Intruder Detection using Geophone Sensor:

Introduction

In this part of Project, we are using a Geophone sensor, which is essentially a seismic sensor which detects vibration of the ground. They are typically placed underground using a mounting case to detect seismic waves in the certain region. We are using this sensor to capture the foot step vibrations of a person walking in the vicinity of the sensor. Using that Vibration signature we can tell various about the person walking like at how much distance the person is etc. but we are limiting ourselves to the detection of a person near its vicinity. The use of this technology is to detect any intruder entering the restricted area.

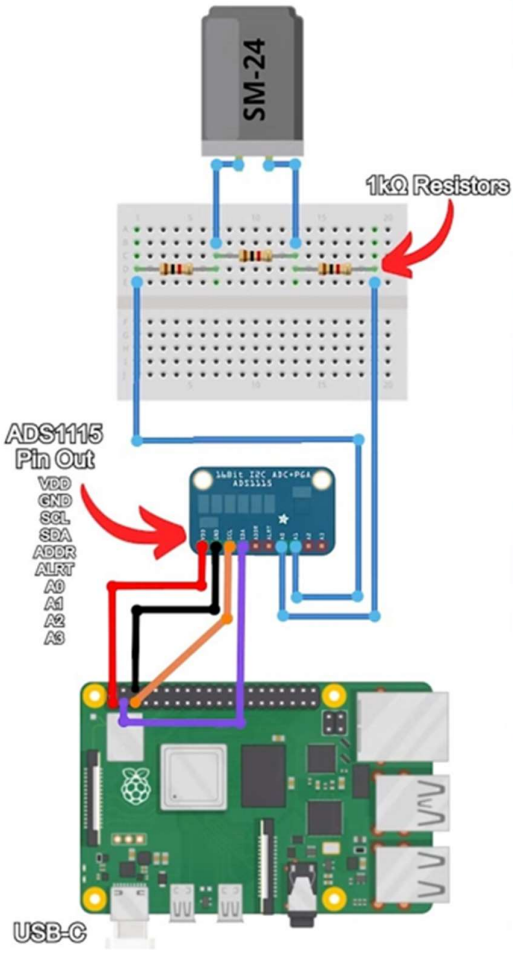
Hardware used

1. ***Geophone Sensor:*** Seismic geophone is a type of sensor that converts ground movement or a kind if vibration into voltage, which could be acquired by the acquisition system.

2. ***Raspberry pi*** (for processing the incoming data though ADC)

3. ***ADC (ADS1115)*** to convert incoming analogue signal from sensor to a digital signal which is then fed to the raspberry pi.
   Configuration of ADC: 16-bit 4 channel ADC

Circuit Diagram:



CODE:

❖ Importing all Libraries

```
#import matplotlib.pyplot as plt
#import matplotlib.animation as animation
import wave
import Adafruit_ADS1x15
from scipy.stats import kurtosis
import numpy as np
import pylab as p
from scipy.stats import skew
from scipy import signal
import sys
import time
import random
import datetime
import telepot
import urllib.request
import RPi.GPIO as GPIO
```

❖ Setting up GPIO Port and the LED

- Three continuous LED blinks indicates: Raspberry pi is connected to Internet
- No Blink with LED ON indicates: no internet connection
- After Network test is over for every step detected the LED will Blink once
-

```
❖
❖ ################### LED ##############################
❖ GPIO.setmode(GPIO.BCM)
❖ GPIO.setwarnings(False)
❖ GPIO.setup(17,GPIO.OUT)
❖
❖ def connect():
❖     try:
❖         urllib.request.urlopen('http://google.com') #Python 3.x
❖         return True
❖     except:
❖         return False
❖
❖ if connect():
❖     GPIO.output(17,GPIO.LOW)
❖     print("LED on")
❖     GPIO.output(17,GPIO.HIGH)
❖     time.sleep(0.5)
❖     print("LED off")
❖     GPIO.output(17,GPIO.LOW)
```

```
    print("LED on")
    time.sleep(0.5)
    GPIO.output(17,GPIO.HIGH)
    time.sleep(0.5)
    print("LED off")
    GPIO.output(17,GPIO.LOW)
    print("LED on")
    time.sleep(0.5)
    GPIO.output(17,GPIO.HIGH)
    time.sleep(0.5)
    print("LED off")
    GPIO.output(17,GPIO.LOW)

else:
    print("LED on")
    GPIO.output(17,GPIO.HIGH)



print( 'connected' if connect() else 'no internet!' )

```

- Telegram Bot for notifying the human detection to the owner via telegram server

```

############ TELEGRAM BOT#######################

chat_id = 603890797

def handle(msg):
    global chat_id
    chat_id = msg['chat']['id']
    command = msg['text']

    print ("Got Chat ID: %s" % chat_id)
    print ("Got command: %s" % command)

```

❖ Code to configure ADC:

```
❖
❖  ######################## ADC ##############################
❖  adc = Adafruit_ADS1x15.ADS1115(address=0x48, busnum=1)
❖
❖  GAIN = 16
❖
❖  Start = 0
❖  fs = 1000
❖  # starting time
❖  t0 = time.time()
❖
❖  # End time
❖  t_end = time.time() + (60 * 60 * 24)
❖
❖  lst= []
❖  N=0
❖  samples = 256
❖
```

❖ Sensing part:

```
❖
❖  ################# SENSING ##############################
❖  #To do a whole day (24hours) of recording data you would want to Comment
   out the above While Loop and Uncomment the Below Loop
❖  while (time.time() < t_end):
❖
❖      # Read the difference between channel 0 and 1 (i.e. channel 0 minus
   channel 1).
❖      # Note you can change the differential value to the following:
❖      #  - 0 = Channel 0 minus channel 1
❖      #  - 1 = Channel 0 minus channel 3
❖      #  - 2 = Channel 1 minus channel 3
❖      #  - 3 = Channel 2 minus channel 3
❖      value = adc.read_adc_difference(1, gain=GAIN, data_rate = 128)
❖      # Note you can also pass an optional data_rate parameter above, see
❖      # simpletest.py and the read_adc function for more information.
❖      # Value will be a signed 12 or 16 bit integer value (depending on the
   ADC
❖      # precision, ADS1015 = 12-bit or ADS1115 = 16-bit).
❖     # f.write(str((value))+'\n')
❖
❖      #Add one to the counter
❖      Start = Start + 1
```

```
    # print(value)
    #print('Channel 0 minus 1: {0}'.format(value))
    # Pause for half a second.
    N=N+1
    if(N < samples):
        lst.append(value)
        N = len(lst)

    elif(N == samples):
        data = lst
        E = (1/fs )*(np.sum(np.  multiply(lst,lst)))
        data_norm = lst/E
        kur = kurtosis(lst, fisher=True)

        #print(f)
        #print(skew(data_norm,axis=0,bias=True))
        print(kur)
        if(kur>5):
            print("footstep detected")
            GPIO.output(17,GPIO.HIGH)
            time.sleep(0.5)
            print("LED off")
            GPIO.output(17,GPIO.LOW)
            bot =
telepot.Bot('5567383577:AAGo8PNoJWE2s3N4Xts8iFyjd0m83GTFC9g')
            bot.message_loop(handle)
            # sarvaji _bot bot.sendMessage(603890797, "IoT
Project(Geophone)")
            bot.sendMessage(603890797, "Intruder detected (Geophone)")

            print ("I am listening...")
        elif():
            print()
    elif(N > samples) :
        lst=[]
        N=0



#As soon as the loop is complete the total time taken for looping to
complete is determined and recorded
t1=time.time()
total= t1-t0

#This time is then printed to shell.
#print('This is the time in seconds it took to complete data collection '+
str(total))
```