5/3/2013

Parthiban Loganathan (pl2487)

Ramses Driskell (rd2491)

Introduction to Electrical Engineering 1201

Final Project Report

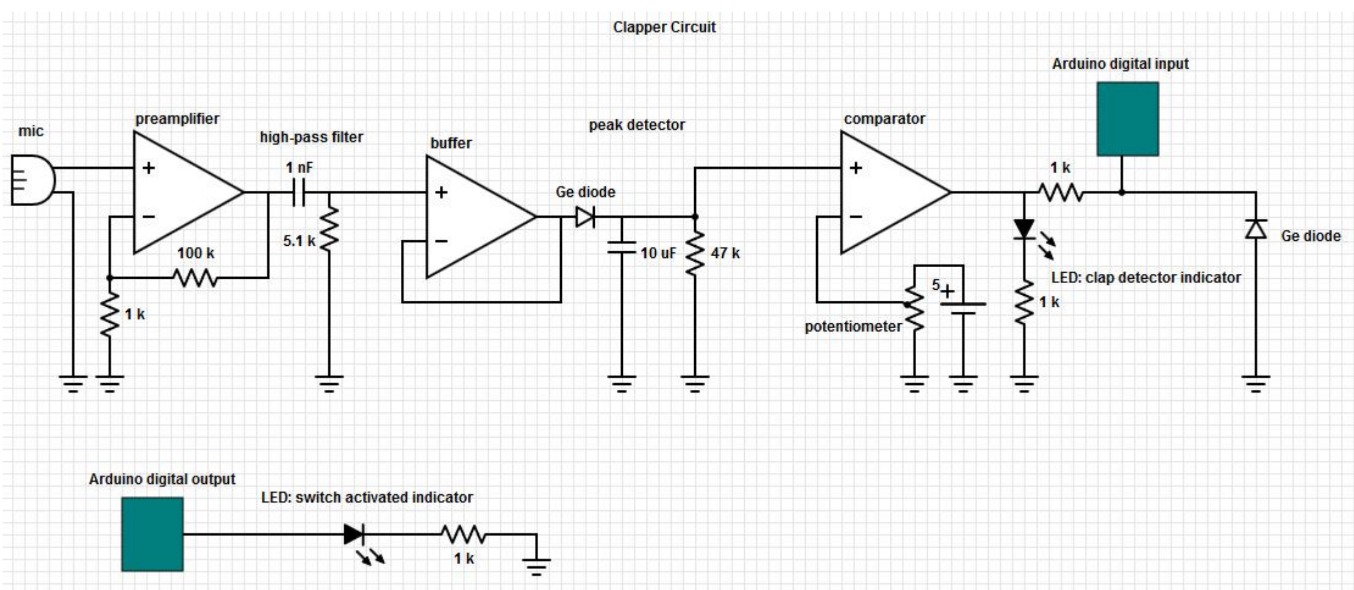
Clapper

Abstract

The clapper circuit we built is inspired by the successful 1980s gadget, The Clapper. The circuit detects a sequence of three claps within a two second time frame to activate a switch. It utilizes a microphone, preamplifer, high-pass filter, buffer, peak detector, comparator with potentiometer for volume control and an Arduino for signal processing.

Input read from a microphone is amplified with a preamplifier. Low frequencies are removed with a high-pass filter before passing the signal to a peak detector with a time constant equal to the expected time between distinct claps. These impulses trigger a comparator which feeds the pulses into an Arduino via digital logic input. The Arduino processes the signal and switches an LED on for 5 seconds if the correct sequence is recorded in a given time span.

Schematic



Stages of Circuit and Components

Preamplifier

The input from the microphone is under 100 mV. A non-inverting amplifier is used with a gain of 101 to amplify the mic input to around 2 V. This range is easier to work with in the filter and with the comparator later in the circuit. This is obtained by using resistor values of R_1 = 100 kΩ and R_2 = 1 kΩ.

Gain = $V_{OUT}/V_{IN} = 1 + R_2/R_1 = 1 + 100/1 = 101$

High-pass Filter

To prevent the clapper from registering noise as a clap, a high-pass filter is used. Low frequency sounds like human voice are removed by setting the critical frequency of the high-pass filter to a large value of 31 kHz. Claps are impulses which consist of a wide range of frequencies. The very high frequency component of a clap can pass through the filter unhindered. This guarantees that low frequencies are completely removed, and that only claps are detected.

$f_C = 1/2\pi RC = 1/2\pi*5.1*10^3*10^{-9} = 31206.8$ Hz

The output of the high-pass filter is buffered through a unity-gain amplifier to separate the high-pass filter from the peak detector. If a buffer was not used, the critical frequency would be modified by the peak detector circuit.

Peak Detector

The peak detector converts the time-varying clap into a relatively stable pulse for a given duration. This is the first step in making the clap readable as digital logic. The time constant of the peak detector is set to 0.47 s. This breaks the erratic impulse of a clap into a single pulse which dies after 2.35 s. It also sets the minimum time that must elapse between two successive claps for them to be distinguishable by the clapper. A Germanium diode with a low voltage drop of 0.2 V is used in the peak detector to prevent loss of signal amplitude.

$T = RC = 47*10^3*10*10^{-6} = 0.47$ s

The capacitor takes around 5*T = 2.35 s to completely discharge, but once the pulse drops below a certain voltage, the comparator no longer registers it as a pulse. The combination of the peak detector and the comparator in the next step determines what constitutes a pulse.

Comparator and Potentiometer

The comparator is used to convert the pulse from the peak detector into a digital logic HIGH or LOW. The comparator determines the length of each pulse. The peak detector output decays over time and the minimum threshold voltage of the comparator sets the voltage after which the decaying capacitor voltage of the peak detector does not count as a pulse. The comparator also serves as a volume threshold controller. Low volume sounds which fail to completely charge the capacitor will not produce enough voltage to trigger the comparator. The 10 kΩ potentiometer acts as a voltage divider for a source voltage of 5 V (from V_{CC} for the opamps). The negative terminal of the opamp receives the voltage output from the potentiometer and the positive terminal receives the signal from the peak detector. If the peak detector output rises above a certain threshold voltage, the comparator produces a 5 V output.

The comparator output is passed through an LED resistor combination in series to indicate that a single clap has been detected. The comparator produces -5 V when the input signal does not meet the threshold minimum. The Arduino uses TTL and expects a HIGH (5 V) or LOW (0 V) input. A Germanium diode and resistor in series are used to pull the voltage up to about 0 V when the comparator output is negative. A Germanium diode has a low voltage drop of 0.2 V which means that the digital output produced is -0.2 V as LOW and 4.8 V as HIGH. The Arduino successfully reads this as digital logic input.

Arduino Signal Processing

The Arduino reads the input as digital logic where a HIGH represents a clap detected. The code posted below waits for an initial clap and counts the number of subsequent claps. If the desired number of claps is achieved within a certain time span, the Arduino outputs a digital HIGH to an LED and resistor in series to indicate that a clap sequence has successfully been detected. The switch stays on HIGH for 5 seconds before turning off. The process is then repeated.

Code:

```
/*
Clapper Circuit
Intro to Electrical Engineering 1201 Final Project

Detects sequences of claps and turns on switch for 5 seconds
if correct sequence is identified.
*/

int in = 2;
int out = 4;
int count = 0;
int clap_flag = 0;
int switch_flag = 0;
int start = 0;
int ending = 0;
int time = 0;

void setup()
{
  Serial.begin(9600);
  pinMode(in, INPUT);
  digitalWrite(out, LOW);
  pinMode(out, OUTPUT);
}

void loop()
{
  if(count == 3 && switch_flag == 0)
  {
    ending = millis();
    Serial.print("ending: ");
    Serial.println(ending);
    switch_flag = 1;
    time = ending - start;
    Serial.print("time: ");
    Serial.println(time);
    if(time < 2000)
    {
      digitalWrite(out, HIGH);
      delay(5000);
```

```
      digitalWrite(out, LOW);
    }
    count = 0;
    clap_flag = 0;
    switch_flag = 0;
  }

  if(digitalRead(in) == 1 && clap_flag == 0)
  {
    if(count == 0)
    {
      start = millis();
      Serial.print("start: ");
      Serial.println(start);
    }

    Serial.println("clap detected");
    count++;
    clap_flag = 1;
  }
  else if(digitalRead(in) == 0 && clap_flag == 1)
  {
    clap_flag = 0;
  }
}
```

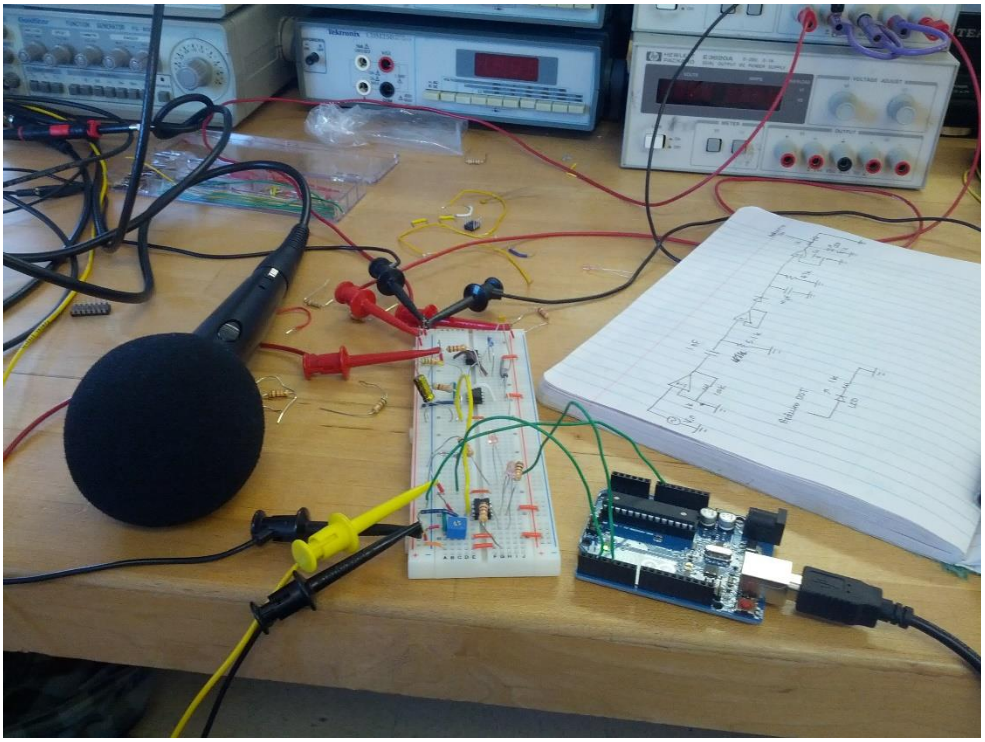
Modifications made to the circuit over the course of the experiment

Initially, a band-pass filter with a bandwidth ranging from 1.3 kHz to 3.3 kHz was used. This led to human voice and other noise registering as claps. This was changes to a high-pass filter with an extremely large critical frequency, filtering out noise. This takes advantage of the fact that claps have a wide range of frequencies.

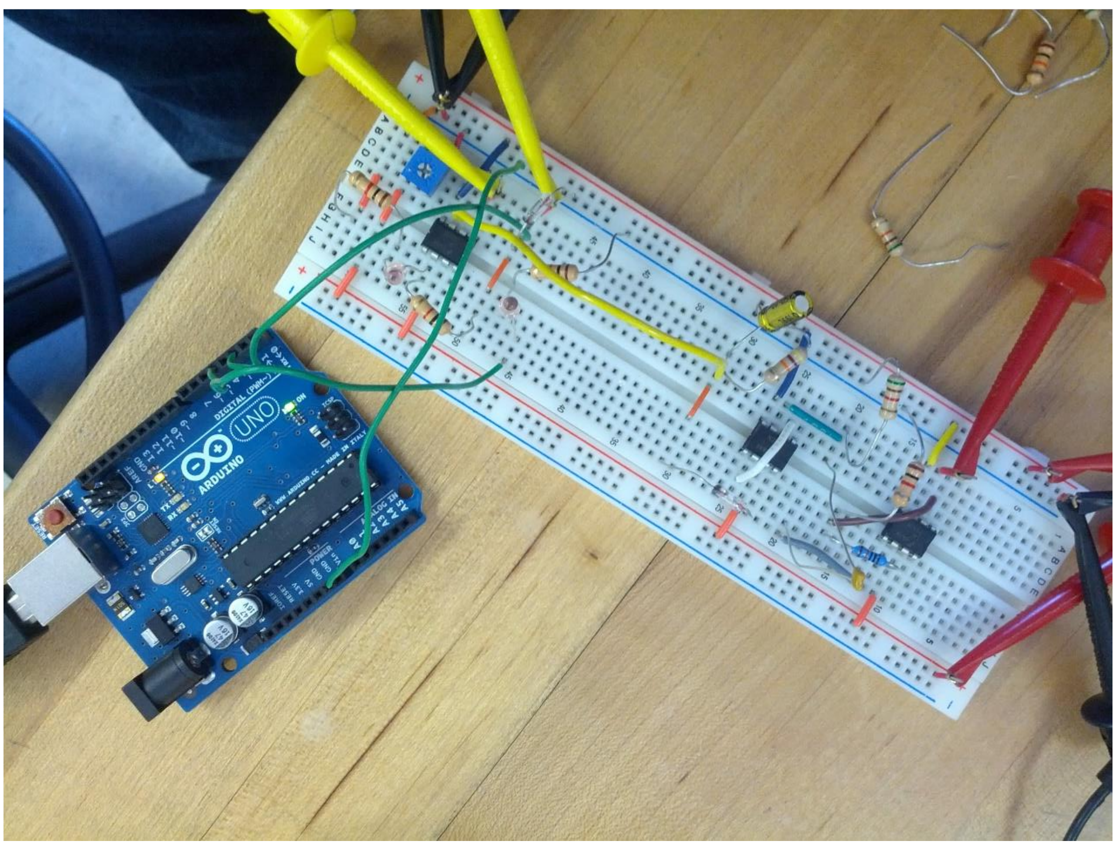
When no signal was received, the comparator output produced a -5 V output. But the Arduino reads 0 V as digital LOW. To fix this issue, a pull-up component was added to produce -0.2 V instead of -5 V, which the Arduino was able to handle.

Thanks to Prof. David Vallancourt, Zhewei Jiang and Andrew Ghazi for their valuable advice and assistance.

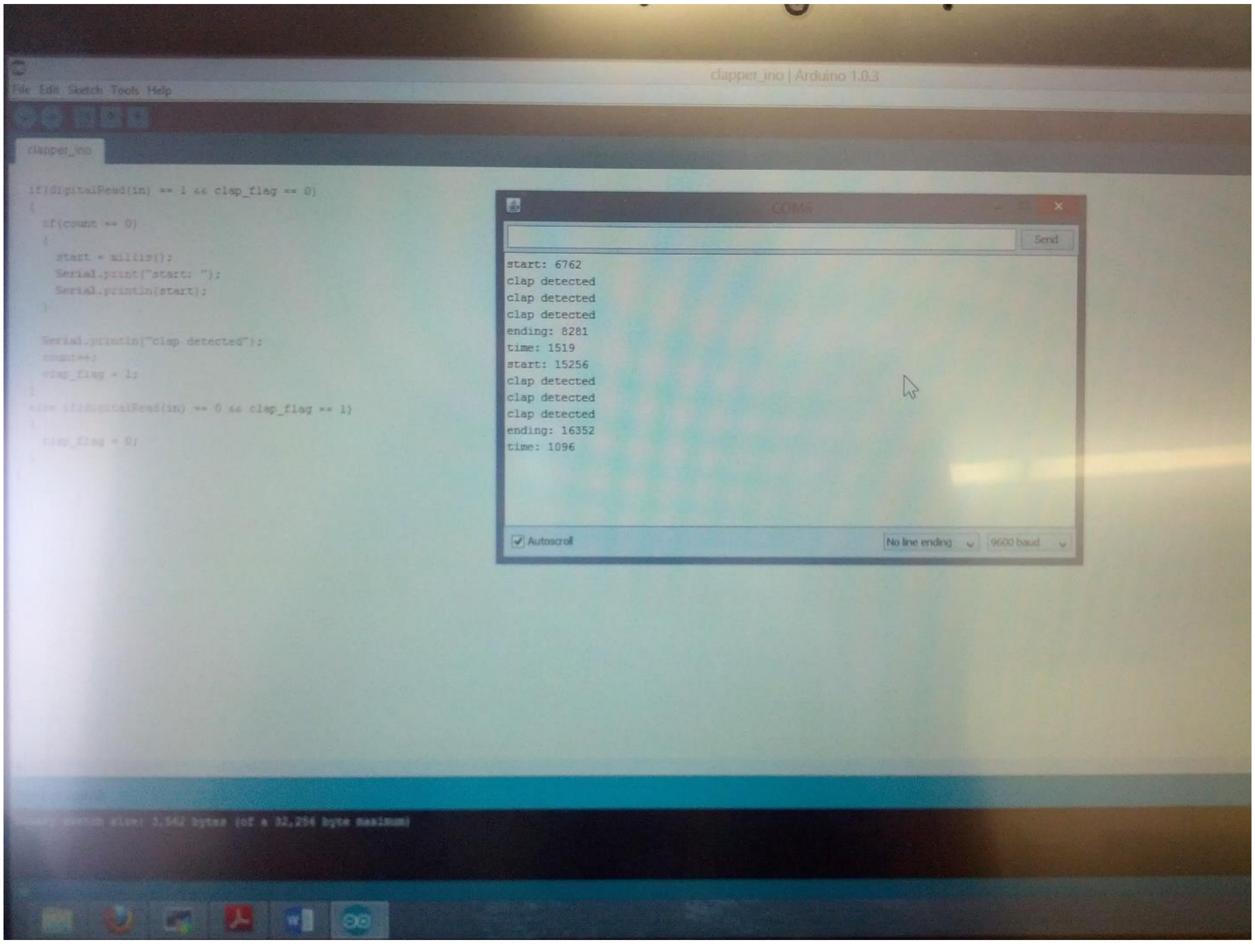
Setup:



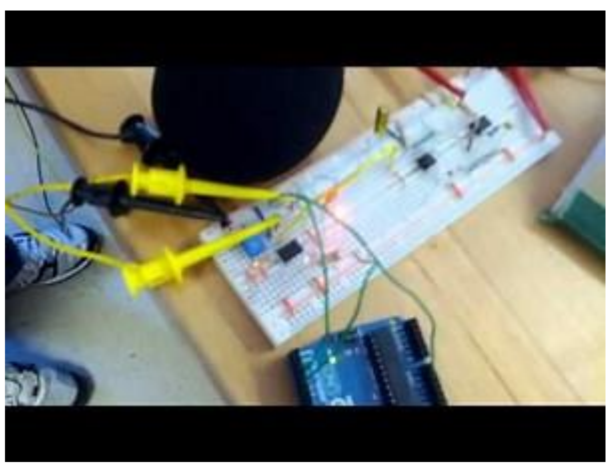
Close-up of circuit:



Arduino code running:



Demo:



If the above embedded video fails, use this link: http://youtu.be/OtIiI1gSooY