L&T Project Report

EARTHQUAKE DETECTION & ALERT SYSTEM

Kedar Kosuri | Microcontrollers and its Applications | October 25, 2023

Project Title: Alert system for Earthquakes

ABSTRACT: An unexpected movement of the earth’s surface is called an earthquake. When two parts of the earth surface move suddenly in relation to each other along a fault line, due to tectonic forces, an earthquake occurs. An earthquake (also known as a tremor or temblor) is the result of a sudden release of energy in the earth's crust that creates seismic waves. Earthquakes are recorded with a seismometer, also known as a seismograph. The moment magnitude of an earthquake is conventionally reported, or the related and mostly obsolete Richter magnitude, with magnitude 3 or lower earthquakes being mostly imperceptible and magnitude 7 causing serious damage over large areas. Intensity of shaking is measured on the modified Marcella scale. An earthquake is an unpredictable natural disaster that causes damage to lives and property. It happens suddenly and we cannot stop it but we can be alerted from it. Now-a-days, there are many technologies which can be used to detect the small shakes and knocks, so that we can take precautions prior to some major vibrations in earth. This work uses an accelerometer to detect the pre-earthquake vibrations. Accelerometer is highly sensitive to shakes and vibrations along with all the three axes. The advantage of building an Arduino based Earthquake Detector using Accelerometer is to reduce its destructivelosses.

PROBLEM STATEMENT: Earthquakes pose a significant threat to life and property worldwide, and rapid detection and alert systems are crucial for mitigating their impact. In many regions, especially in developing countries, there is a lack of effective and affordable earthquake detection systems. The existing earthquake detection systems are often expensive, complex, and not easily accessible to the general population. Therefore, there is a pressing need to develop a cost-effective and user-friendly earthquake detection system using Arduino technology. In many regions around the world, the absence of robust and efficient earthquake detection and early warning systems hampers our ability to mitigate the potentially catastrophic consequences of seismic events. To address this critical issue, there is an urgent need for the development and implementation of an advanced earthquake detection system

SCOPE OF SOLUTION: With the extensive literature review above, the following are the objectives formed:

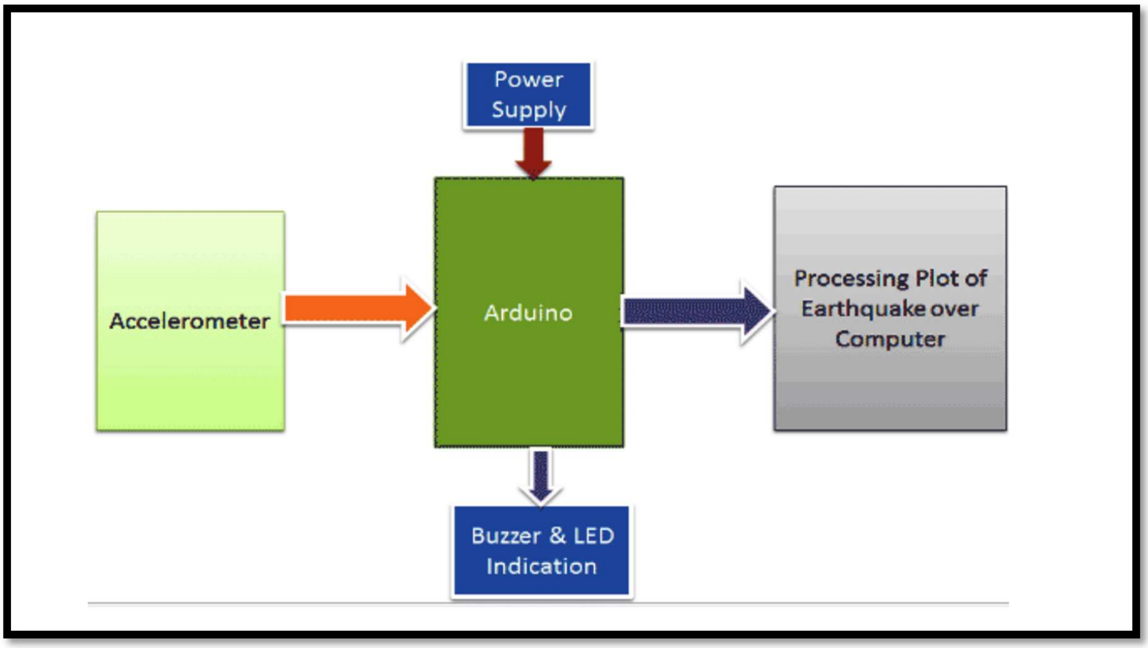
- Designing and development of earthquake alarm detection circuit based on electronic devices which will be helpful to save lives andproperty.
- Main objective is to detect S-waves (seismic waves) or we can say high frequency vibrations at the Arduino Uno input through accelerometer which will give us an indication about earthquake before it strikes at the core of thatplace
- Another thing is to make a device compact, cheap and error free because earthquake is a very big event in this error may cause very bigloss. To make a device error free we have to find a position where the device should bemounted on the building which will sense the best frequency vibrations, and the device should not damage until the building do not fall. Because the device has done its work it alarmed all the members of building to escape before the building fallover.

REQUIRED COMPONENTS TO DEVELOP SOLUTIONS: In Order to develop the required solution for the given problem statement, we must require the following components:

1) Arduino UNO Board
2) ADXL335 3 axis Accelerometer
3) 16x2 LCD
4) Buzzer
5) LED
6) Breadboard
7) 1.7k ohm Resistor (one)
8) Potentiometer (To control the brightness of the backlight in LCD)
9) Jumper wires

IDE used – Arduino IDE
Software – Windows OS 11
Hardware – Arduino Development Board, Power Cord for Arduino (To power and transfer codes)

BLOCK DIAGRAM:



BLOCK DIAGRAM DISCRIPTION:

Working of this earthquake detector is simple.
As we mentioned earlier that we have used Accelerometer for detecting earthquake vibrations along any of the three axes so that whenever vibrations occur accelerometer senses the vibrations and convert them into equivalent ADC value. Then these ADC values are read by Arduino and shown over the 16x2 LCD. First we need to calibrate the Accelerometer by taking the samples of surrounding vibrations whenever Arduinopowers up. Then we need to subtract those sample values from the actual readings to get the real readings.

This calibration is needed so that it will not show alerts with respect to its normal surrounding vibrations. After finding real readings, Arduino compares these values with predefined max and min values. If Arduino finds any changes values are more then or less then the predefined values of any axis in both direction (negative and positive) then Arduino trigger the buzzer and shows the status of alert over the 16x2 LCD and a LED also turned on as well. We can adjust the sensitivity of earthquake detector by changing the predefined values in Arduino code.

WORKING EXPLANATION OF THE PROJECT:

1. **Sensor Setup**:
   - Connect the ADXL335 accelerometer to the Arduino using jumper wires. The sensor has three output pins: X, Y, and Z. Connect these pins to three analog input pins on the Arduino.
   - Connect the sensor's Vcc (power) and GND (ground) pins to the appropriate pins on the Arduino.
2. **LCD Setup**:
   - Connect the 16x2 LCD to the Arduino. This typically involves connecting the LCD's Vcc, GND, SDA, and SCL pins to the corresponding pins on the Arduino. The specific connections may vary depending on the type of LCD you're using.
3. **Code Implementation**:

- Write an Arduino sketch (program) that reads data from the ADXL335 accelerometer using analogRead() for each of the X, Y, and Z axes. The accelerometer outputs analog voltage signals that vary with acceleration.
- Process the sensor data to detect abnormal vibrations, which can indicate seismic activity (earthquakes). You can calculate the magnitude of acceleration vector by combining data from all three axes. If this magnitude exceeds a predefined threshold, consider it as a vibration event.

Example code for reading accelerometer data and checking for vibrations:

```
const int xPin = A0; // Analog input pins for X, Y, and Z

const int yPin = A1;
const int zPin = A2;
const int threshold = 100; // Set a threshold for vibration detection

void setup() {
  // Initialize LCD, setup pins, etc.
}

void loop() {
  int xValue = analogRead(xPin);
  int yValue = analogRead(yPin);
  int zValue = analogRead(zPin);

  // Calculate magnitude of acceleration
  float acceleration = sqrt(sq(xValue) + sq(yValue) + sq(zValue));

  if (acceleration > threshold) {
    // Vibration detected, earthquake alert!
      // Display a message on the LCD screen.
```

```
}

// Update the LCD display with sensor data and alerts.}
```

4. **LCD Display**:
   - Modify your code to display the sensor data, earthquake alerts, and relevant information on the LCD. You can use the LiquidCrystal library for controlling the LCD display.

Example code for updating the LCD display:

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(0); // Initialize the LCD object

void setup() {
  lcd.begin(16, 2); // Set the LCD dimensions (16x2)
  // Other setup code
}

void loop() {
  // Read sensor data and detect earthquakes as explained earlier.

  // Update the LCD display with sensor data and earthquake alerts
  lcd.clear(); // Clear the display
  lcd.setCursor(0, 0); // Set cursor to the first row
  lcd.print("Earthquake Alert");
  lcd.setCursor(0, 1); // Set cursor to the second row
  lcd.print("Magnitude: X.X"); // Display earthquake magnitude
  delay(1000); // Delay to avoid rapid updates
}
```

5. **Testing**:
   - Power up your Arduino and run the code. The system will continuously monitor the accelerometer data, and when it detects significant vibrations (above the set threshold), it will trigger an earthquake alert displayed on the LCD.

This earthquake detection system, while relatively simple, can serve as a starting point for more sophisticated projects. To enhance its accuracy and usefulness, you can consider incorporating additional sensors, data logging, and communication capabilities for remote monitoring and alerting.

ARDUINO CODE:

Here is the complete working code of the project:-

```
#include<LiquidCrystal.h> // lcd Header
LiquidCrystal lcd(7,6,5,4,3,2); // pins for LCD Connection

#define buzzer 12 // buzzer pin
#define led 13 //led pin

#define x A0 // x_out pin of Accelerometer
#define y A1 // y_out pin of Accelerometer
#define z A2 // z_out pin of Accelerometer

/*variables*/
int xsample=0;
int ysample=0;
int zsample=0;
long start;
int buz=0;

/*Macros*/
#define samples 50
#define maxVal 20 // max change limit
#define minVal -20 // min change limit
#define buzTime 5000 // buzzer on time

void setup()
{
lcd.begin(16,2); //initializing lcd
Serial.begin(9600); // initializing serial
delay(1000);
lcd.print("EarthQuake ");
lcd.setCursor(0,1);
lcd.print("Detector ");
delay(2000);
lcd.clear();
lcd.print("Calibrating.....");
lcd.setCursor(0,1);
lcd.print("Please wait...");
pinMode(buzzer, OUTPUT);
pinMode(led, OUTPUT);
buz=0;
digitalWrite(buzzer, buz);
digitalWrite(led, buz);
for(int i=0;i<samples;i++) // taking samples for calibration
{
xsample+=analogRead(x);
ysample+=analogRead(y);
zsample+=analogRead(z);
}
```

```
xsample/=samples; // taking avg for x
ysample/=samples; // taking avg for y
zsample/=samples; // taking avg for z

delay(3000);
lcd.clear();
lcd.print("Calibrated");
delay(1000);
lcd.clear();
lcd.print("Device Ready");
delay(1000);
lcd.clear();
lcd.print(" X Y Z ");
}

void loop()
{
int value1=analogRead(x); // reading x out
int value2=analogRead(y); //reading y out
int value3=analogRead(z); //reading z out

int xValue=xsample-value1; // finding change in x
int yValue=ysample-value2; // finding change in y
int zValue=zsample-value3; // finding change in z

/*displying change in x,y and z axis values over lcd*/
lcd.setCursor(0,1);
lcd.print(xValue);
lcd.setCursor(6,1);
lcd.print(yValue);
lcd.setCursor(12,1);
lcd.print(zValue);
delay(100);

/* comparing change with predefined limits*/
if(xValue < minVal || xValue > maxVal || yValue < minVal || yValue > maxVal || zValue < minVal ||
zValue > maxVal)
{
if(buz == 0)
start=millis(); // timer start
buz=1; // buzzer / led flag activated
}

else if(buz == 1) // buzzer flag activated then alerting earthquake
{
lcd.setCursor(0,0);
lcd.print("Earthquake Alert ");
if(millis()>= start+buzTime)
buz=0;
```

```
}

else
{
lcd.clear();
lcd.print(" X Y Z ");
}

digitalWrite(buzzer, buz); // buzzer on and off command
digitalWrite(led, buz); // led on and off command

/*sending values to processing for plot over the graph*/
Serial.print("x=");
Serial.println(xValue);
Serial.print("y=");
Serial.println(yValue);
Serial.print("z=");
Serial.println(zValue);
Serial.println(" $");
}
```

PROCESSING CODE:

```
import processing.serial.*;
PFont f6,f8,f12,f10;
PFont f24;
Serial myPort; // The serial port
int xPos = 0; // horizontal position of the graph
float y1=0;
float y2=0;
float y3=0;

void setup ()
{
// set the window size: and Font size
f6 = createFont("Arial",6,true);
f8 = createFont("Arial",8,true);
f10 = createFont("Arial",10,true);
f12 = createFont("Arial",12,true);
f24 = createFont("Arial",24,true);
size(1200, 700);

// List all the available serial ports
println(Serial.list());
myPort = new Serial(this, "COM10", 9600);
println(myPort);
myPort.bufferUntil('\n');
background(80);
```

```
}

void draw ()
{
serial ();
}

void serial()
{
String inString = myPort.readStringUntil('$'); // reading incomming date from serial
if (inString != null)
{
// extracting all required values of all three axis:
int l1=inString.indexOf("x=")+2;
String temp1=inString.substring(l1,l1+3);
l1=inString.indexOf("y=")+2;
String temp2=inString.substring(l1,l1+3);
l1=inString.indexOf("z=")+2;
String temp3=inString.substring(l1,l1+3);

//mapping x, y and z value with graph dimensions
float inByte1 = float(temp1+(char)9);
inByte1 = map(inByte1, -80,80, 0, height-80);
float inByte2 = float(temp2+(char)9);
inByte2 = map(inByte2,-80,80, 0, height-80);
float inByte3 = float(temp3+(char)9);
inByte3 = map(inByte3,-80,80, 0, height-80);
float x=map(xPos,0,1120,40,width-40);

//ploting graph window, unit
strokeWeight(2);
stroke(175);
Line(0,0,0,100);
textFont(f24);
fill(0,00,255);
textAlign(RIGHT);
xmargin("EarthQuake Graph (SESMIOGRAPH)",200,100);

fill(100);
strokeWeight(100);
line(1050,80,1200,80);

strokeWeight(1);
textAlign(RIGHT);
fill(0,0,255);
String temp="X:"+temp1;
Text(temp,100,95);

fill(0,255,0);
temp="Y:"+temp2;
```

```
Text(temp,100,92);

fill(255,0,0);;
temp="Z:"+temp3;
Text(temp,100,89);


//ploting x y and z values over graph
strokeWeight(2);
int shift=40;

stroke(0,0,255);
if(y1 == 0)
y1=height-inByte1-shift;
line(x, y1, x+2, height-inByte1-shift) ;
y1=height-inByte1-shift;

stroke(0,255,0);
if(y2 == 0)
y2=height-inByte2-shift;
line(x, y2, x+2, height-inByte2-shift) ;
y2=height-inByte2-shift;

stroke(255,0,0);
if(y2 == 0)
y3=height-inByte3-shift;
line(x, y3, x+2, height-inByte3-shift) ;
y3=height-inByte3-shift;

xPos+=1;

if (x >= width-30) // go back to begining
{
xPos = 0;
background(80);
}
}
}

void Line(int x1, int y1, int x2, int y2)
{
float xx1=map(x1,0,100,40,width-40);
float xx2=map(x2,0,100,40,width-40);
float yy1=map(y1,0,100,height-40,40);
float yy2=map(y2,0,100,height-40,40);

line(xx1,yy1,xx2,yy2);
xx2=map(100,0,100,40,width-40);
yy2=map(0,0,100,height-40,40);
line(xx1,yy1,xx2,yy2);
```

```
strokeWeight(1);
for(int i=1;i<21;i++)
{
yy2=map(i*10,0,200,height-40,40);
yy1=yy2;
line(xx1,yy1,xx2,yy2);
}
yy2=map(100,0,100,height-40,40);
yy1=map(0,0,100,height-40,40);
for(int i=1;i<41;i++)
{
xx1=map(i*5,0,200,40,width-40);
xx2=map(i*5,0,200,40,width-40);
line(xx1,yy1,xx2,yy2);
}

textAlign(RIGHT); // 100 degree
// result+=yy1;
fill(255);
strokeWeight(1);
textFont(f12);
for(int i=-10;i<11;i++)
{
String result="";
result+=5*i;
ymargin(result, x1,y1);
y1+=5;
}

x1=0;
y1=0;
strokeWeight(1);
textFont(f10);
for(int i=0;i<41;i++)
{
String result="";
result+=28*3*i;
xmargin(result, x1,y1);
x1+=5;
}

textAlign(RIGHT);

textAlign(RIGHT);
}

void ymargin(String value, int x1, int y1)
{
```

```
float xx1=map(x1,0,100,40,width-40);
float yy1=map(y1,0,100,height-40,40);
text(value,xx1-5,yy1+5);
}

void xmargin(String value, int x1, int y1)
{

float xx1=map(x1,0,200,40,width-40);
float yy1=map(y1,0,100,height-25,25);
text(value,xx1+7,yy1);
}

void Text(String value, int x1, int y1)
{

float xx1=map(x1,0,100,40,width-40);
float yy1=map(y1,0,100,height-25,25);
text(value,xx1,yy1);
}
```

ADVANTAGES OF PROPOSED SCHEME:

- An earthquake detector can be useful home safety devices because it alerts people to earthquake before it happens.
- It is simple to interface and rugged in design.
- The given system is handy and portable, and thus can be carried from one place to another.
- The circuitry is not that complicated and thus can be easily troubleshot.
- The given system is sets off a powerful buzzer, and it is effective than other alarm system available in the market.

CONCLUSION: This project titled “Earthquake detection & alert system” is succesfully running and is implemented perfectly. Thus to sum up we have introduced this product with a view to reduce the destruction caused by earthquakeby alerting the people. It is economical and its price is quoted in such a way that it is affordable by every individual. We have presented a novel technique to solve the automatic detection and classification problem of earth tremor in single step by using Arduino based earthquake detecting device. In our system the majority of cases offer real practical benefits in the event of an earthquake to safeguard lives and resources. We can easily set up this system for household purposes as it consumes less power.

NOTE: **ME KEDAR KOSURI, COULD NOT UPLOAD THE GERBER FILE FOR THE PCB DESIGNING, SINCE I FACED DIFFICULTIES IN UNDERSTANDING THE PROCEDURE AND WAS NOT ABLE TO GENERATE THE FILE AND DESIGN THE PCB DUE TO LACK OF RESOURCES AND UNDERSTANDING. PLEASE FORGIVE ME, AND ITS MY SINCERE APOLOGIES FOR NOT BEING ABLE TO UPLOAD THE GERBER FILES.**

THANK YOU