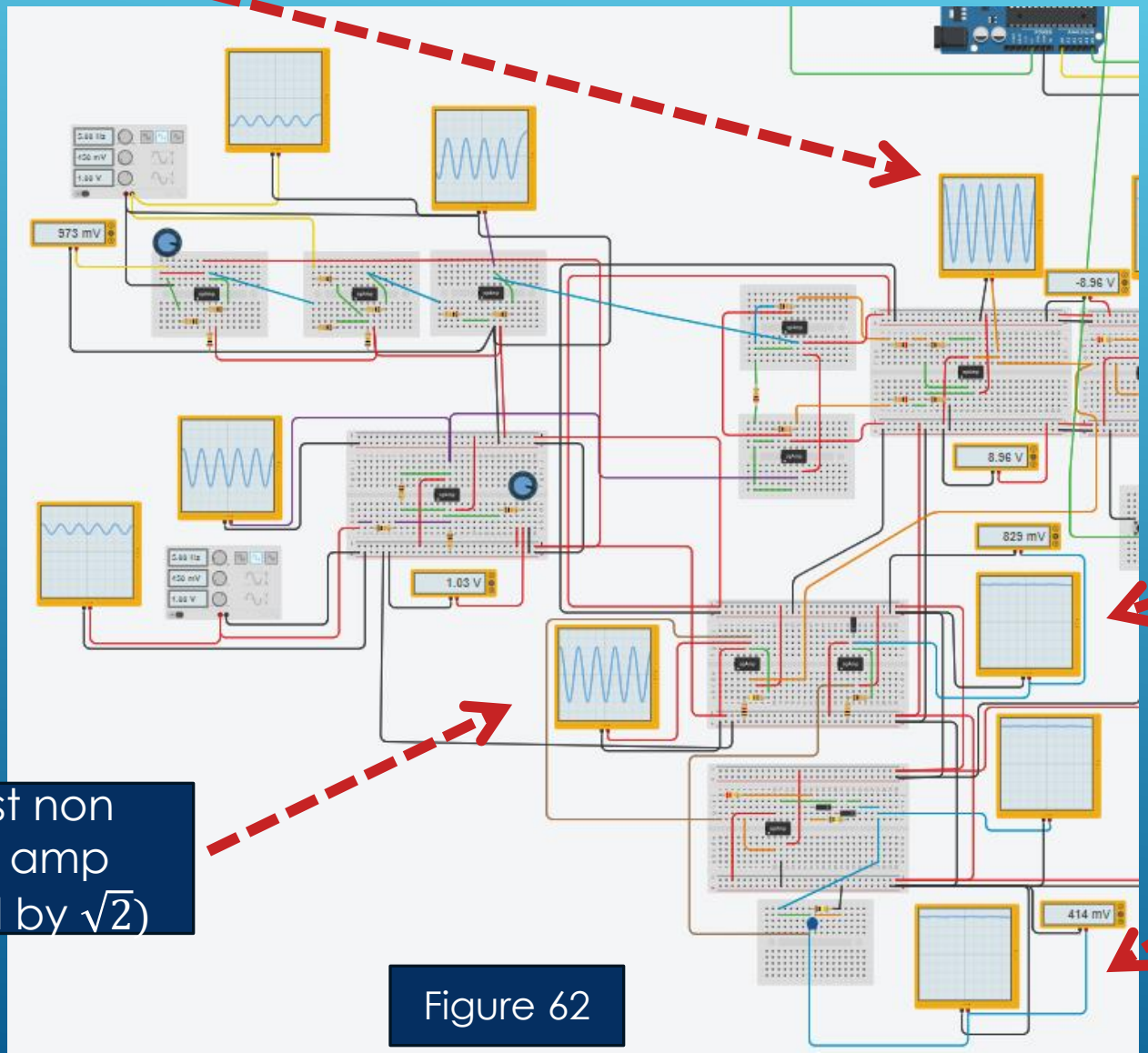
"classical approach" – without using the Arduino to evaluate the Differential AC voltage



P22 - DRAW THE FULL CIRCUIT OF THE DACDCA – PART 4

Output of instrumentation amplifier stage

"classical approach" – without using the Arduino to evaluate the Differential AC voltage – TinkerCad simulation



TinkerCad simulation gives less accurate results than in Multisim

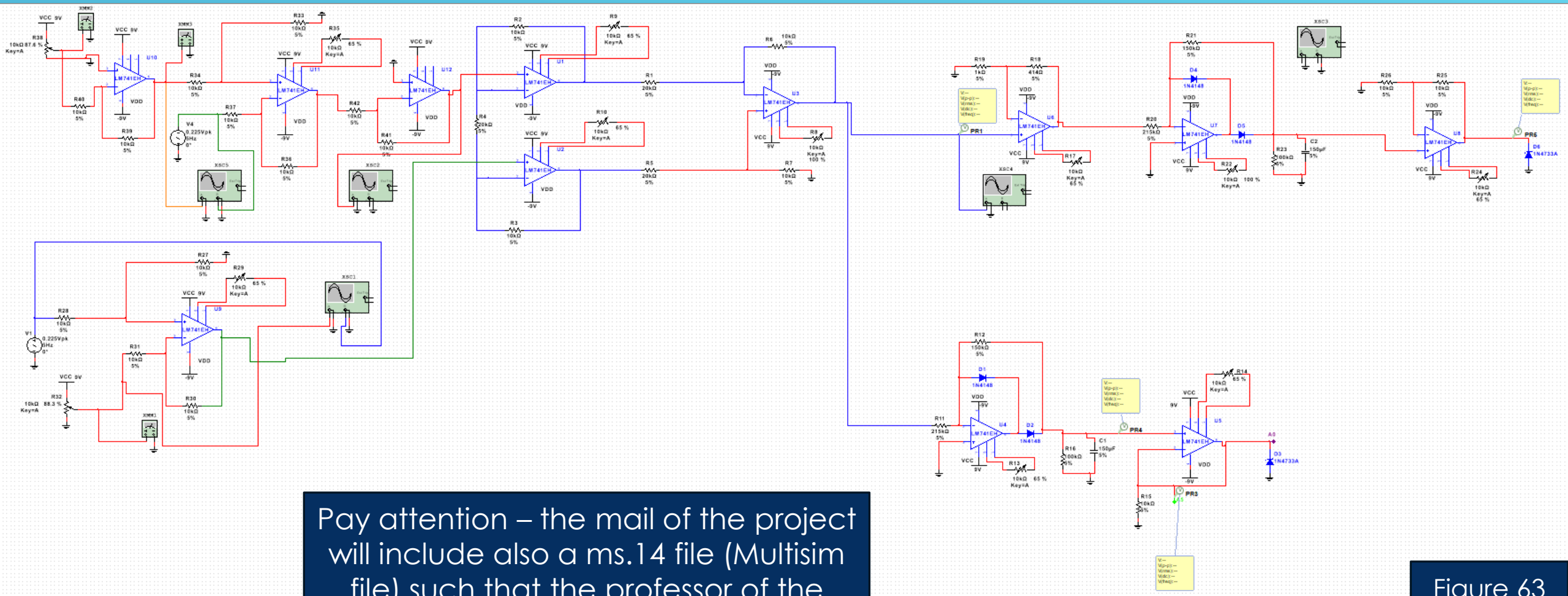
Output of second non inverting OP amp (amplify signal by 2) – final result with classical approach (without Arduino)

Output of the rectify stage + effect of capacitor filter

Output of first non inverting OP amp (amplify signal by $\sqrt{2}$)

Figure 62

P22 - DRAW THE FULL CIRCUIT OF THE DACDCA – PART 5



Pay attention – the mail of the project will include also a ms.14 file (Multisim file) such that the professor of the course will be able to simulate the circuit in full and big screen

Figure 63

P22 - DRAW THE FULL CIRCUIT OF THE DACDCA – PART 6

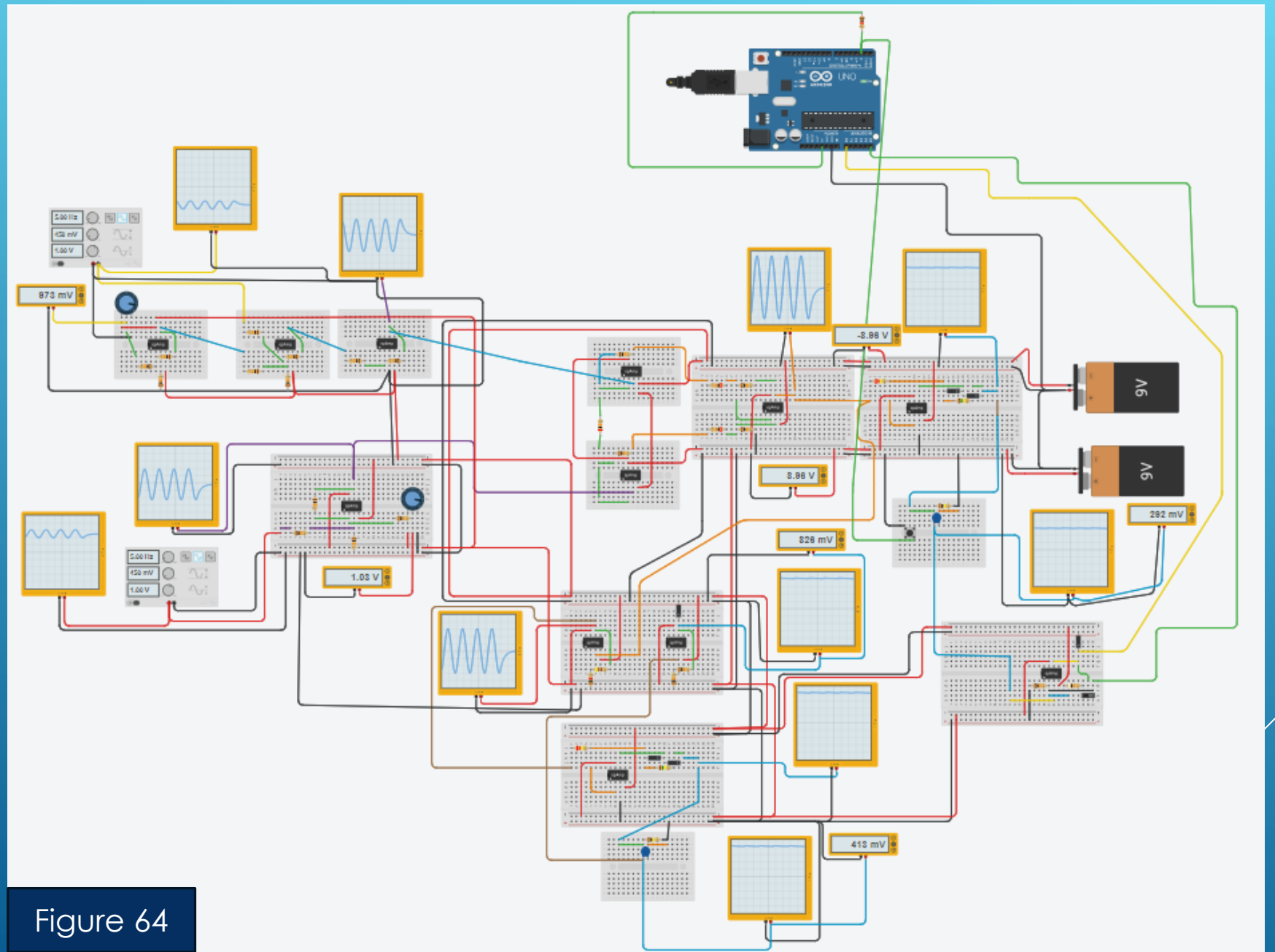


Figure 64

P24-PEXPLAIN HOW MAXIMAL POSSIBLE CMRR WAS ACHIEVED – PART 1

The following Figure presents the offset error of the non inverting operational amplifier that we tried to fix using the Potentiometer-one can see that: $V_E = 31.103\ mV$

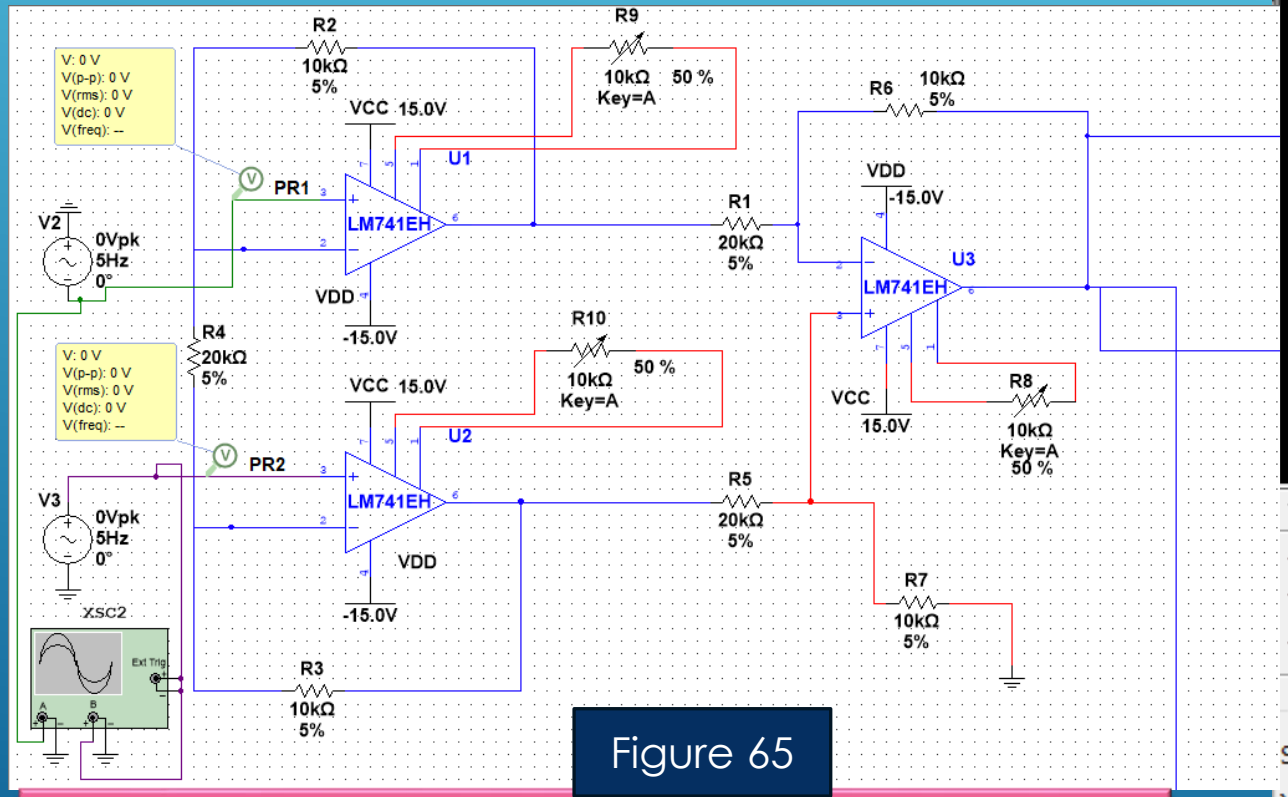


Figure 65

We provide zero input in order to find that error and take it into consideration

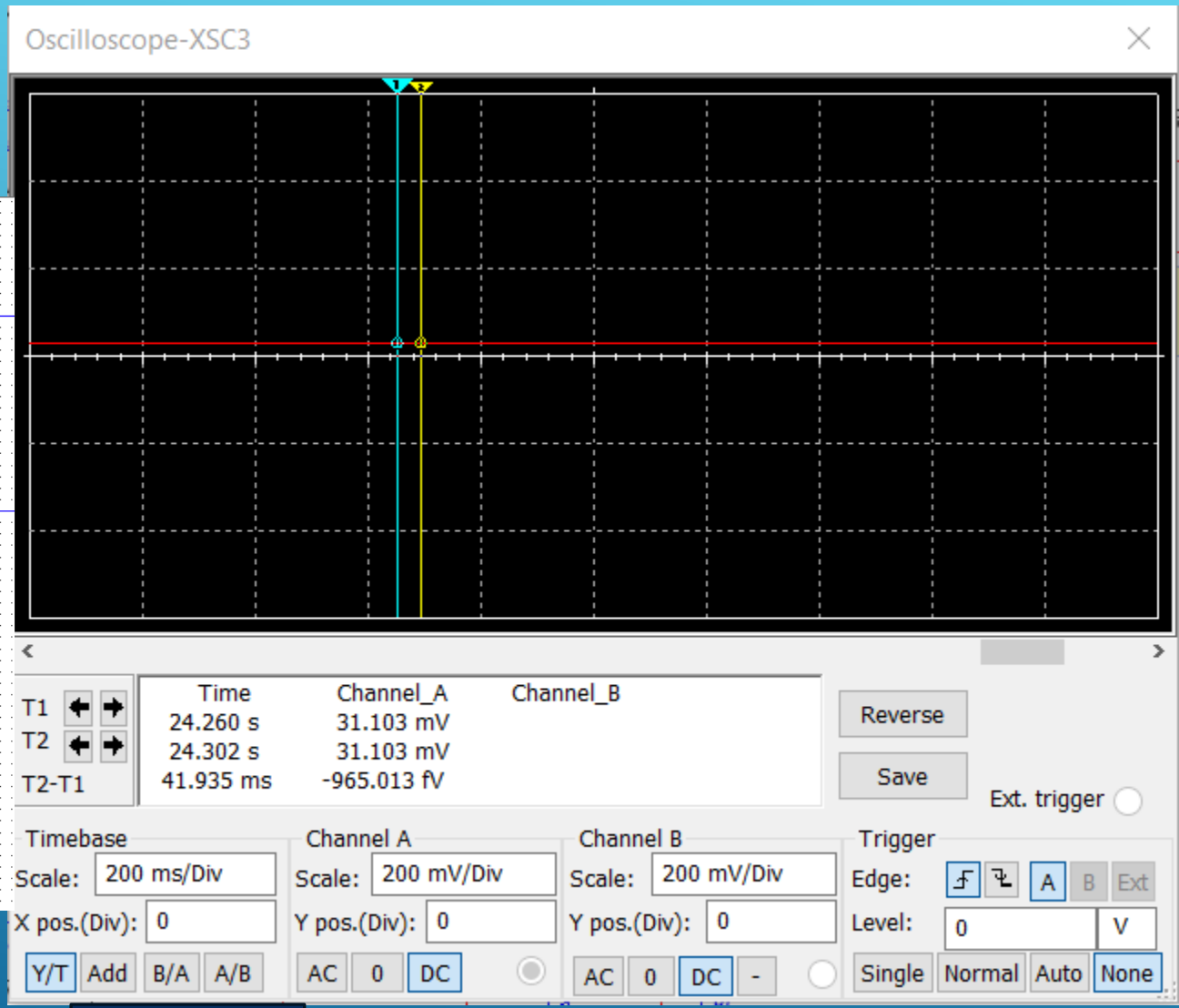


Figure 66

P24-PEXPLAIN HOW MAXIMAL POSSIBLE CMRR WAS ACHIEVED – PART 2

As can be seen this is the common mode circuit in order to calculate the CMRR-we connected DC sources in the same polarity to the inputs of the system.

the common mode rejection ratio (CMRR) of a differential amplifier (or other device) is a metric used to quantify the ability of the device to reject common-mode signals, i.e. those that appear simultaneously and in-phase on both inputs. An ideal differential amplifier would have infinite CMRR, however this is not achievable in practice.

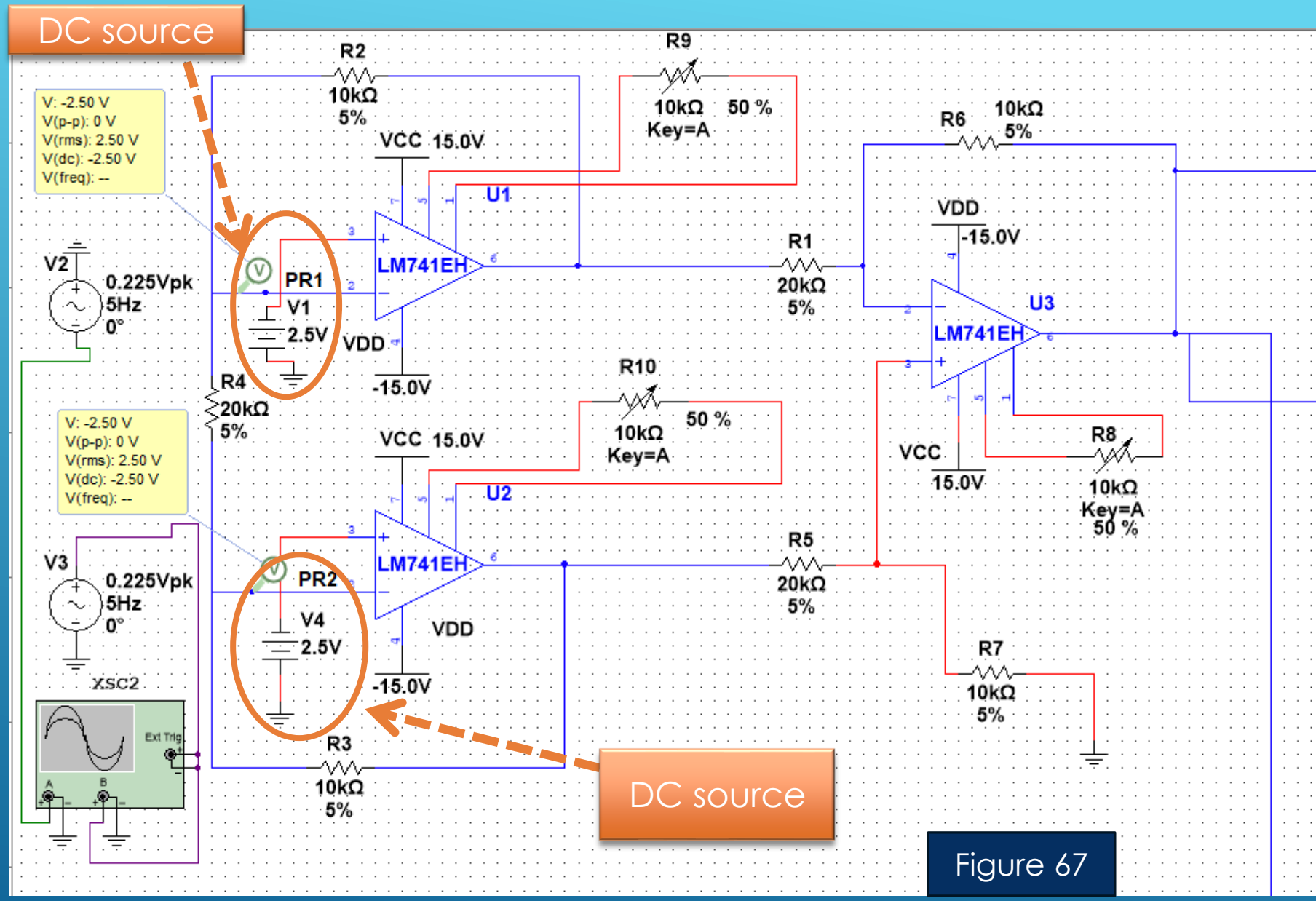


Figure 67

https://en.wikipedia.org/wiki/Common-mode_rejection_ratio

P24-PEXPLAIN HOW MAXIMAL POSSIBLE CMRR WAS ACHIEVED – PART 3

We will calculate the CMRR according to our recent results:

1) Providing zero voltage (CM) in the input will result 31.103 mV at the output.

2) Providing 2.5V voltage (CM) in the input will result 31.2 mV at the output.

3) 31.2 mV - 31.103 mV =0.097 mV =A_{CM}

The CMRR is:

$$CMRR = 20\log\left(\frac{A_d}{A_{CM}}\right)$$

FOR A_d=1:

$$20\log\left(\frac{1}{0.097\ mV}\right) = 80.26$$

FOR A_d=10:

$$20\log\left(\frac{10}{0.097\ mV}\right) = 100.264$$

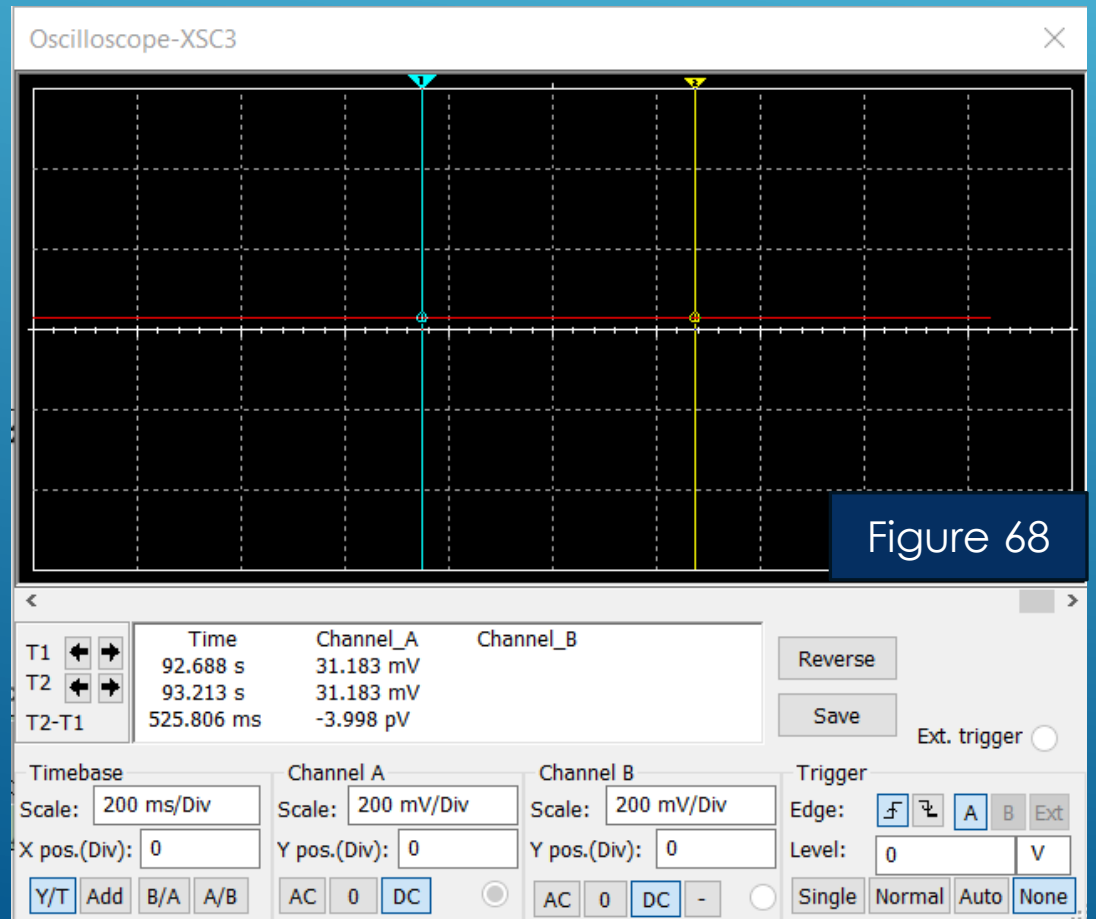
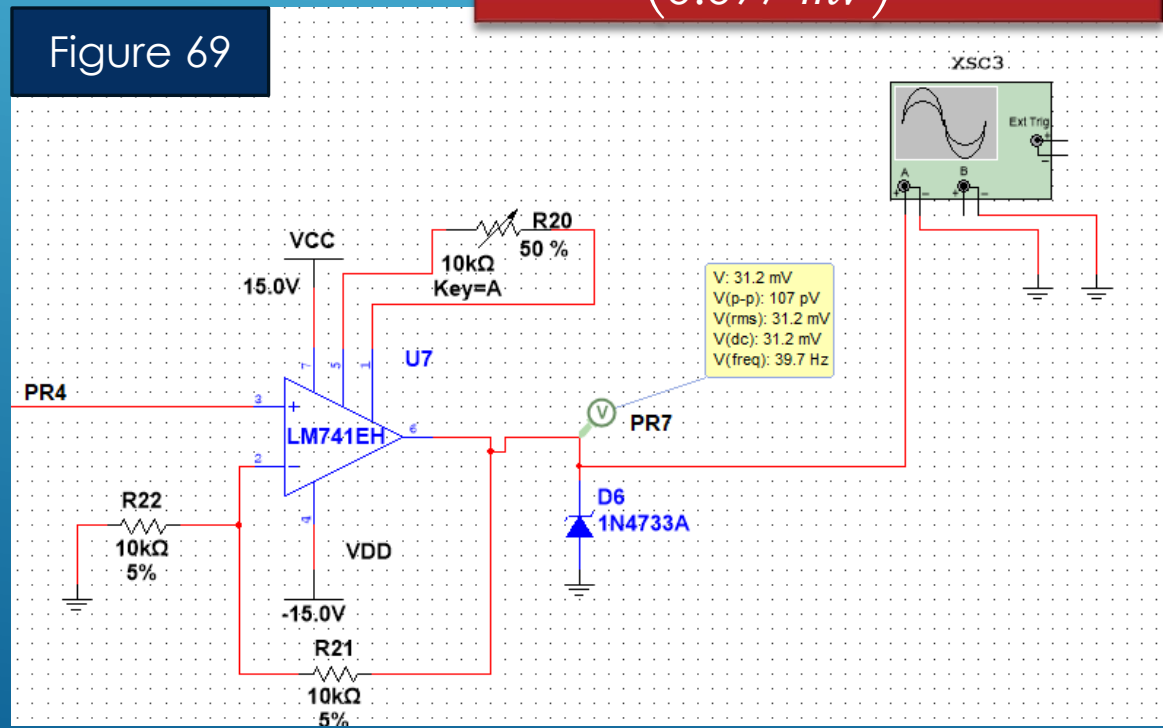


Figure 68

Figure 69



P25-FULL CODE OF THE SKETCH WITH PROPER COMMENTS AND EXPLANATIONS–

PART 1

```
1  float value_amplifier_output = A0;
2  float multy = A5;
3  float val = 0;
4  float acquired = 0;
5  float sum = 0;
6  float mean = 0;
7  float tmp = 0;
8  int flag = 0;
9  float variance = 0;
10 float standard_deviation = 0;
11 int iterations = 1;
```

Figure 70

value_amplifier_output and multy:
These are float variables set to A0 and A5 respectively. These represent analog input pins on an Arduino board.

val, acquired, sum, mean, tmp, variance, standard_deviation:
These are float variables initialized to 0, used for calculating statistical data such as mean and standard deviation from voltage readings.

1) val is used to store the current sensor reading.
2) acquired is used to store a processed value of the reading.
3) sum accumulates the acquired values for averaging.
4) mean calculates the average of acquired values.
5) tmp is used as a helper for variance calculation.
6) variance and standard_deviation are used to calculate the spread of the sensor readings around the mean.

Flag : This is an int variable initialized to 0. It is used as a condition indicator to trigger certain parts of the code based on the comparison between values from value_amplifier_output and multy.

Iterations : An int variable set to 1. This is used to keep track of the number of readings or operations performed since the start or reset.

P25-FULL CODE OF THE SKETCH WITH PROPER COMMENTS AND EXPLANATIONS–

PART 3

```
12  const int button_pin=2;
13  int press_button=0;
14  int counter=0;
15  float mean_save=0;
16  float sd_save=0;
```

Figure 71

button_pin, press_button, counter:
1) button_pin is a constant integer set to 2, representing the digital pin where a button is connected.
2) press_button is an integer initialized to 0, which gets set to 1 when the button is pressed, enabling certain actions in an interrupt service routine.
3) counter is used to count specific events or iterations within a loop or ISR.

mean_save, sd_save:
These float variables initialized to 0 are used to store the final calculated mean and standard deviation values before resetting the system or to display them after computations are complete.

P25-FULL CODE OF THE SKETCH WITH PROPER COMMENTS AND EXPLANATIONS–

PART 4

```
17  void setup() {
18    Serial.begin(9600); // Initialize serial communication at 9600 bps
19    pinMode(value_amplifier_output, INPUT);
20    pinMode(multy, INPUT);
21    pinMode(button_pin, INPUT_PULLUP);  //set Button pin as pullup input
```

Figure 72

pinMode(value_amplifier_output, INPUT); → configures the pin assigned to value_amplifier_output (an analog pin) as an input to read data from an external sensor or device.

pinMode(multy, INPUT); → sets another pin, multy, as input for similar purposes as value_amplifier_output.

pinMode(button_pin, INPUT_PULLUP); → sets the button_pin as an input with an internal pull-up resistor. This is useful for button inputs where the default state is high (logic level HIGH), and pressing the button pulls the pin to ground (logic level LOW).

Serial.begin(9600); sets up the serial communication at 9600 bits per second (bps). This is used for data logging, debugging, and communication between the Arduino and a computer or other serial devices.

P25-FULL CODE OF THE SKETCH WITH PROPER COMMENTS AND EXPLANATIONS–

PART 5

```
22    //set Button as External Interrupt with change_mode function
23    attachInterrupt(digitalPinToInterrupt(button_pin), myISR,FALLING);
```

attachInterrupt(digitalPinToInterrupt(button_pin), myISR, FALLING); attaches an interrupt to button_pin, triggering the myISR function when the pin transitions from high to low (falling edge). This is used for handling button presses.

```
25    // TIMER 1 for interrupt frequency 100 Hz:
26    cli(); // stop interrupts
27    TCCR1A = 0; // set entire TCCR1A register to 0
28    TCCR1B = 0; // same for TCCR1B
29    TCNT1 = 0; // initialize counter value to 0
30    OCR1A = 19999; // = 16000000 / (8 * 100) - 1 (must be <65536)
31    TCCR1B |= (1 << WGM12); // turn on CTC mode
32    TCCR1B |= (0 << CS12) | (1 << CS11) | (0 << CS10); // Set CS12, CS11 and CS10 bits for 8 prescaler
33    TIMSK1 |= (1 << OCIE1A); // enable timer compare interrupt
34    sei(); // allow interrupts
35  }
```

The code sets up Timer 1 to generate interrupts at a frequency of 100 Hz (every 10 milliseconds), which can be used for periodic tasks like sensor readings or time-sensitive calculations.

Figure 72

Figure 73

```
37  void myISR() {
38    Serial.println("Acquisition and analysis are going on now\n");
39    press_button=1;
40   }
```

The myISR() function is the interrupt service routine for the button press and sets the press_button variable to 1.

P25-FULL CODE OF THE SKETCH WITH PROPER COMMENTS AND EXPLANATIONS–

PART 3

```
ISR(TIMER1_COMPA_vect) {
  if(press_button==1)
  {
    if(counter<5)
    {
        val = analogRead(multy);
        if (analogRead(multy) >= analogRead(value_amplifier_output)) {
        flag = 0;
        acquired = analogRead(value_amplifier_output);
        acquired = map(acquired, 0, 1023, 0, 15000);
        } else {
            flag = 1;
            acquired = analogRead(value_amplifier_output);
            acquired = map(acquired, 0, 1023, 0, 1500);
        }
        sum += acquired;
        mean = sum / iterations;
        mean_save=mean;
        tmp += (acquired - mean) * (acquired - mean);
        variance = tmp / iterations;
        standard_deviation = sqrt(variance);
        sd_save=standard_deviation;
        iterations++;

        if (iterations % 100 == 0) {
            printToMonitor();
            counter++;
        }

        if (iterations >= 500) {
            clearVariables();
        }
    }
```

Figure 74

```
    else
    {
      press_button=0;
      counter=0;
      Serial.println("Acquisition and analysis finished\n");
      Serial.print("Mean [mV]: ");
      Serial.print(mean_save);
      Serial.print(", Standard Deviation [mV]: ");
      Serial.println(sd_save);
    }
  }
}
```

The **ISR(TIMER1_COMPA_vect)** function is the interrupt service routine for the timer interrupt. It checks if the **press_button** variable is 1 and if the **counter** is less than 5.
If both conditions are true, it takes a measurement of the value amplifier output and the multiplier, maps the value to a range of 0 to 15000 or 0 to 1500 depending on the comparison of the two inputs. It then calculates the sum of the measurements, the mean, the variance, and the standard deviation. The mean and standard deviation are printed to the serial monitor every 100 measurements and the variables are cleared when 500 measurements have been taken.

P25-FULL CODE OF THE SKETCH WITH PROPER COMMENTS AND EXPLANATIONS–

PART 4

```
89  void printToMonitor() {
90    Serial.print("Mean [mV]: ");
91    Serial.print(mean);
92    Serial.print(", Standard Deviation [mV]: ");
93    Serial.println(standard_deviation);
94  }
95
96  void clearVariables() {
97    val = 0;
98    acquired = 0;
99    sum = 0;
100   mean = 0;
101   tmp = 0;
102   variance = 0;
103   standard_deviation = 0;
104   iterations = 1;
105 }
106
107 void loop() {
108   // Empty loop since all the work is done in the ISR
109 }
```

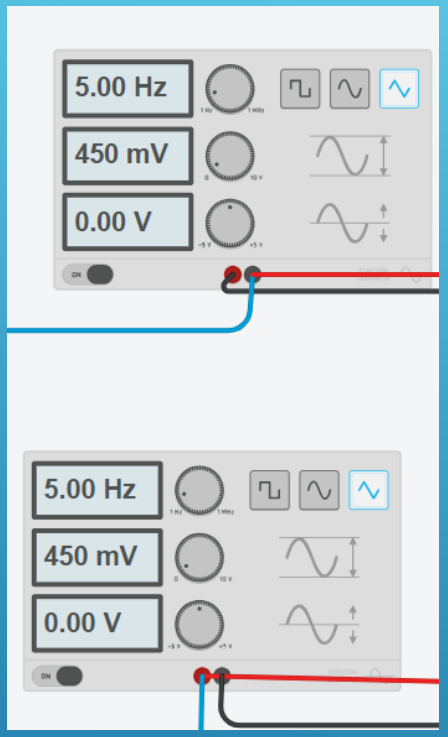
Figure 75

The **printToMonitor()** function prints the mean and standard deviation to the serial monitor.

The **clearVariables()** function resets all the variables used in the calculation to their initial values.
The **loop()** function is empty since all the work is done in the interrupt service routines.

P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 1.1

Signal without DC offset (level shifter isn't in the circuit yet)



Serial Monitor

```
Acquisition and analysis are going on now

Mean [mV]: 903.85, Standard Deviation [mV]: 6.98
Mean [mV]: 903.80, Standard Deviation [mV]: 7.05
Mean [mV]: 903.78, Standard Deviation [mV]: 7.08
Mean [mV]: 903.77, Standard Deviation [mV]: 7.10
Mean [mV]: 903.77, Standard Deviation [mV]: 7.11
Acquisition and analysis finished

Mean [mV]: 903.77, Standard Deviation [mV]: 7.11
```

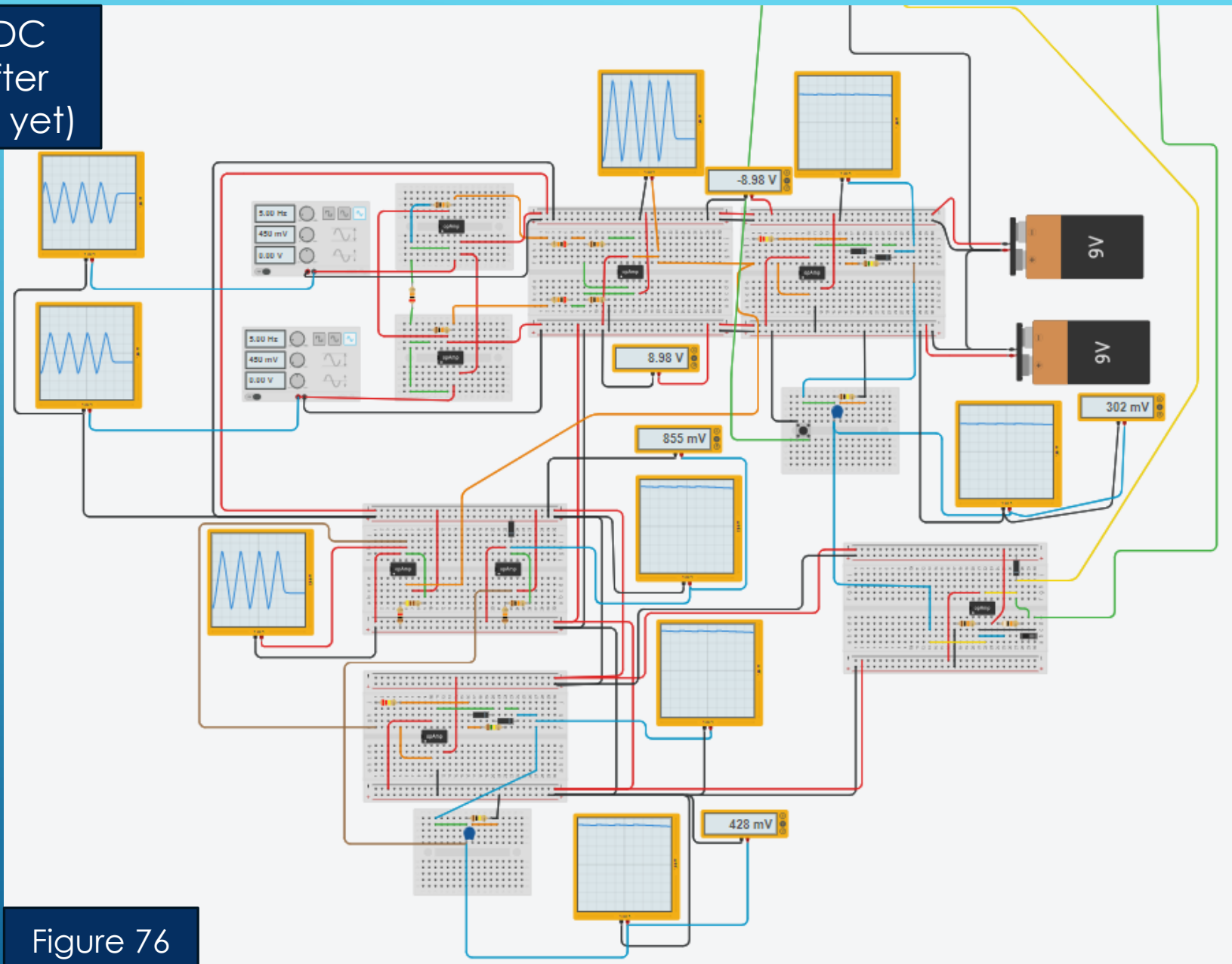
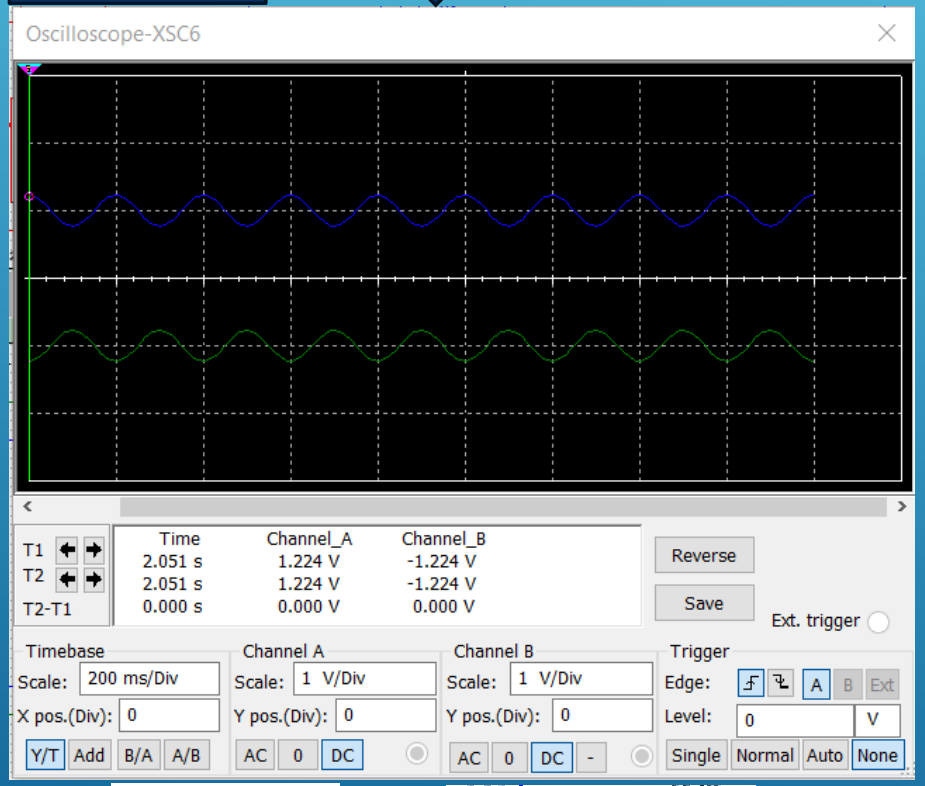


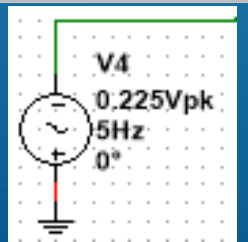
Figure 76

P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 2.1

Input signals : {-450 m[V] , 450m[V]}
sinusoidal signal with DC offset of 1V

Figure 79





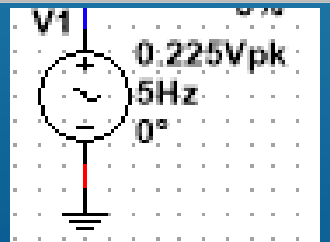
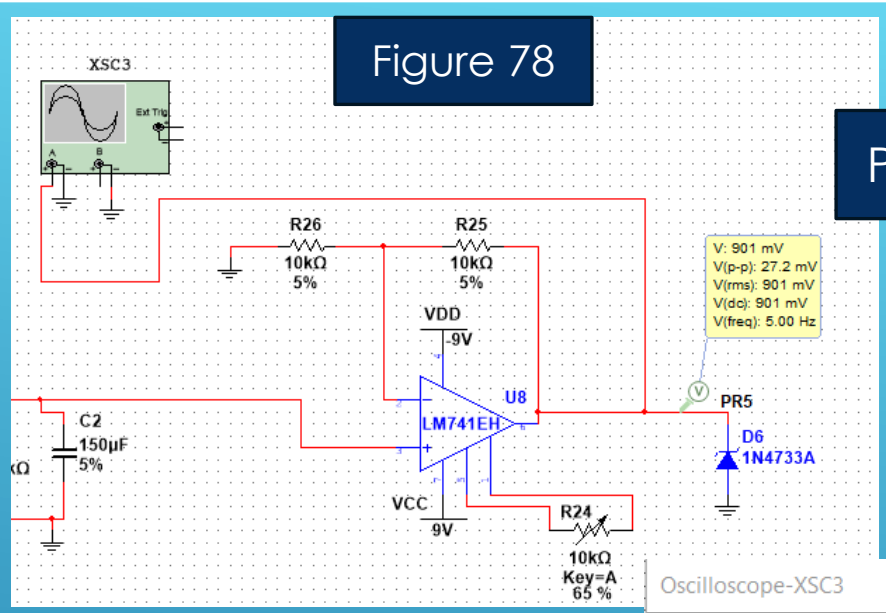


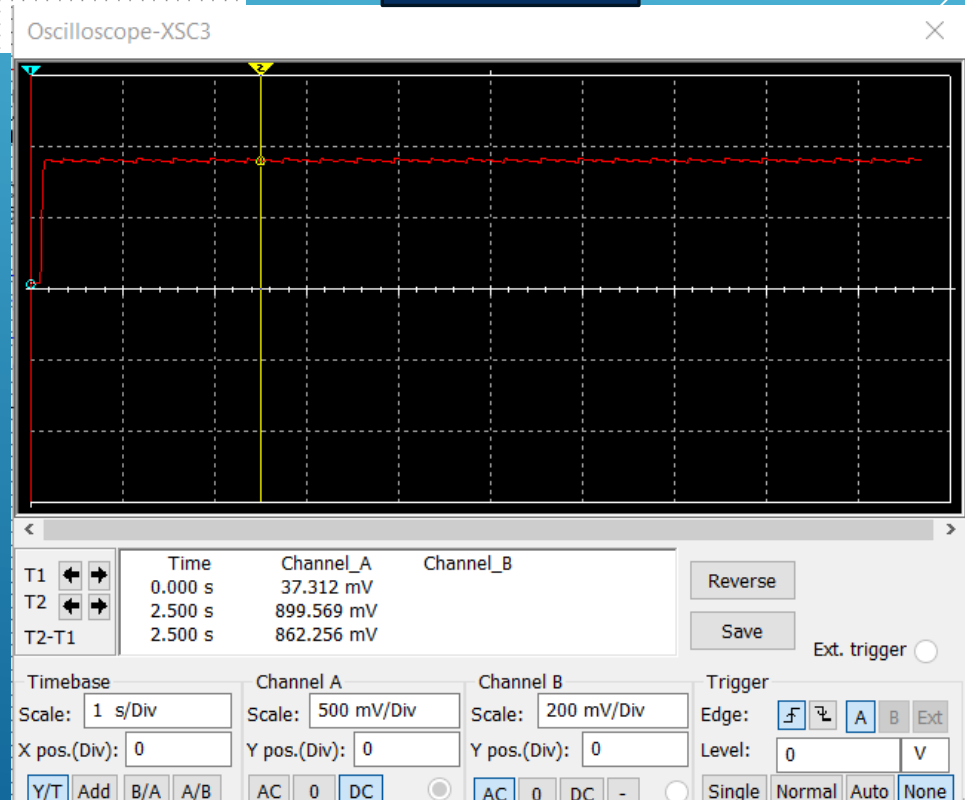
Figure 78



Part 2.1 is a Multisim simulation

Part 2.2 is a TinkerCad simulation

Figure 77



P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 2.2

Figure 79

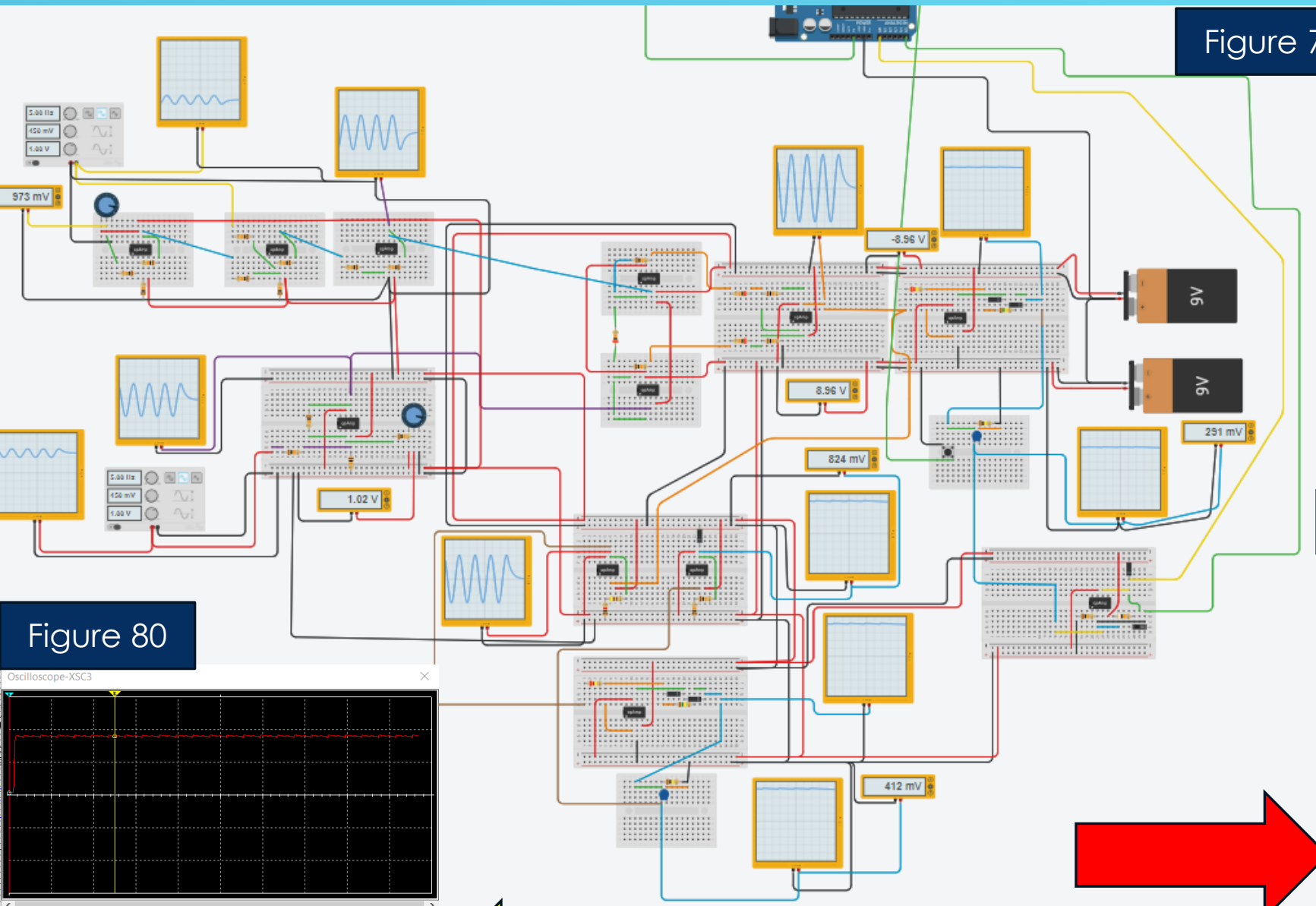


Figure 80

Input signals : {-450 m[V] , 450m[V]}
sinusoidal signal with DC offset of 1V

Figure 81

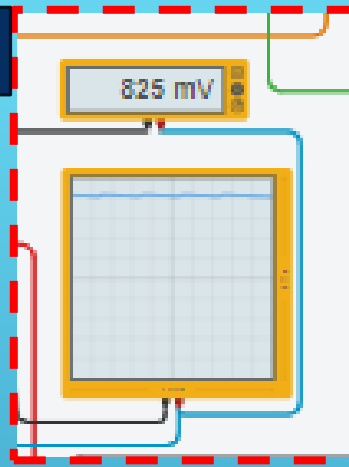


Figure 82

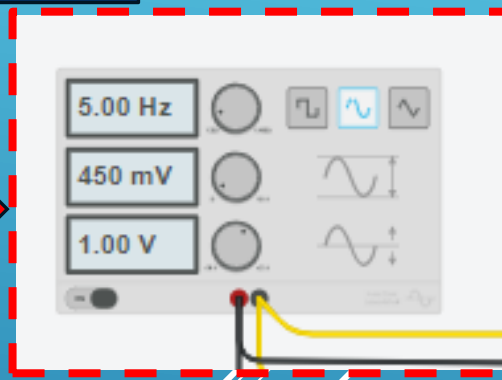
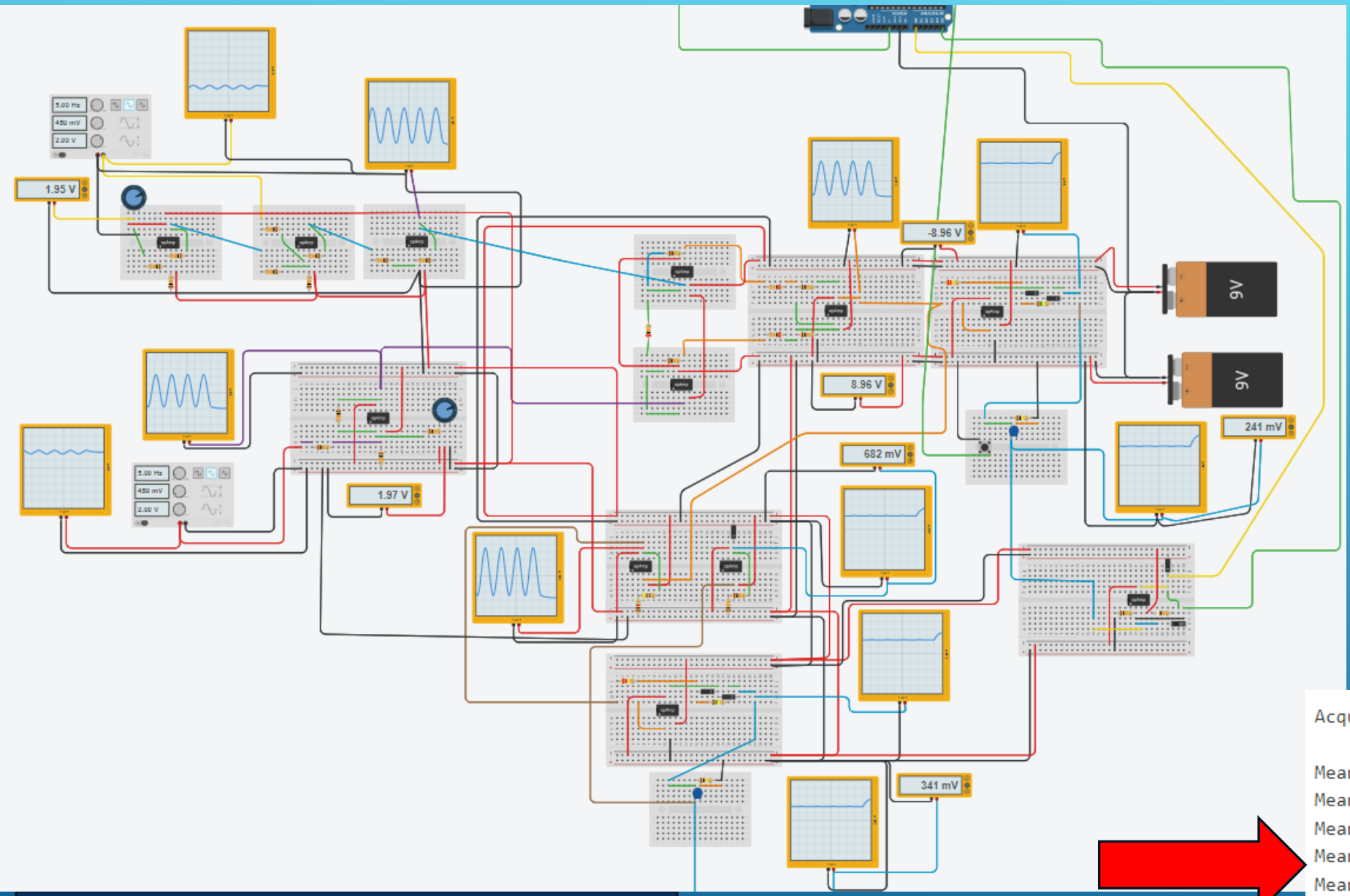
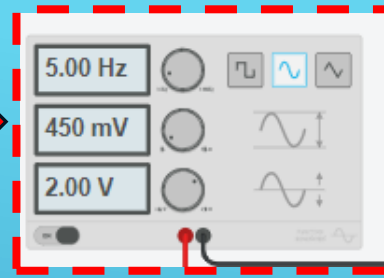


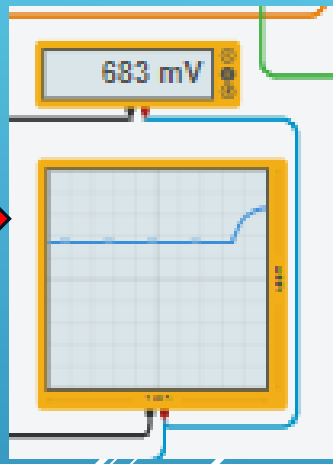
Figure 83

```
Acquisition and analysis are going on now

Mean [mV]: 877.07, Standard Deviation [mV]: 6.52
Mean [mV]: 877.01, Standard Deviation [mV]: 6.65
Mean [mV]: 876.99, Standard Deviation [mV]: 6.70
Mean [mV]: 876.98, Standard Deviation [mV]: 6.72
Mean [mV]: 876.97, Standard Deviation [mV]: 6.74
Acquisition and analysis finished

Mean [mV]: 876.97, Standard Deviation [mV]: 6.74
```

P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 3







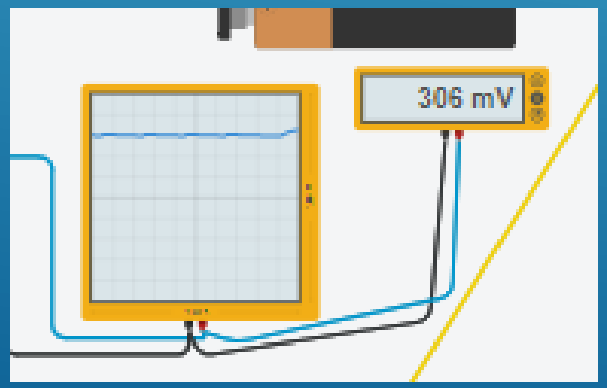

```
Acquisition and analysis are going on now

Mean [mV]: 727.70, Standard Deviation [mV]: 7.04
Mean [mV]: 727.72, Standard Deviation [mV]: 7.09
Mean [mV]: 727.73, Standard Deviation [mV]: 7.10
Mean [mV]: 727.74, Standard Deviation [mV]: 7.11
Mean [mV]: 727.74, Standard Deviation [mV]: 7.12
Acquisition and analysis finished

Mean [mV]: 727.74, Standard Deviation [mV]: 7.12
```

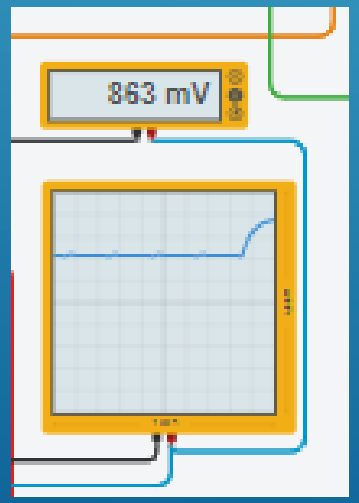
Input signals : {-450 m[V] , 450m[V]}
sinusoidal signal with DC offset of 2V

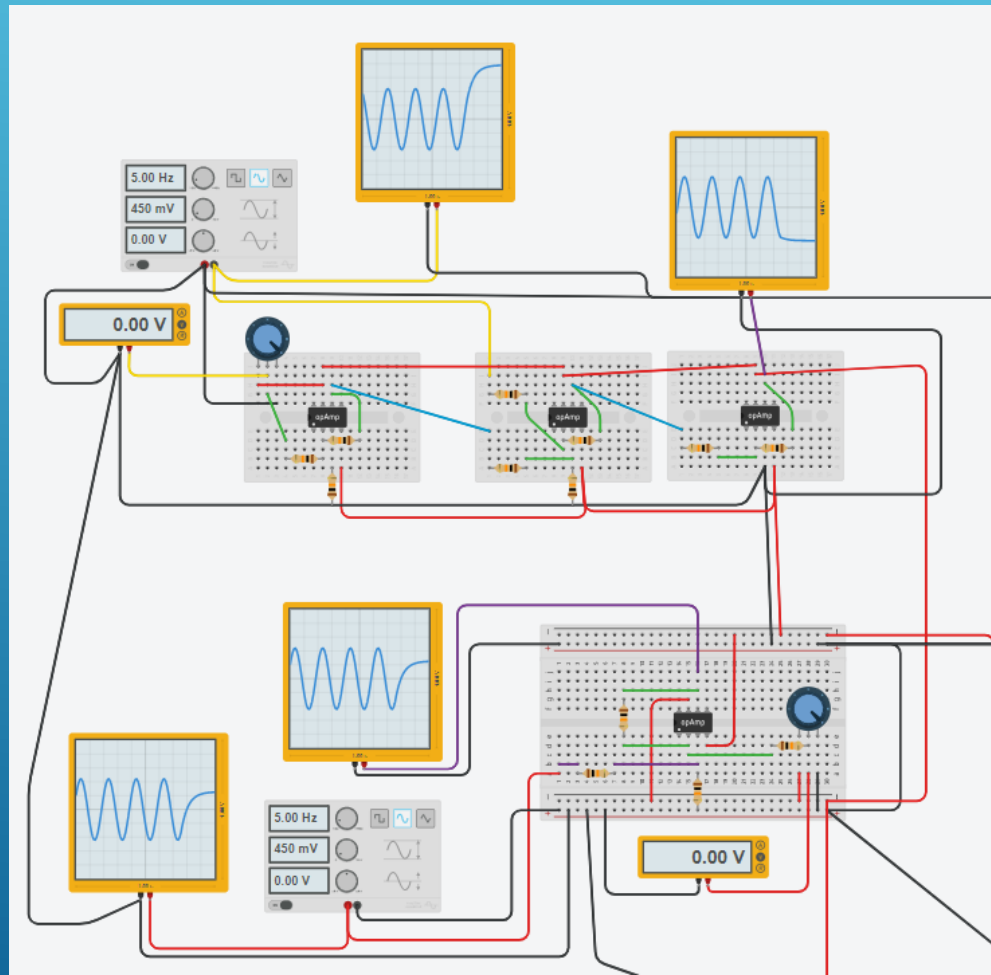
P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 4

```
Acquisition and analysis are going on now

Mean [mV]: 914.66, Standard Deviation [mV]: 6.77
Mean [mV]: 914.63, Standard Deviation [mV]: 6.80
Mean [mV]: 914.62, Standard Deviation [mV]: 6.82
Mean [mV]: 914.61, Standard Deviation [mV]: 6.82
Mean [mV]: 914.61, Standard Deviation [mV]: 6.83
Acquisition and analysis finished

Mean [mV]: 914.61, Standard Deviation [mV]: 6.83
```

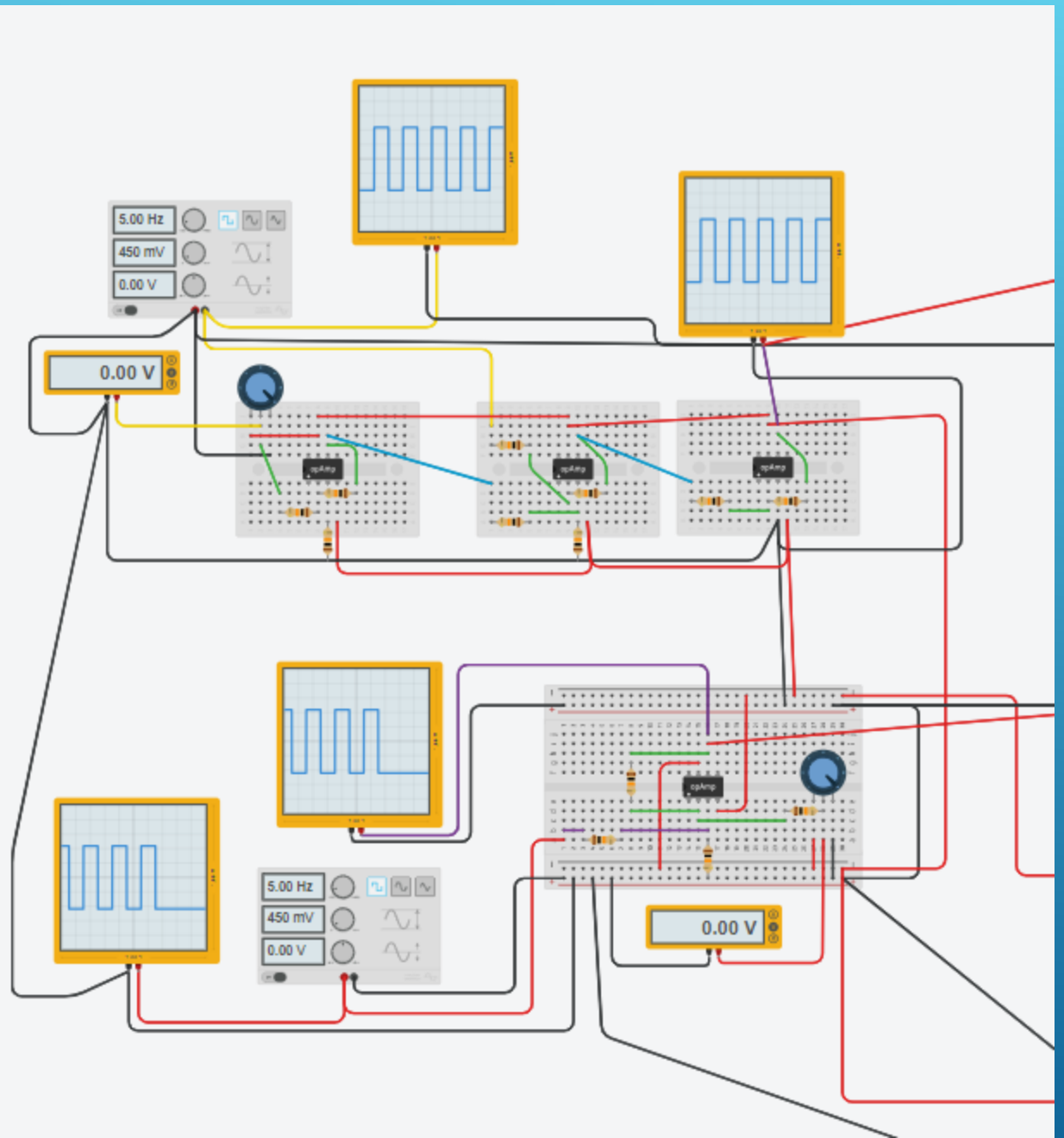
Input signals : {-450 m[V] , 450m[V]}
sinusoidal signal with DC offset of 0V







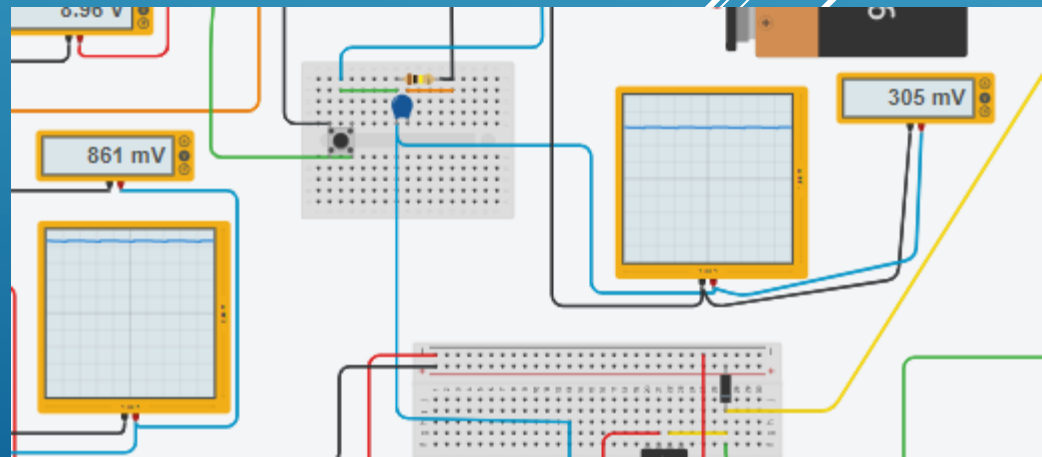
P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 5



Input signals : {-450 m[V] , 450m[V]}
rectangular signal with DC offset of 0V

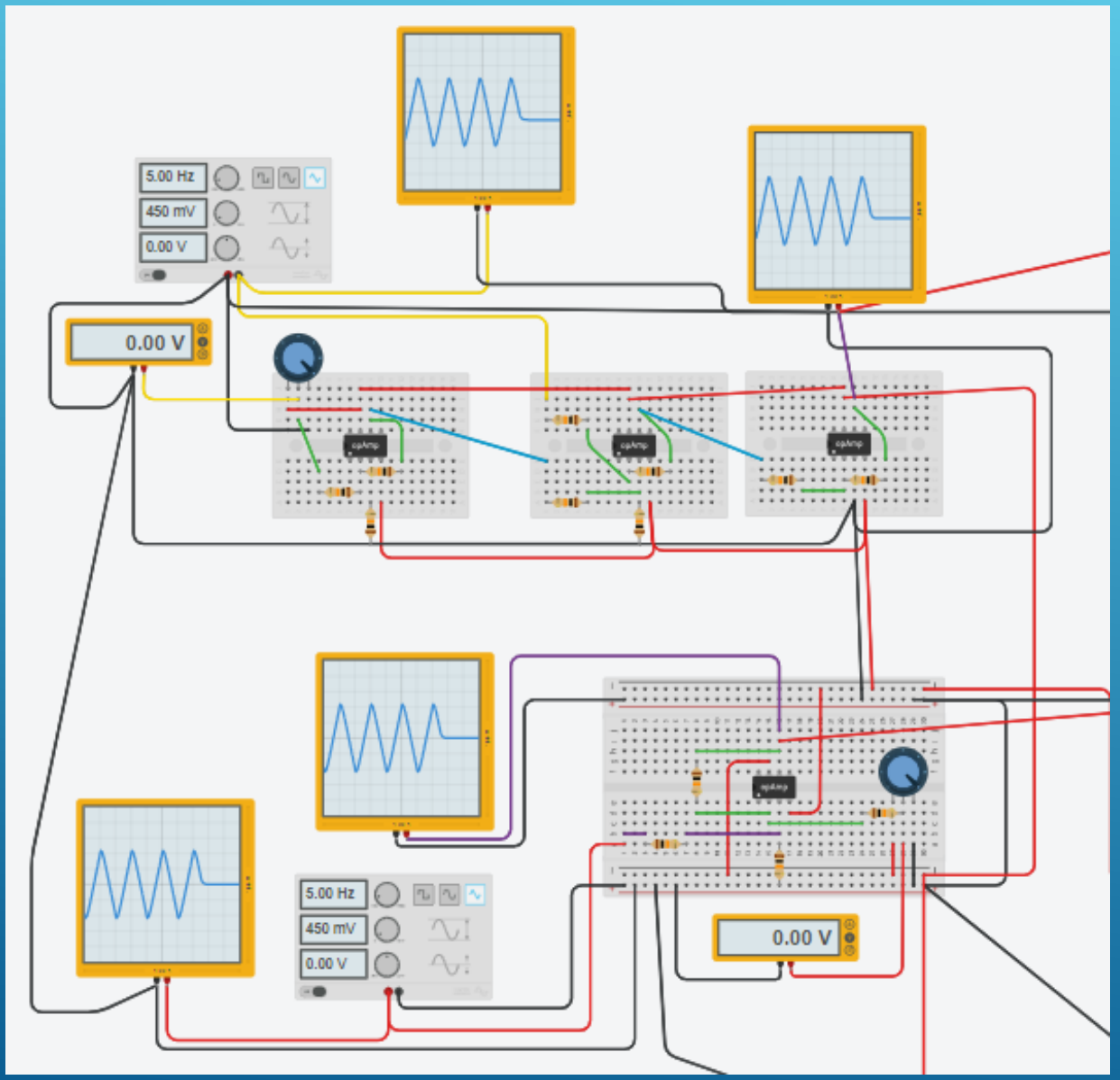
Serial Monitor

```
Acquisition and analysis are going on now

Mean [mV]: 918.76, Standard Deviation [mV]: 6.34
Mean [mV]: 918.78, Standard Deviation [mV]: 6.37
Mean [mV]: 918.79, Standard Deviation [mV]: 6.38
Mean [mV]: 916.71, Standard Deviation [mV]: 41.78
Mean [mV]: 915.40, Standard Deviation [mV]: 52.56
Acquisition and analysis finished

Mean [mV]: 915.40, Standard Deviation [mV]: 52.56
```



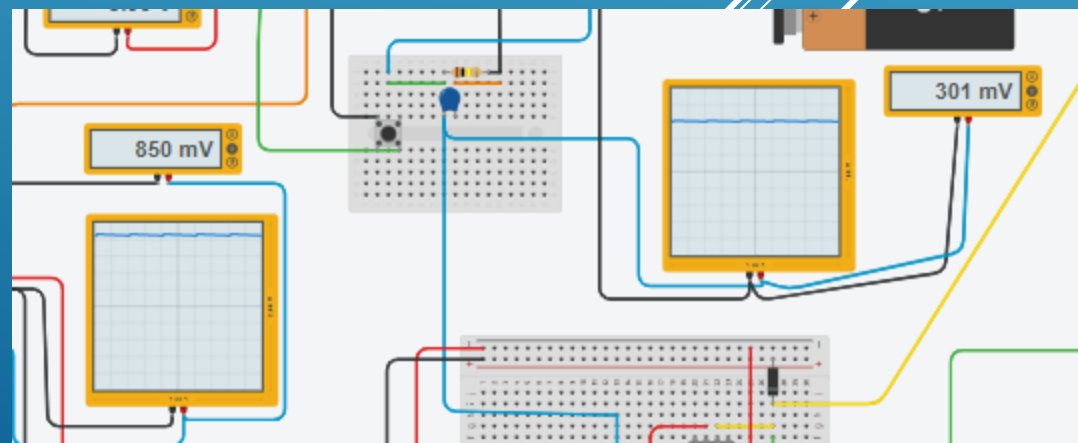
P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 6

Input signals : {-450 m[V] , 450m[V]}
triangular signal with DC offset of 0V



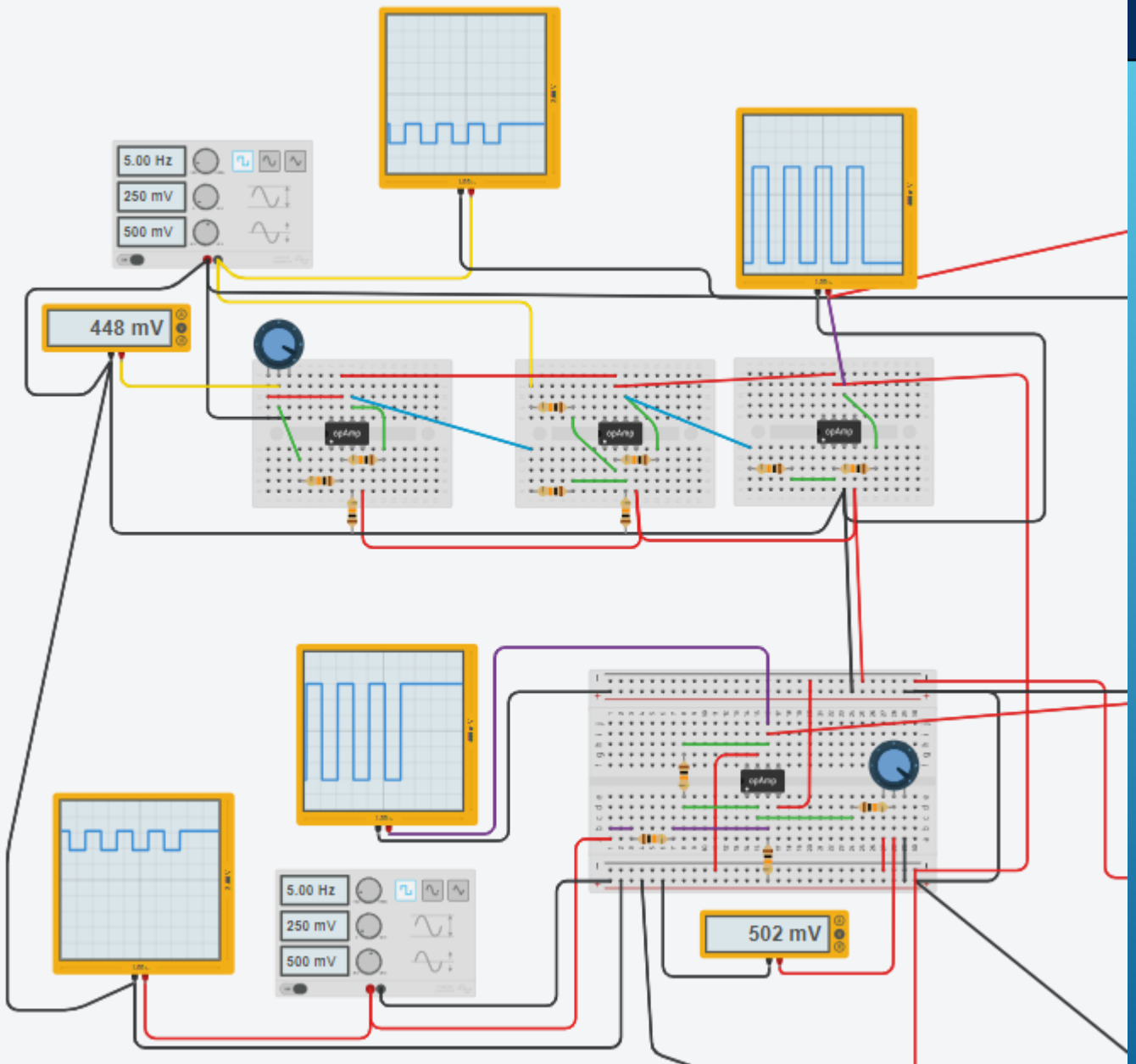
Serial Monitor

```
Acquisition and analysis are going on now

Mean [mV]: 906.68, Standard Deviation [mV]: 6.93
Mean [mV]: 906.69, Standard Deviation [mV]: 6.97
Mean [mV]: 906.69, Standard Deviation [mV]: 6.99
Mean [mV]: 906.69, Standard Deviation [mV]: 7.00
Mean [mV]: 906.70, Standard Deviation [mV]: 7.00
Acquisition and analysis finished

Mean [mV]: 906.70, Standard Deviation [mV]: 7.00
```



P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 7

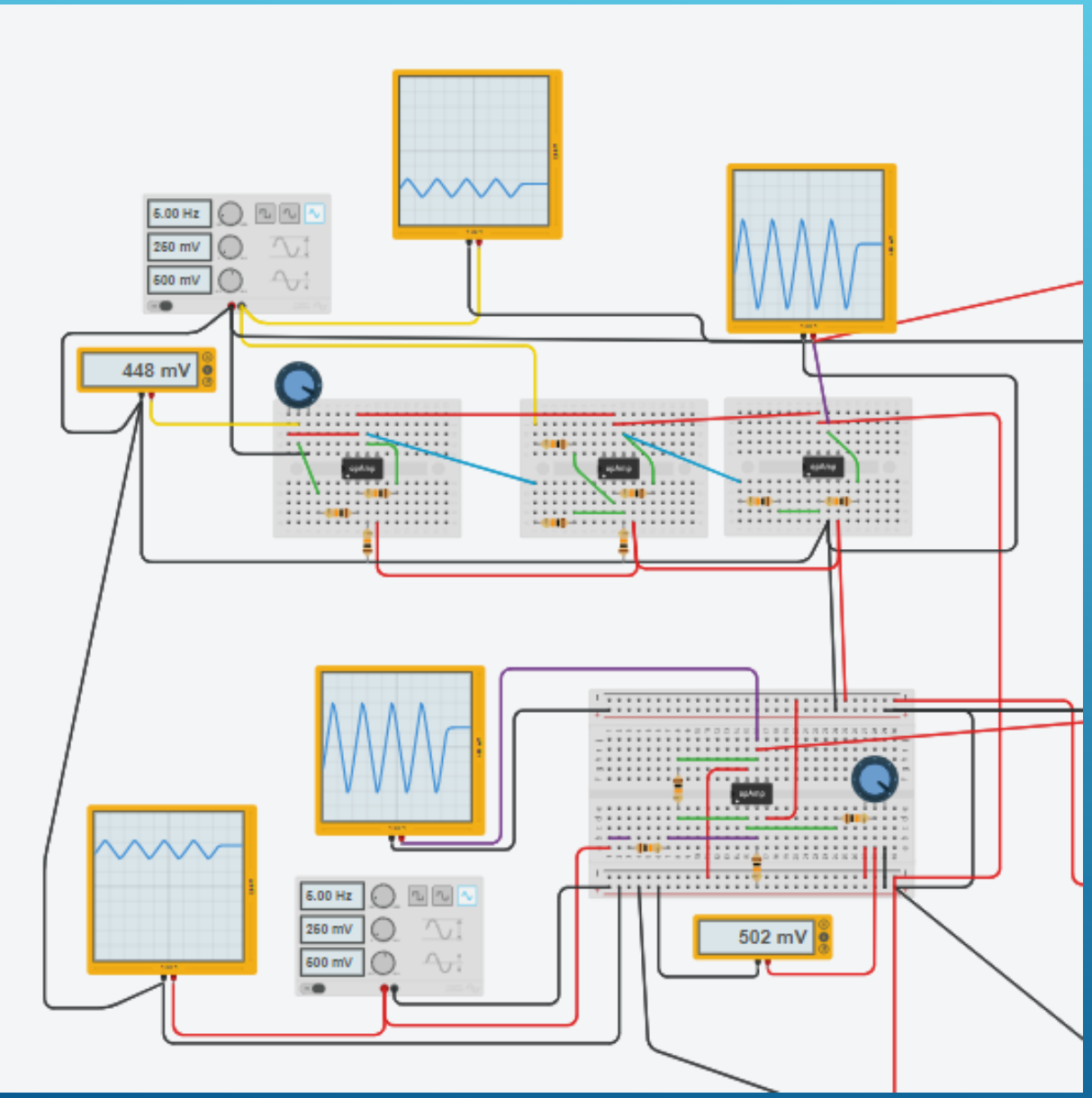
Input signals : {-250 m[V] , 250m[V]}
rectangular signal with DC offset of 0.5V



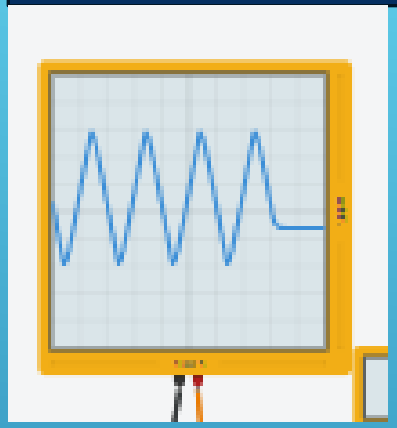
Serial Monitor

```
Acquisition and analysis are going on now

Mean [mV]: 702.93, Standard Deviation [mV]: 22.63
Mean [mV]: 665.84, Standard Deviation [mV]: 42.54
Mean [mV]: 631.24, Standard Deviation [mV]: 60.75
Mean [mV]: 599.23, Standard Deviation [mV]: 76.72
Mean [mV]: 569.31, Standard Deviation [mV]: 91.19
Acquisition and analysis finished

Mean [mV]: 569.31, Standard Deviation [mV]: 91.19
```

P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 8

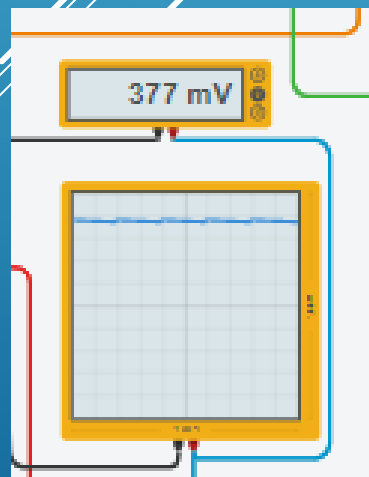


Input signals : {-250 m[V] , 250m[V]}
triangular signal with DC offset of 0.5V



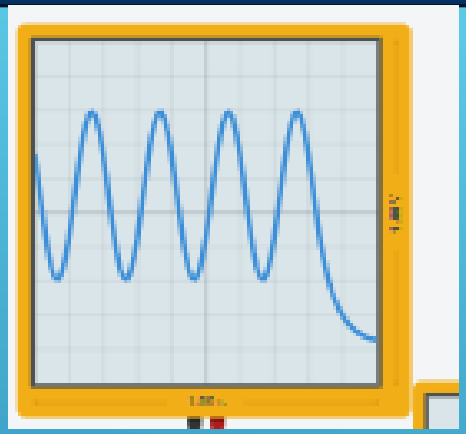
Serial Monitor

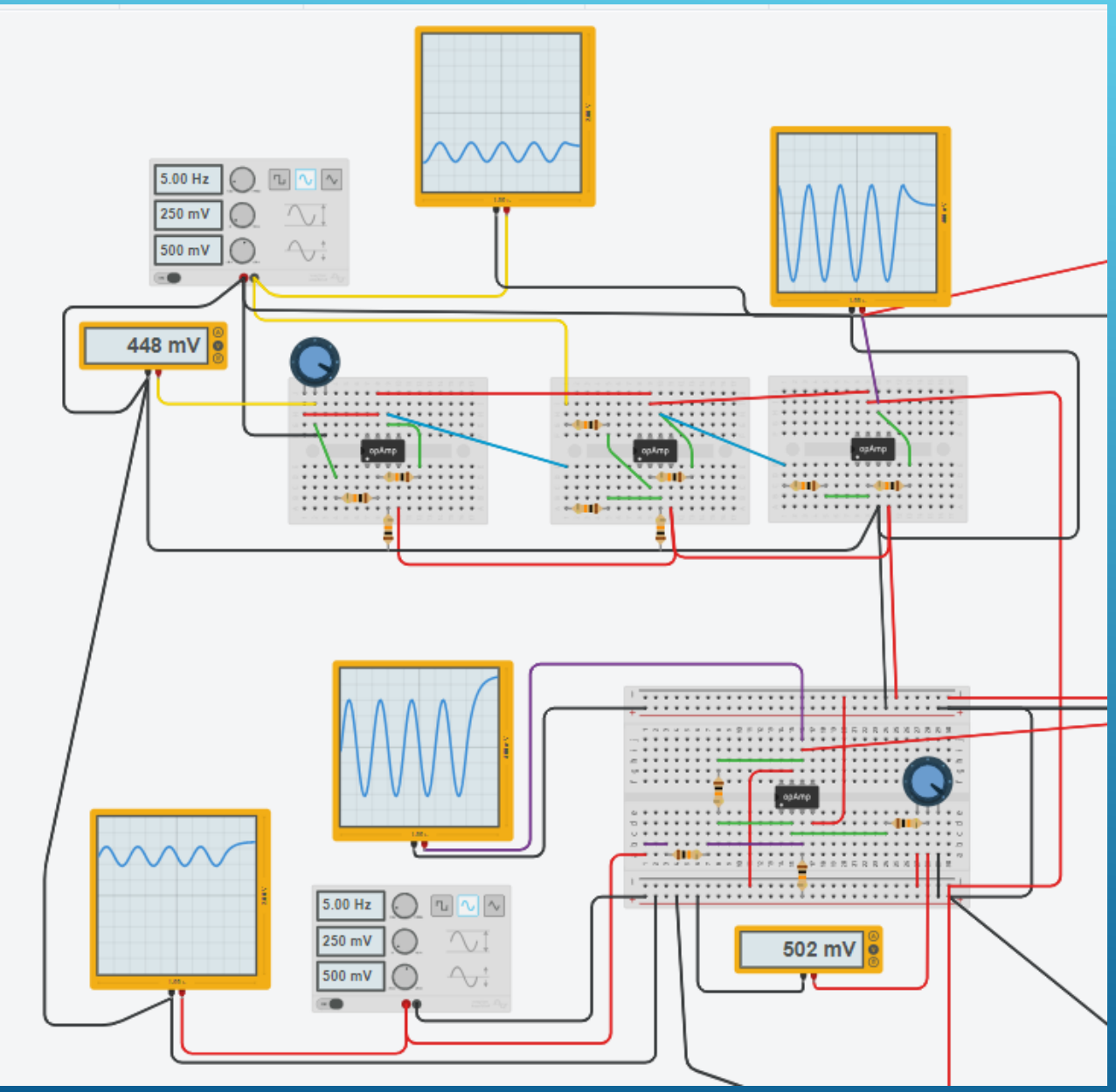
```
Acquisition and analysis are going on now

Mean [mV]: 398.03, Standard Deviation [mV]: 5.94
Mean [mV]: 398.02, Standard Deviation [mV]: 5.96
Mean [mV]: 398.01, Standard Deviation [mV]: 5.97
Mean [mV]: 398.01, Standard Deviation [mV]: 5.98
Mean [mV]: 398.01, Standard Deviation [mV]: 5.98
Acquisition and analysis finished

Mean [mV]: 398.01, Standard Deviation [mV]: 5.98
```

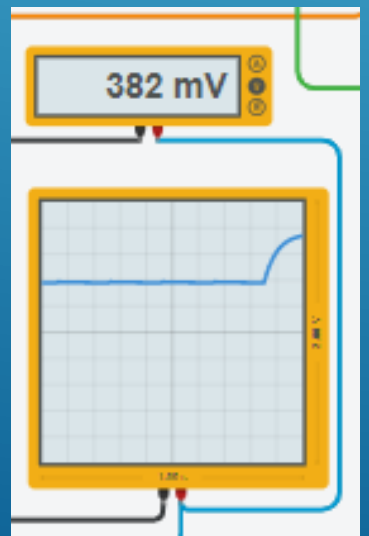


P26-PROVIDE SCREENSHOTS OF AT LEAST 12 DIFFERENT SIGNALS– PART 9

Input signals : {-250 m[V] , 250m[V]}
sinusoidal signal with DC offset of 0.5V



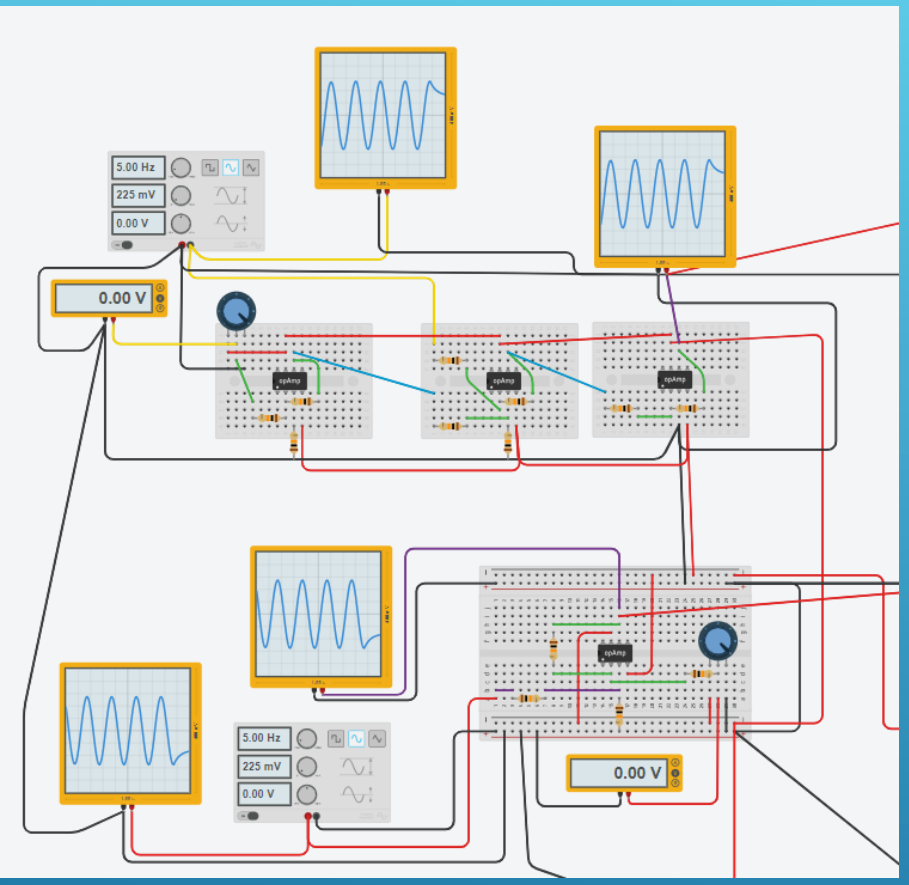




Serial Monitor

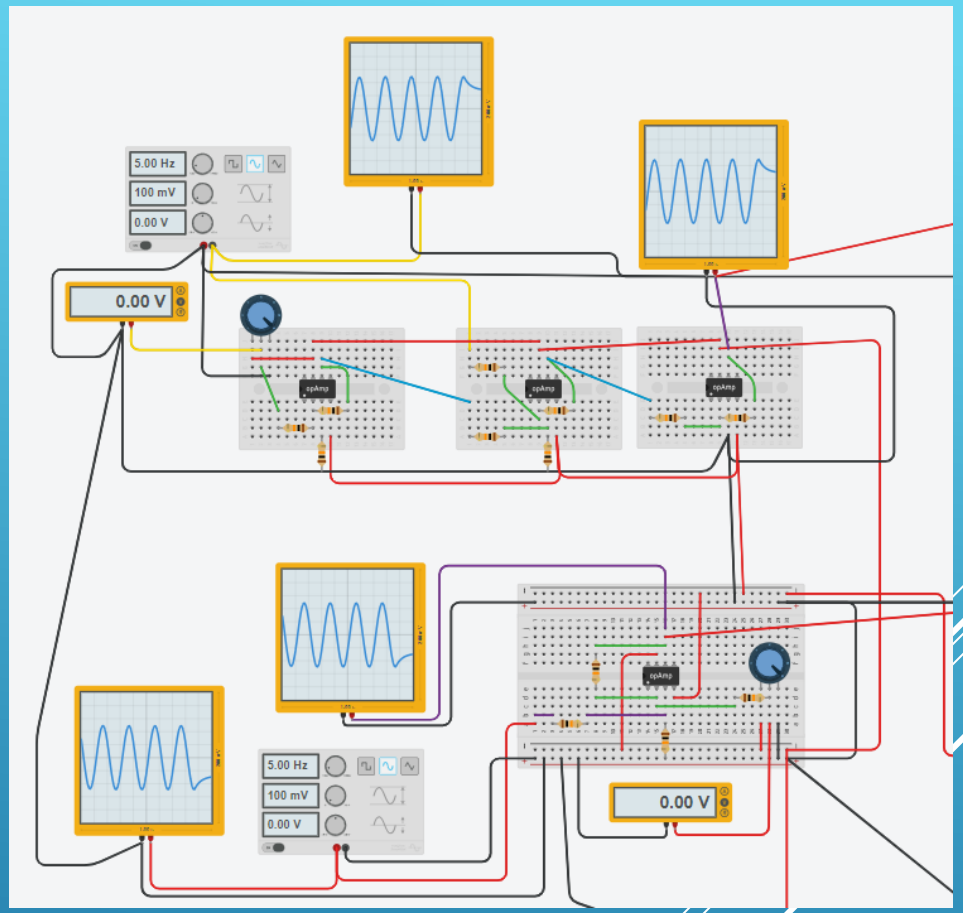
```
Acquisition and analysis are going on now

Mean [mV]: 405.45, Standard Deviation [mV]: 6.77
Mean [mV]: 405.48, Standard Deviation [mV]: 6.81
Mean [mV]: 405.48, Standard Deviation [mV]: 6.83
Mean [mV]: 405.49, Standard Deviation [mV]: 6.84
Mean [mV]: 405.49, Standard Deviation [mV]: 6.84
Acquisition and analysis finished

Mean [mV]: 405.49, Standard Deviation [mV]: 6.84
```

P27-EVALUATE MAXIMAL SPEED OF DACDCA POSSIBLE WITH TINKERCAD –PART 1



Simulator time: 00:00:00.509

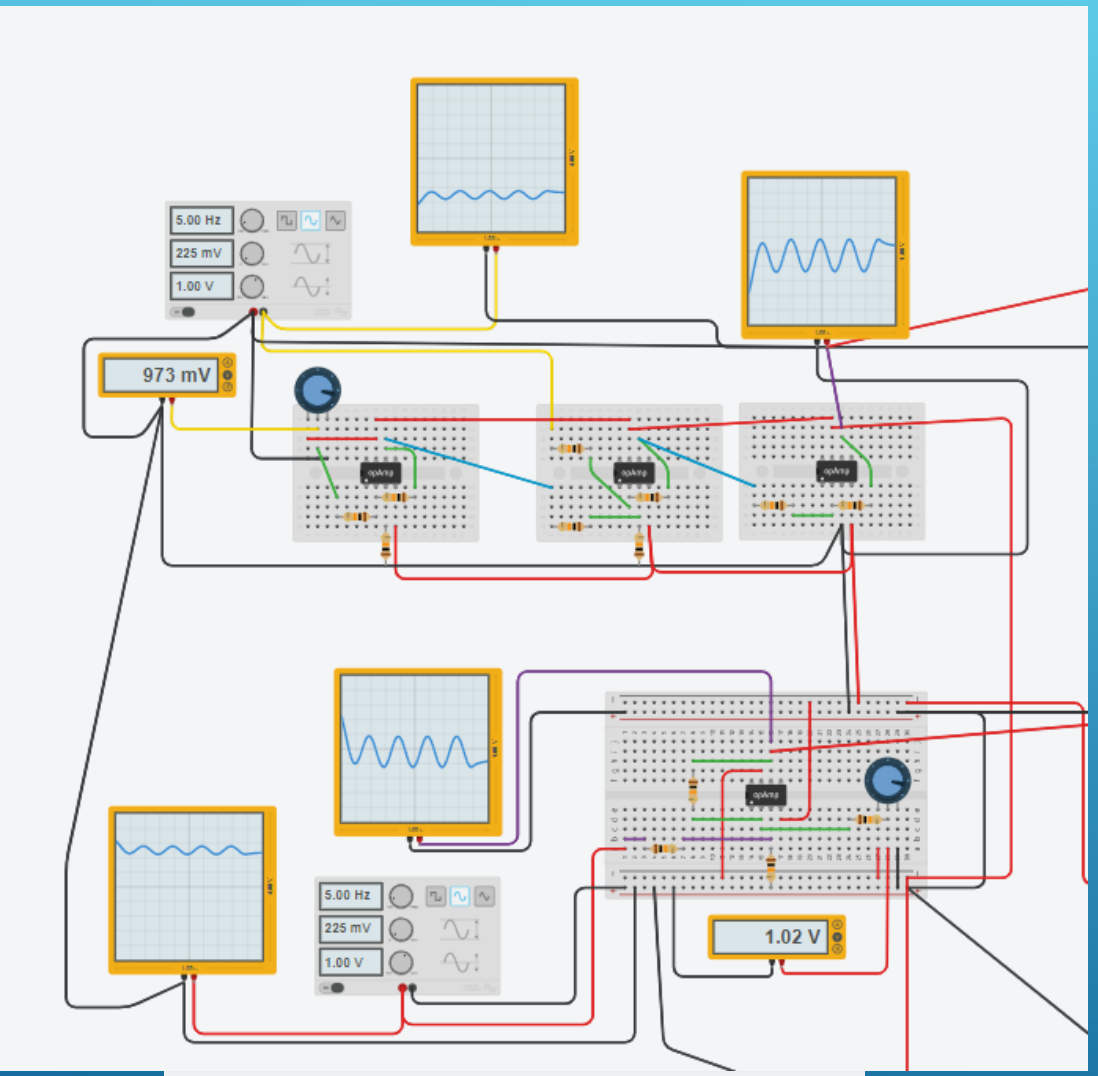
Input signals : {-225 m[V] , 225m[V]}
sinusoidal signal with DC offset of 0V



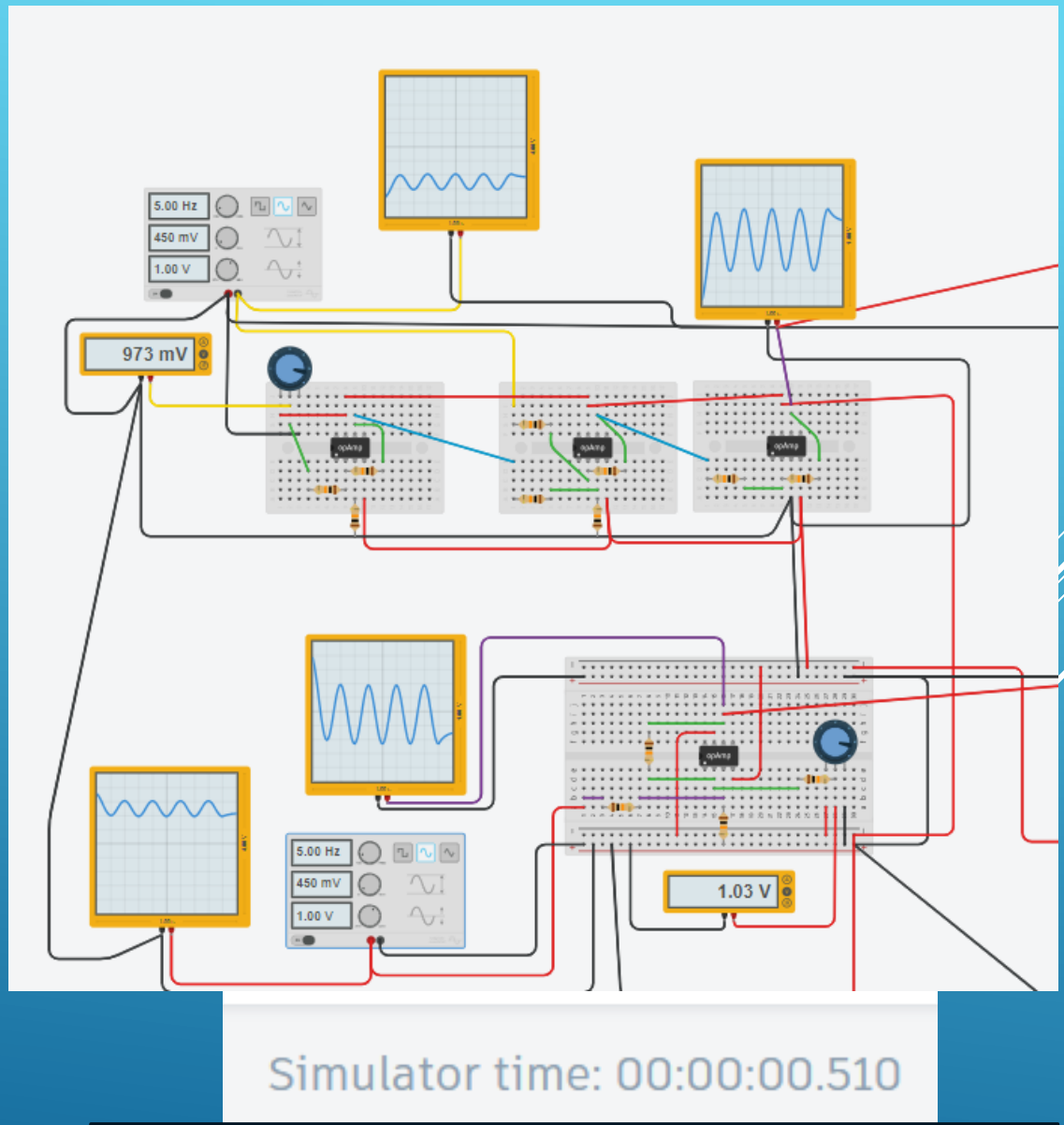
Simulator time: 00:00:00.530

Input signals : {-100 m[V] , 100m[V]}
sinusoidal signal with DC offset of 0V

P27-EVALUATE MAXIMAL SPEED OF DACDCA POSSIBLE WITH TINKERCAD –PART 2



Input signals : {-225 m[V] , 225m[V]}
sinusoidal signal with DC offset of 1V



Input signals : {-450 m[V] , 450m[V]}
sinusoidal signal with DC offset of 1V

P27-EVALUATE MAXIMAL SPEED OF DACDCA POSSIBLE WITH TINKERCAD –PART 3

Simulator time: 00:00:00.509

Simulator time: 00:00:00.530

Simulator time: 00:00:00.510

Simulator time: 00:00:00.510

These are the results of the running – time until results since pressing the simulation button ~ 0.5 sec for multiple simulations

BIBLIOGRAPHY (IEEE STYLE)-PART 1

[1] Energig, Meters, Blue Sea Systems AC Voltmeter – 0 to 150V AC.
Retrieved from https://energig.com/shop/monitoring-interfacing/meters/blue-sea-systems-ac-voltmeter-0-to-150v-ac/ on July-2021

[2] Adel S. Sedra & Kenneth C. Smith, Microelectronic Circuits, 7th Edition, New York: Oxford University Press, July 12, 2009, pp. 82-85.

[3] Elprocus, What is an Instrumentation Amplifier? Circuit Diagram, Advantages, and Applications.
Retrieved from https://www.elprocus.com/what-is-an-instrumentation-amplifier-circuit-diagram-advantages-and-applications/ on July-2021

[4] Adel S. Sedra & Kenneth C. Smith, Microelectronic Circuits, 7th Edition, New York: Oxford University Press, July 12, 2009, pp. 69-78.

[5] Wikipedia, Precision Rectifier, Improved Circuit.
Retrieved from https://en.wikipedia.org/wiki/Precision_rectifier on July-2021

[6] Physics-and-radio-electronics, Half Wave Rectifier.
Retrieved from https://www.physics-and-radio-electronics.com/electronic-devices-and-circuits/rectifier/halfwaverectifier.html on July-2021

[7] Wikipedia, Precision Rectifier, Improved Circuit.
Retrieved from https://en.wikipedia.org/wiki/Precision_rectifier on July-2021

[8] Adel S. Sedra & Kenneth C. Smith, Microelectronic Circuits, 7th Edition, New York: Oxford University Press, July 12, 2009, pp. 213-219.

[9] Electronicshub, Non Inverting Operational Amplifiers, Ravi Teja, April 9, 2021.
Retrieved from https://www.electronicshub.org/non-inverting-operational-amplifiers/ on July-2021

BIBLIOGRAPHY (IEEE STYLE)-PART 2

[10] Electronicshub, Non Inverting Operational Amplifiers, Ravi Teja, April 9, 2021.
Retrieved from https://www.electronicshub.org/non-inverting-operational-amplifiers/ on July-2021

[11] Learningaboutelectronics, Articles, Unity-gain-buffer.
Retrieved from http://www.learningaboutelectronics.com/Articles/Unity-gain-buffer on July-2021

[12] Wikipedia, Arduino UNO, July 11, 2021.
Retrieved from https://en.wikipedia.org/wiki/Arduino_Uno on July-2021

[13] Tinkercad, Arduino UNO R3 Picture.
Retrieved from https://www.tinkercad.com/ on July-2021

[14] Arduino, analogRead, July 23, 2021.
Retrieved from https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/ on July-2021

[15] Microcontrollerslab, LCD Interfacing with Arduino – Explained with Example Codes, Muhammad Bilal, April 1, 2015.
Retrieved from https://microcontrollerslab.com/lcd-interfacing-arduino-uno-r3/ on July-2021