**Vibration Triggered Security Surveillance System Design**

**INTRODUCTION**

Necessity of Security Surveillance system is the insecurity and crime constitute faced by our immediate society today. People live with fear of being attacked by thieves and burglars. Despite of all the effort, resources and time that has been developed the problems are still on the increase which gave rise to the need for an increasing development in the technology of alarm system which utilizes infrared motion detection, light (photo) sensitive electronic devices and so on. Even with the introduction of these alarm systems which have reduced greatly the level of insecurity, there is still a problem of false alarm which needs to be minimized. In order to effectively increase the level of security and to avoid false alarms which can create unnecessary unrest and to create and design a cost effective security system, a ***vibration triggered security system*** is required. This system if properly designed will provide security and ensure alarms are activated only when unauthorized person trying to gain access to the protected material (door or entrance of a living area) based on the intensity at which the material is vibrating.

Here, the system has been implemented in such a manner that if there is any vibration sensed for the protected material (door or entrance of a living area or any protected area) the user will be informed through mail and the web-cam system will take snap of the intruder standing in front of the material of observation.

**SYSTEM DESIGN AND IMPLEMENTATION**

Here the design procedure and the basic theory of components used for this project work. This section is further divided into sub-sections on basis of design theory and system design analysis.

**Stages of Implementation.**

The system design was implemented in three units:

1. Developing algorithm to detect vibration.
2. Interfacing components and embedding the developed algorithm in used micro controller.
3. Interfacing operating system with Arduino and sending mail using ***Mailgun transactional Email API Service***.

**DEVELOPING ALGORITHM TO DETECT VIBRATION**

**What is Vibration?**

A body is said to vibrate when it describes an oscillating motion about a reference position. The number of times a complete motion cycle takes place during the period of one second is called the Frequency and is measured in hertz (Hz). The motion can consists of a single component occurring at a single frequency, as with a tuning fork, or of several components occurring at different frequencies simultaneously, as for example, with the piston motion of an internal combustion engine. Vibration signals in practice usually consist of very many frequencies occurring simultaneously so that we cannot immediately see just by looking at the amplitude-time pattern, how many components there are, and at what frequencies they occur.

These components can be revealed by plotting vibration amplitude against frequency. The breaking down of vibration signals into individual frequency components is called frequency analysis, a technique which may be considered the cornerstone of diagnostic vibration measurements. The graph showing the vibration level as a function of frequency is called a frequency spectrogram.

When frequency analysing machine vibrations we normally find a number of prominent periodic frequency components which are directly related to the fundamental movements of various parts of the machine. With frequency analysis we are therefore able to track down the source of undesirable vibration.

**The Vibration Parameters, Acceleration, Velocity and Displacement.**

When we looked at the vibrating tuning fork we considered the amplitude of the wave as the physical displacement of the fork ends to either side of the rest position. In addition to Displacement we can also describe the movement of the fork leg in terms of its velocity and its acceleration. The form and period of the vibration remain the same whether it is the displacement, velocity or acceleration that is being considered. The main difference is that there is a phase difference between the amplitude-time curves of the three parameters as shown in the drawing. For sinusoidal signals, displacement, velocity and acceleration amplitudes are related mathematically by a function of frequency and time, this is shown graphically in the diagram. If phase is neglected, as is always the case when making time-average measurements, then the velocity level can be obtained by dividing the acceleration signal by a factor proportional to frequency, and the displacement can be obtained by dividing the acceleration signal by a factor proportional to the square of frequency. This division is performed by electronic integrators in the measuring instrumentation. The vibration parameters are almost universally measured in metric units in accordance with ISO requirements, these are shown in the table.

**Considerations in choosing Acceleration, Velocity, or Displacement parameters.** By detecting vibratory acceleration we are not tied to that parameter alone, with electronic integrators we can convert the acceleration signal to velocity and displacement. Most modern vibration meters are equipped to measure all three parameters. Where a single, wide frequency band vibration measurement is made, the choice of parameter is important if the signal has components at many frequencies. Measurement of displacement will give the low frequency components most weight and conversely acceleration measurements will weigh the level towards the high frequency components. Experience has shown that the overall RMS value of vibration velocity measured over the range 10 to 1000 Hz gives the best indication of a vibration's severity. A probable explanation is that a given velocity level corresponds to a given energy level so that vibration at low and high frequencies are equally weighted from a vibration energy point of view. In practice many machines have a reasonably flat velocity spectrum. Where narrow band frequency analysis is performed the choice of parameter will be reflected only in the way the analysis plot is tilted on the chart paper (as demonstrated in the middle diagram on the opposite page). This leads us to a practical consideration that can influence the choice of parameter. It is advantageous to select the parameter which gives the flattest frequency spectrum in order to best utilise the dynamic range (the difference between the smallest and largest values that can be measured) of the instrumentation. For this reason the velocity or acceleration parameter is normally selected for frequency analysis purposes. Because ***acceleration measurements are weighted towards high frequency vibration components, this parameters tends to be used where the frequency range of interest covers high frequencies.***

The nature of mechanical systems is such that appreciable displacements only occur at low frequencies, therefore displacement measurements are of limited value in the general study of mechanical vibration. Where small clearances between machine elements are being considered, vibratory displacement is of course an important consideration. Displacement is often used as an indicator of unbalance in rotating machine parts because relatively large displacements usually occur at the shaft rotational frequency, which is also the frequency of greatest interest for balancing purposes.

**Vibration Detection Technique used in the Project:**

From the above discussion it is clear that ***acceleration of a particular body is a vital parameter for calculating the vibration of the body (while the frequency of the body is high). Occurrence of low frequency has not considered as the low frequency vibration may occur because of very slow and gradual movement of the body (mostly caused by wind considered as noise component).***

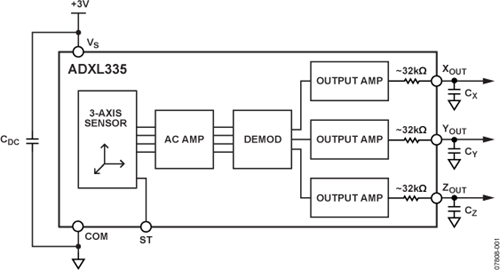
So, any high frequency vibration can be detected by detecting any sudden change in the acceleration of the body.

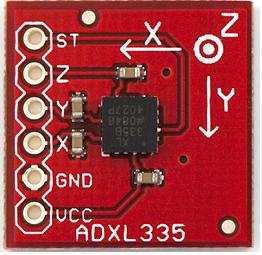
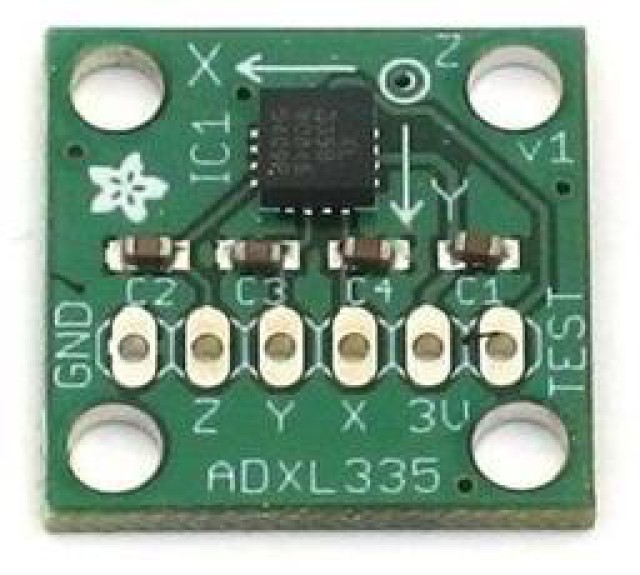
**Components Required:**

**Accelerometer (ADXL-335):** The ADXL335 is a small, thin, low power, complete 3-axis accelerometer with signal conditioned voltage outputs. The product measures acceleration with a minimum full-scale range of ±3 *g*. It can ***measure the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion, shock, or vibration***.

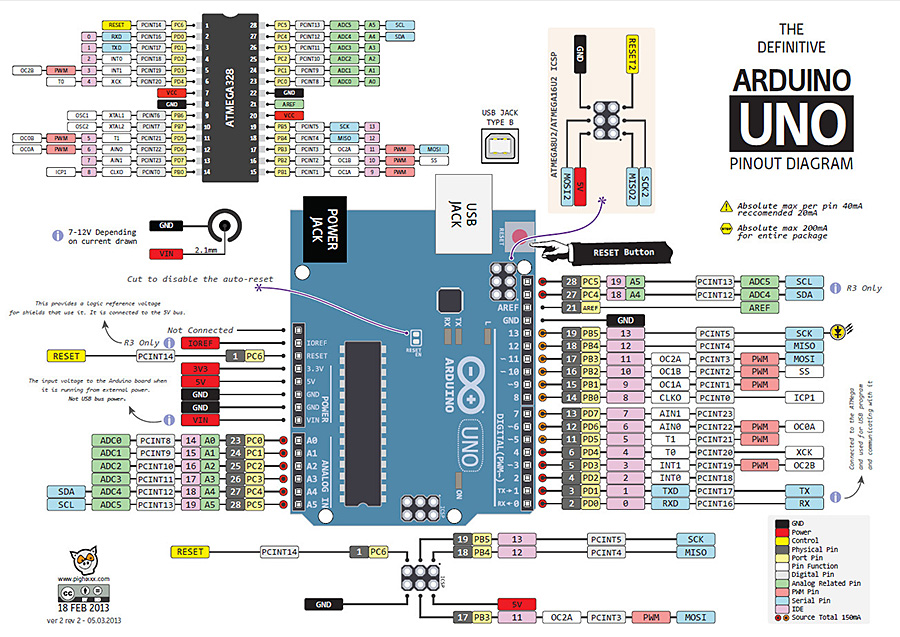
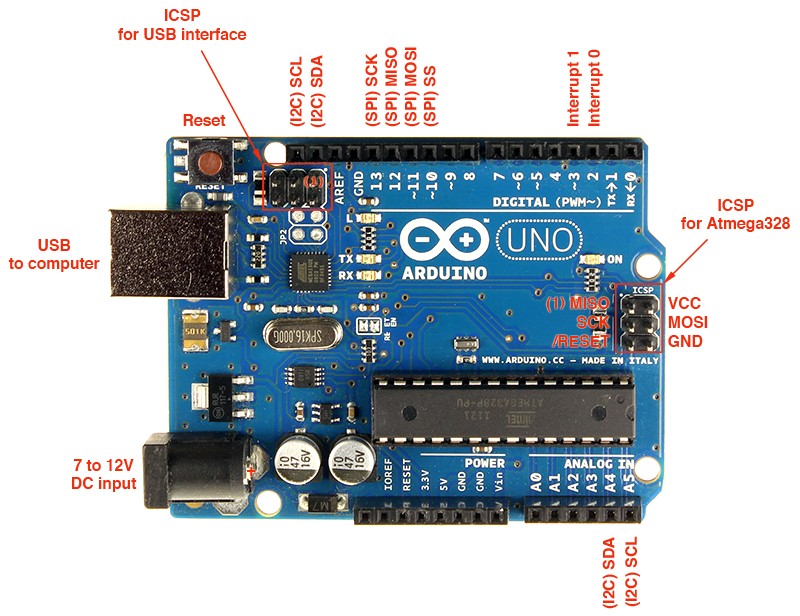
The user selects the bandwidth of the accelerometer using the CX, CY, and CZ capacitors at the XOUT, YOUT, and ZOUT pins. Bandwidths can be selected to suit the application, with a range of 0.5 Hz to 1600 Hz for the X and Y axes, and a range of 0.5 Hz to 550 Hz for the Z axis.

The ADXL335 is available in a small, low profile, 4 mm × 4 mm × 1.45 mm, 16-lead, plastic lead frame chip scale package (LFCSP\_LQ).





**Arduino UNO:** The Arduino UNO R3 is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller. ***The standard arduino runs at 16MHz, that is the clock goes 16,000,000 times a second and some instructions can run in just one clock cycle.***



**Developing the Algorithm:** The accelerometer ADXL335 not only measures the dynamic acceleration of a body but also it measures the static acceleration of gravity which is not required in the application as mentioned earlier vibration of a body (both natural vibration and the vibration caused by external momentum) is sinusoidal and periodic in nature. So, natural acceleration of the body is also sinusoidal in nature;

Let, the displacement of the body from the reference point can be considered as,

Where A=amplitude, f=frequency and t=time

Deriving the equation by t;

Deriving the equation again,

From the above equation it is quite clear that ***the acceleration of a vibrating body is proportional to square of the frequency at which the body is vibrating, when the amplitude and phase is constant. This is the reason why acceleration is used as a parameter for the detection of high frequency vibration detection.***

So, considering the static acceleration because of gravity the ideal output given by the accelerometer is,

Similarly, for y and z axis,

As, the processing time of a loop in arduino is finite, so the computation of reading the analog pins connected to the accelerometer is digital in nature and it obtains sample values while the sampling time is equal to the computation time of the loop in algorithm (***computation time depends on the machine cycle each operation is taking inside the loop***). As, the computation rate of the arduino micro-processor is high (***the standard arduino runs at 16MHz, that is the clock goes 16,000,000 times a second and some instructions can run in just one clock cycle***), the sampling time or computation time is less, which directly implies that the sampling rate is much more higher than the frequency of vibration; which leads to the decision that in the system follows the Nyquist criteria of sampling.

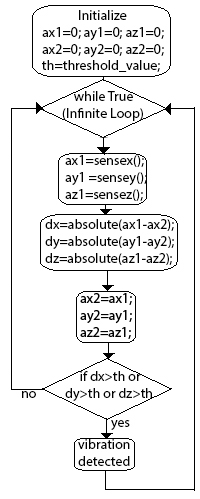
Suppose, at any time instant t1 the output measured in the x direction is outputx1 and in the next time instant the measured output in the x direction is outputx2. So,

So, the static acceleration value can be omitted by getting the difference between this two instances. The difference is,

As, the computation rate is high the change in amplitude will be less and the change in frequency will be negligible if no external momentum is applied.

If, external momentum is applied in between the two instances of time the frequency of the body will be the result of superposition of the frequency of the body and the external momentum and the change in amplitude will be high. So, ***vibration caused by external momentum can easily be detected using a threshold filter on the absolute value of Δoutputx***.

**Flow Chart Diagram:** Let, (ax1, ay1, az1) =current accelerometer value; (ax2, ay2, az2) =previous value.

****

**INTERFACING COMPONENTS AND EMBEDDING THE ALGORITHM**

**Arduino Pin Layout:**



**Arduino Code:**

int sensePinX = 8; int sensePinY = 9;

int sensePinZ = 10; int ledPinX = 10;

int ledPinY = 11; int ledPinZ = 12;

int powerPin = 13; int th = 10;

void setup() {

pinMode(sensePinX, INPUT);

pinMode(sensePinY, INPUT);

pinMode(sensePinZ, INPUT);

pinMode(ledPinX, OUTPUT);

pinMode(ledPinY, OUTPUT);

pinMode(ledPinZ, OUTPUT);

pinMode(powerPin, OUTPUT);

Serial.begin(9600);}

void loop() {

int valueX = 0;

int previousX = 0;

int valueY = 0;

int previousY = 0;

int valueZ = 0;

int previousZ = 0;

digitalWrite(powerPin, HIGH);

while(1){

valueX = analogRead(sensePinX);

valueY = analogRead(sensePinY);

valueZ = analogRead(sensePinZ);

int changeX = abs(valueX - previousX);

int changeY = abs(valueY - previousY);

int changeZ = abs(valueZ - previousZ);

previousX = valueX;

previousY = valueY;

previousZ = valueZ;

if(changeX>th){

digitalWrite(ledPinX, HIGH);

Serial.print("Vibration in x\_axis detected. Intensity: ");

Serial.print(changeX);

Serial.println();

delay(1000);}

else{

digitalWrite(ledPinX, LOW);}

if(changeY>th){

digitalWrite(ledPinY, HIGH);

Serial.print("Vibration in y\_axis detected. Intensity: ");

Serial.print(changeY);

Serial.println();

delay(1000);}

else{

digitalWrite(ledPinY, LOW);}

if(changeZ>th){

digitalWrite(ledPinZ, HIGH);

Serial.print("Vibration in z\_axis detected. Intensity: ");

Serial.print(changeZ);

Serial.println();

delay(1000);}

else{

digitalWrite(ledPinZ, LOW);}

Serial.print(changeX);

Serial.print(" ");

Serial.print(changeY);

Serial.print(" ");

Serial.print(changeZ);

Serial.println();}}

**INTERFACING OS WITH ARDUINO AND SENDING MAIL USING MAILGUN API SERVICE**

**Components Required:**

**Raspberry PI with Raspbian (Debian based operating System) installed:**

The **Raspberry Pi** is a series of [credit card](https://en.wikipedia.org/wiki/Credit_card)–sized [single-board computers](https://en.wikipedia.org/wiki/Single-board_computer) developed in the [UK](https://en.wikipedia.org/wiki/United_Kingdom) by the [Raspberry Pi Foundation](https://en.wikipedia.org/wiki/Raspberry_Pi_Foundation) with the intention of promoting the teaching of basic [computer science](https://en.wikipedia.org/wiki/Computer_science) in schools. ***To make this project cost effective and open sourced instead of using windows based operating system and computers, Raspberry PI B+ Model has been used***. This system can be implemented in any windows based operating system.

**OR, any general PC (Not cost effective).**

**Operating System Used**:

**Raspbian: *Raspbian is a free operating system based on Debian optimized for the Raspberry Pi hardware.*** An operating system is the set of basic programs and utilities that make your Raspberry Pi run. However, Raspbian provides more than a pure OS: it comes with over 35,000 packages, pre-compiled software bundled in a nice format for easy installation on your Raspberry Pi.

**OR, any windows operating system (Not supported by Raspberry PI).**

**Programing Language Used:**

**Python 2.7.10: Python** is a widely used [general-purpose](https://en.wikipedia.org/wiki/General-purpose_programming_language), [high-level programming language](https://en.wikipedia.org/wiki/High-level_programming_language). Its design philosophy emphasizes code [readability](https://en.wikipedia.org/wiki/Readability), and its syntax allows programmers to express concepts in fewer [lines of code](https://en.wikipedia.org/wiki/Lines_of_code) than would be possible in languages such as [C++](https://en.wikipedia.org/wiki/C%2B%2B) or [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [object oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [imperative](https://en.wikipedia.org/wiki/Imperative_programming) and [functional programming](https://en.wikipedia.org/wiki/Functional_programming) or [procedural](https://en.wikipedia.org/wiki/Procedural_programming) styles. It features a [dynamic type](https://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory management](https://en.wikipedia.org/wiki/Memory_management) and has a large and comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library).

***Python 2.7.10 is supported by most of the operating systems and pre-installed package for most of the Linux based operating System.***

**Python Modules Used:**

**PySerial:** PySerial is a library which provides support for serial connections ("RS-232") over a variety of different devices: old-style serial ports, Bluetooth dongles, infra-red ports, and so on. It also supports remote serial ports via RFC 2217 (since V2.5).

**Datetime Module:** The datetime module supplies classes for manipulating dates and times in both simple and complex ways. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation.

**Requests:** Requests is a Python HTTP library, released under the Apache2 License. The goal of the project is to make HTTP requests simpler and more human-friendly. The current version is 2.6.0.

**Mailgun: Transactional Email API Service for Developers**

A set of powerful APIs that enable you to send, receive and track email from application effortlessly for Python, Ruby, PHP, C#, Node.js or Java.

**Python Code for sending mail using Mailgun API Service:**

Import requests

def send\_simple\_message():

return requests.post(

"https://api.mailgun.net/v3/samples.mailgun.org/messages",

auth=("api", "key-3ax6xnjp29jd6fds4gc373sgvjxteol0"),

data={"from": "Excited User <excited@samples.mailgun.org>",

"to": ["devs@mailgun.net"],

"subject": "Hello",

"text": "Testing some Mailgun awesomeness!"})

**Python code used in the project to interface with Arduino and to send mail on Vibration Detection:**

import serial  
import datetime  
import requests  
def send\_simple\_message(text\_line):  
 return requests.post(  
 "https://api.mailgun.net/v3/sandbox39bc8ec3a6b14007ba79857e89f6857b.mailgun.org/messages",  
 auth=("api", "key-a66d8d62daeb52405993c2aea0c7813c"),  
 data={"from": "Mailgun Sandbox <postmaster@sandbox39bc8ec3a6b14007ba79857e89f6857b.mailgun.org>",  
 "to": "Diptanu Das <dd10@iitbbs.ac.in>",  
 "subject": "Hello Diptanu Das",  
 "text": text\_line})  
  
ser = serial.Serial('COM3', 9600)  
while True:  
 text = str(ser.readline())  
 *#if text[0] == "Vibration":* words = text.split(" ")  
 if words[0] == "Vibration":  
 dt = str(datetime.datetime.now())  
 text\_line = text[0:len(text)-2]+"\n"+dt[0:len(dt)-6]  
 print(text\_line)  
 send\_simple\_message(text\_line)

**FUTURE WORK:**

1. The security system can be further developed by integrating PIR sensor based system or infrared motion detection or light sensitive sensor with this vibration triggered system.
2. By interfacing with web-cams the system can take snap shots at the moment vibration is detected on the surface of protected body. By interfacing the described system with face detecting system (image processing) intruder alarm can be triggered if required. The snap shot can be mailed to the registered (authorized) user. Face recognition system can also be used in order to recognize the detected face is familiar or intruder.

**CONCLUSION:**

It can be concluded that the sole aim of carrying out the design, analysis and implementing of a simple and reliable vibration sensitive security system was achieved, in that the aim was to develop cheap, affordable, reliable and efficient security system, which was successfully realized at the end of the design process. One factor that accounts for the cheapness of the product was the proper choice of components used. The system was tested and found to be working to specifications and predictions. A cheap and reliable way of checking the activities of burglars and intruders has been successfully developed, which is the aim of the research.

**REFERENCES:**

Measuring Vibration - by Brüel & Kjær

A Simple and Relianble Touch Sensitive Security System - by Adamu Murtala Zungeru, Jonathan Gana Kolo and Ijarotimi Olumide

https://en.wikipedia.org/

https://wiki.python.org/

http://playground.arduino.cc/Interfacing/Python

http://www.mailgun.com/

https://www.raspberrypi.org/

https://www.raspbian.org/