

Computer Engineering Department

Electronics

21ELEC02C

Project Report

***Submitted to***

Dr. Hany Bastawrous & Eng. Kareem Abozeid

**ELEC. Dep., BUE**

***Students:***

*Mohamed Ahmed Abdel Galeil* ***ID:****214618*

*Mostafa Mohamed Ahmed* ***ID:*** *212577*

**COMP. Dep. DY1**

PC-based oscilloscope using Arduino

Contents

[1. Project description: 2](#_Toc128165729)

[2. Circuit Design: 2](#_Toc128165730)

[3. Components Table: 2](#_Toc128165731)

[4. Testing Setup: 3](#_Toc128165732)

[5. Testing Result: 4](#_Toc128165733)

[6. Code 5](#_Toc128165734)

# Project description:

The concept behind the PC-based Oscilloscope is that it allows us to simulate a real oscilloscope using 2 items: An Arduino, and a laptop. By connecting the Arduino and the processing circuit to a computer, the waves of the input signal can be displayed on the screen throughout the PC scope application. The maximum input voltage for the circuit is 5 V DC. It cannot accept more than 5 volts due to safety measures, as per the original author of the circuit.

# Circuit Design:

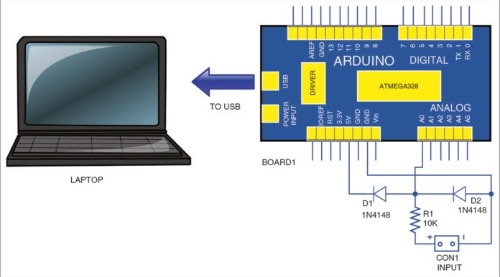


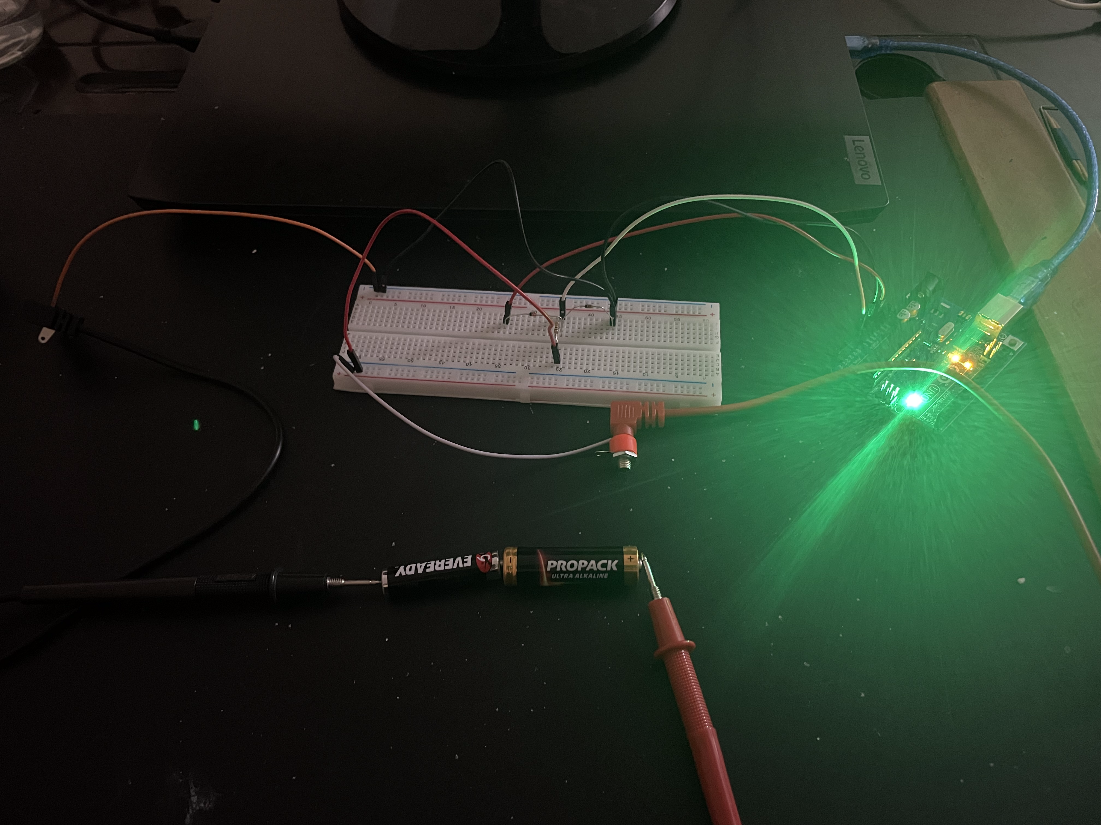
Figure 2.1: Circuit Design

# Components Table:

|  |  |
| --- | --- |
| Components | Price list |
| Arduino UNO | 230 L.E |
| Breadboard | 30 L.E |
| Ten Male-Male Jumper Wires | 7 L.E |
| Two 1N4148 Diodes | 0.7 L.E |
| 10kΩ Resistance | 1 L.E |
| Two Red & Black Probes | 16 L.E |
| Four Batteries | 35 L.E |
| PC | N/A |
| Function generator | N/A |

Figure 3.1: Components Table with Prices

# Testing Setup:



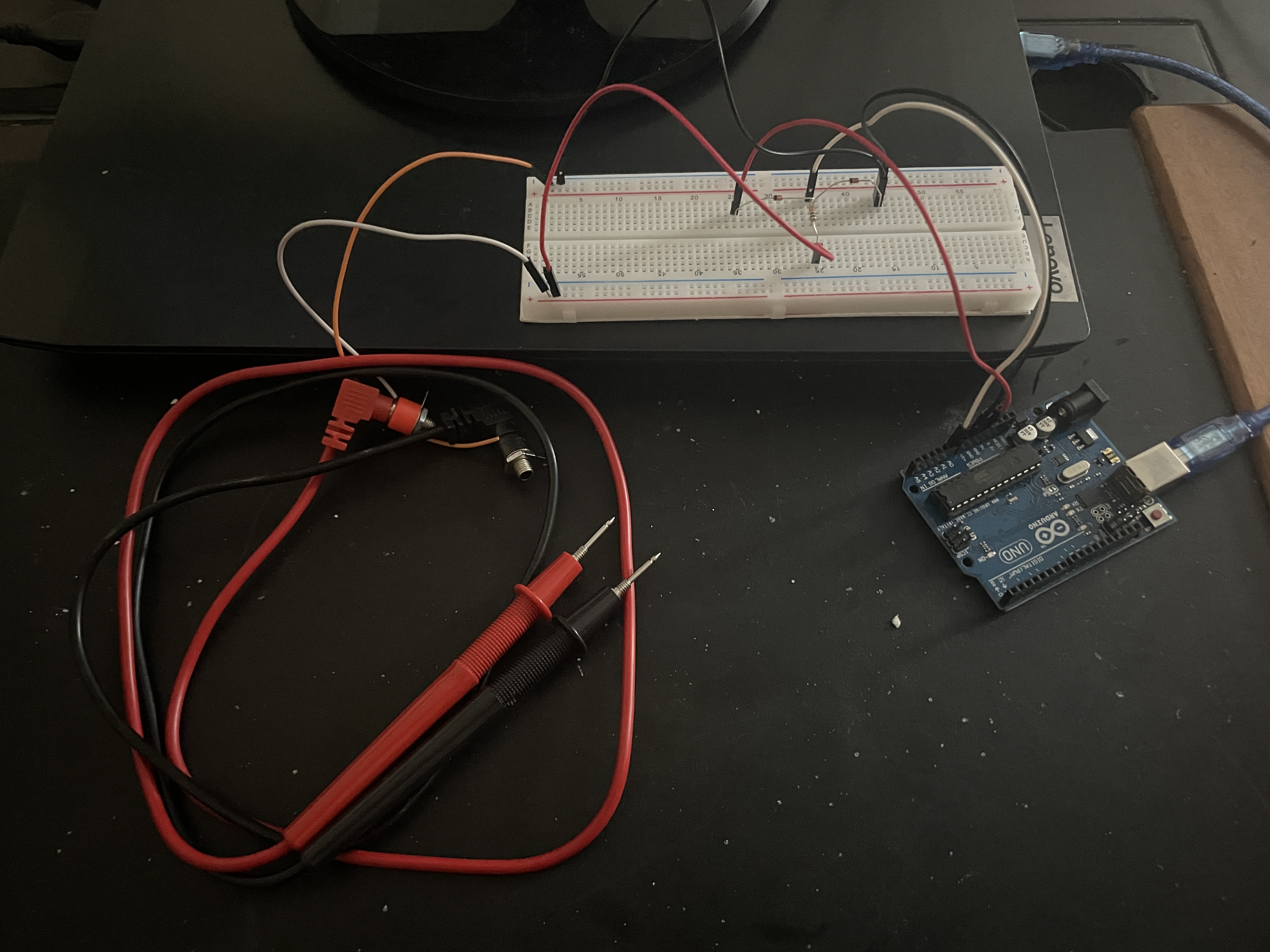


Figure 4.1: Testing Setup

Shown above is the circuit board, Arduino, and the laptop in its connected form. The probes are used so we can input our voltage, in this case 3 V DC connected in series. The breadboard is used as a base for the integrating circuit, to make it possible to connect the elements to the Arduino. To send the signal to the laptop, a USB cable exiting from the Arduino is required.

# Testing Result:

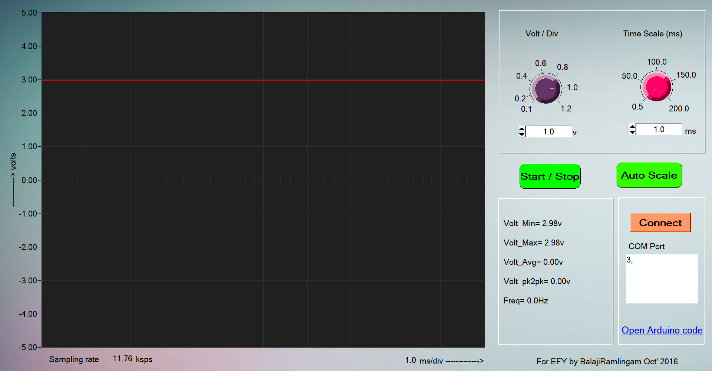


Figure 5.1: Testing Results with 3 V DC

Throughout the program shown above, 2 types of reading can be made. The first is a wave diagram, and the second is the voltage reading. As for the first case, the program is designed to show an output wave with differing shapes, according to the type of the input circuit and whether or not a function generator is included. As for the second case, the circuit is designed to show a voltage reading of up to 5 V D, due to safety reasons. The result shown above is due to an input voltage of 3 V DC (Two AAA 1.5 batteries connected in series)

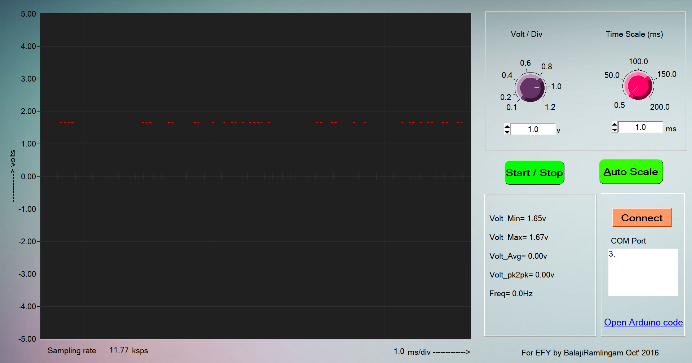


Figure 5.2: Testing Results with 1.5 V DC

The output signal shown above has a similar wave form to Figure 5.1, but it differs in magnitude. One AAA 1.5 V battery is required to achieve this wave output.

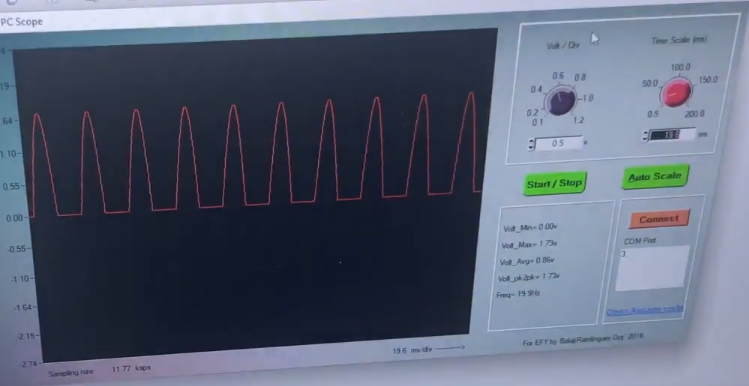


Figure 5.3: Testing Results with 3.6 V AC

Shown above is the wave output signal after a function generator was connected to the circuit. The shape in this case is a AC Sine wave of magnitude 3.6 V AC and Frequency of 50 Hz.

# Code

/\*

EFY PC OsciloScope

Read A0 and send to UART at 115200.

Read the voltages as fast as possible and send to serial port.

115200 baud pushes each byte at around 85us.

But the default ADC config by Arduino gives ADC at 116us. so here ADC config with additional lines of code and get samples faster than 85us.

Data throughput is decided by the serial baud rate

@ 115.2k baud we can get 12kSps

Created on 18th Oct 2016

by Balaji ramalingam, Robert Bosch, Banagalore

Balajir@in.bosch.com

\*/

// These constants won't change. They're used to give names

// to the pins used:

const int analogInPin1 = A0; // Analog input pin that the potentiometer is attached to

static int ctr,flag\_tog;

static unsigned char adcval;

void setup()

{

Serial.begin(115200); //bits/sec

pinMode(13, OUTPUT);

ADCSRA = 0; // clear ADCSRA register

ADCSRB = 0; // clear ADCSRB register

ADMUX |= (0 & 0x07); // set A0 analog input pin

ADMUX |= (1 << REFS0); // set reference voltage

ADMUX |= (1 << ADLAR); // left align ADC value to 8 bits from ADCH register

// sampling rate is [ADC clock] / [prescaler] / [conversion clock cycles]

ADCSRA |= (1 << ADPS2); // 16 prescaler for 76.9 KHz

ADCSRA |= (1 << ADATE); // enable auto trigger

ADCSRA |= (1 << ADIE); // enable interrupts when measurement complete

ADCSRA |= (1 << ADEN); // enable ADC

ADCSRA |= (1 << ADSC); // start ADC measurements

}

ISR(ADC\_vect) // interupts the hardware to do its function then resumes the hardware

{

adcval = ADCH; // read 8 bit value from ADC

}

void loop()

{

Serial.write(adcval);

// Following code to generate ref signal at pin 13 @ 50HZ. You can connect A0 to see the waveform in PCScope.exe

ctr++;

if(ctr>117) //117=10.03ms

{

ctr=0;

flag\_tog = !flag\_tog;

digitalWrite(13, flag\_tog);

}

}

The code is an Arduino sketch that implements a simple oscilloscope using an Arduino board. The goal of the code is to read an analog voltage from pin A0 as fast as possible and send it to a computer via a serial port. The voltage reading can then be used to visualize the signal on a computer screen or a terminal emulator.

The code uses the ADC (analog-to-digital converter) of the Arduino to read the voltage from A0. The ADC is configured to use a 16x prescaler for a sampling rate of approximately 76.9 kHz, which is faster than the Arduino's default ADC configuration. The code also enables auto-triggering and interrupts when a measurement is completed. When a voltage reading is complete, the ISR (Interrupt Service Routine) reads the 8-bit value from the ADC and stores it in the **adcval** variable.

In the **loop** function, the code continuously reads the voltage from A0 using the ADC and writes the 8-bit value to the serial port using the **Serial.write()** function. This is done as fast as possible, with a data throughput of up to 12,000 samples per second at a baud rate of 115,200 bits per second. The serial port data can then be captured and analyzed by a computer program or a terminal emulator.

In addition to the voltage reading, the code includes a block of code to generate a 50 Hz reference signal at pin 13. This can be used to observe the waveform of the signal being measured on an oscilloscope or other visualization tool.

Overall, this code is a useful starting point for building a simple oscilloscope using an Arduino board. It demonstrates how to configure the ADC for fast sampling rates and how to send data to a computer via serial communication. With additional software, the voltage readings can be visualized and analyzed in real-time, providing a powerful tool for electronics experimentation and prototyping.