**Digital Spirit Level using Arduino and ADXL 335**

**Category:** Arduino Project



**Name:** Sajid Shaikh

**Email-ID:** sajid6300shaikh@gmail.com

**Contact number:** 9987371251

**Profession:** Student

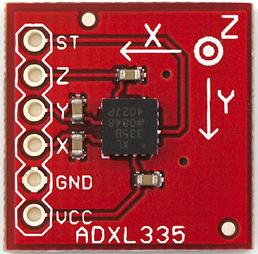
**City & Country:** Kalyan, India

**Description:**

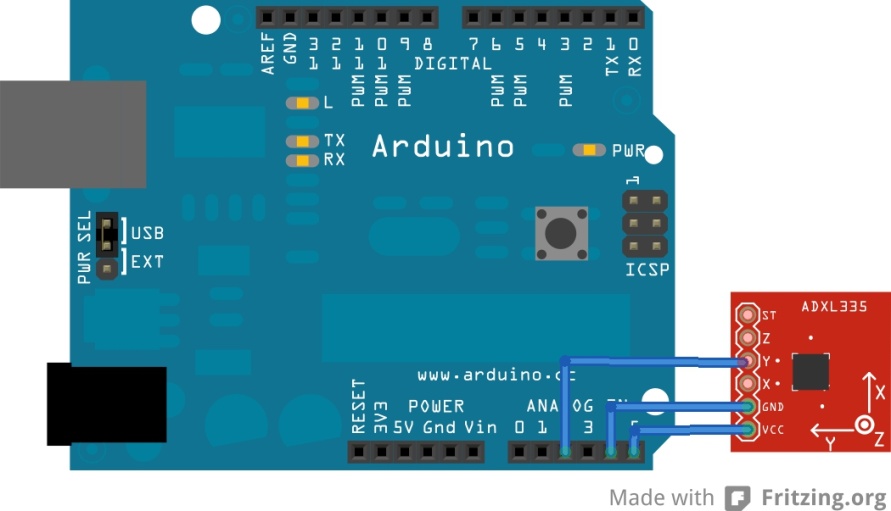
Hello Friends,  
Here is a very simple college project using Arduino and Accelerometer.  
Spirit Level (also called Level Bottle) is a tool which is used to see whether the surface is perfectly horizontal or not, this tool is used by plumbers and carpenters and some hobbyist to make perfectly level surface.

In this project I present a Digital Version of this Spirit level.

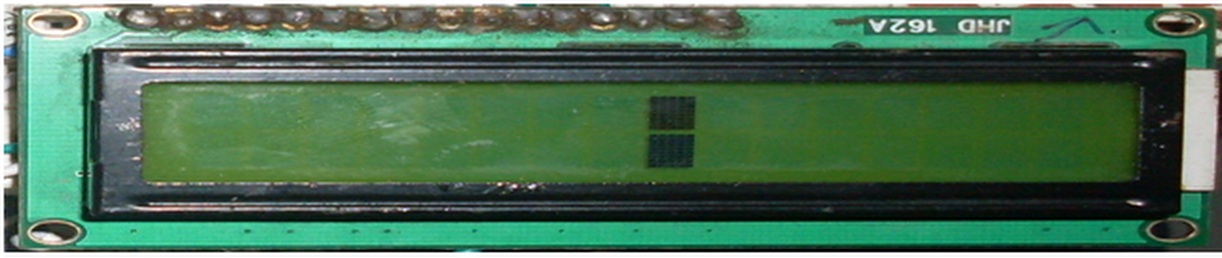
A typical analog one looks like this,

  
  
**Interface Devices:  
1. Sensor:**Now the most important part of this project is to measure the tilt of our board w.r.t to ground level. For this we use an Inertial Measurement Unit Sensor called Accelerometer.   
The same sensor is used in our smart phone to detect the orientation of our phone, that’s why when we tilt the phone the screen changes from portrait to landscape.  
Now there are many types of accelerometer sensor whose working principles, Axis & Interfacing methods are different (like I2C, SPI, Analog).  
And obviously the simplest of all sensors is the one which gives out analog values and we can feed these analog values to the ADC and get the readings in our microcontroller.  
In this project I have used ADXl 335 Accelerometer sensor.  


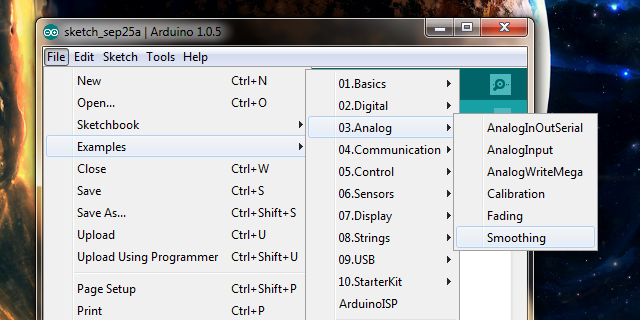
ADXL 335 is a 3-DOF(Degrees of Freedom) Accelerometer.  
3-DOF:Because it gives measure of tilt along 3 axis.

In this project we are making only 1 axis tilt sensor, so we require connecting only Y pin to Arduino’s ADC input.  
Here’s the connection.  
***Note:*** *Vcc and gnd are connected to Analog pin 5 &4 respectively, and in the program we will set pin 5 & 4 as High and Low which will serve as Vcc and gnd.*

**2. Display:**This project uses a 16x2 character LCD to show a dramatic air bubble as in Air Bubble Spirit Level available in hardware stores



**Smoothing Example of Arduino**  
Now directly using sensor data is not recommended because it gives very noisy output.  
So here I found something very interesting in Arduino IDE’s inbuilt examples.  
It is called as smoothing which can be opened from here



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**It is a very interesting example because it gives a very clean data out of your noisy sensor (like Accelerometer, its too much noisy) by filtering out high frequency noise.**

**After careful study of the code I found out that it actually calculates the moving average from sensor data which is a type of *Finite Impulse Response Low Pass Filter*.**

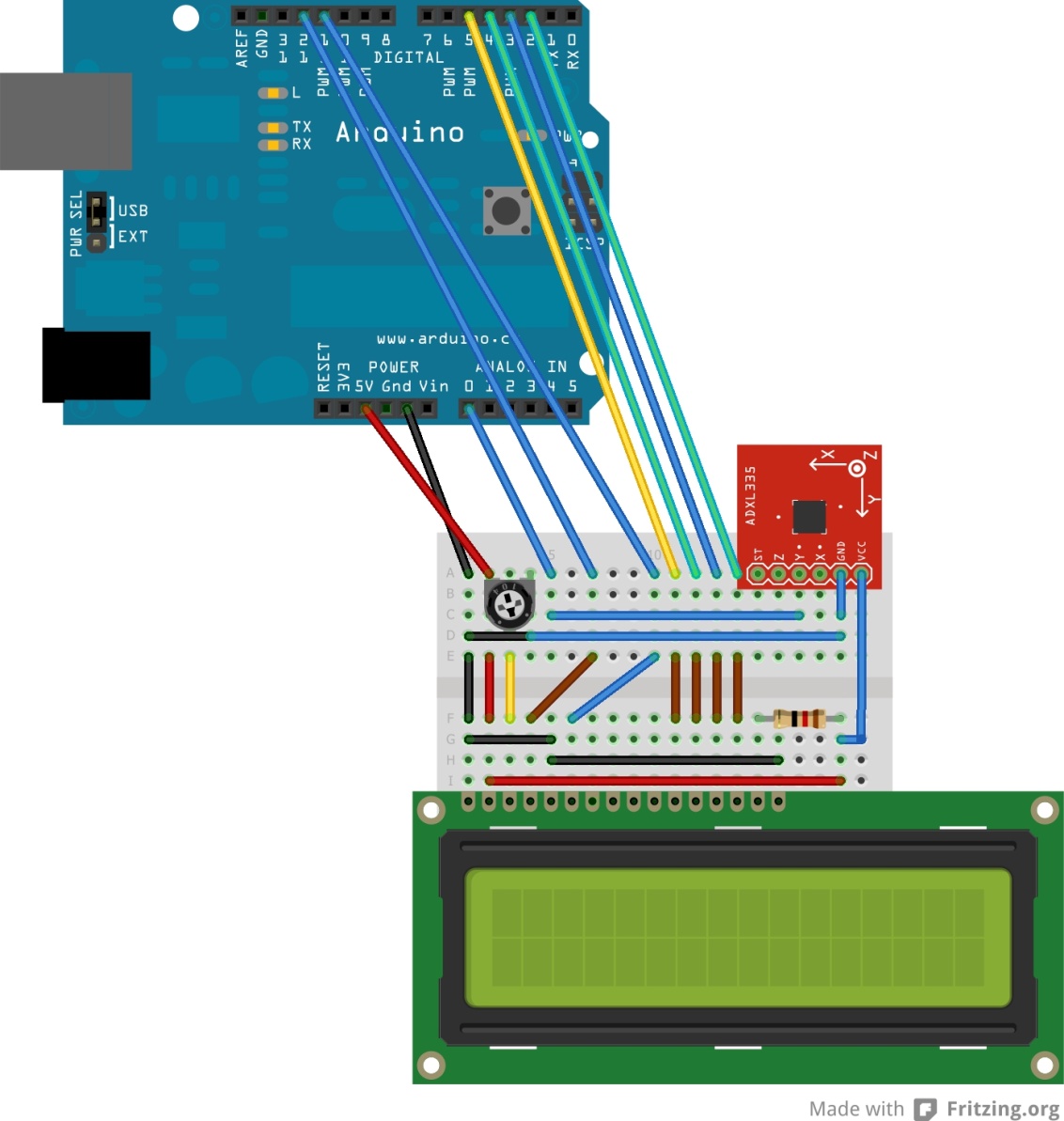
**It actually samples some predefined number of sensor readings and then puts it in series in a subset. Then the subset is modified by "shifting forward"; that is, excluding the first number of the series and including the next number following the original subset in the series. This creates a new subset of numbers, which is averaged. This process is repeated over the entire data series.  
  
You can change the no of elements you want in the series by changing the number in this line of smoothing code.**

*const int numReadings = 10;*

Too high number will result in slow response.

So I tinkered with the smoothing example and added code for LCD and that's all.  
So all we do is set cursor on our LCD according to the sensor data and print a special character (like air bubble).

**Circuit**

****

Working Video

<http://www.youtube.com/watch?v=D2um2-suLt4>

#### Source code

#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //pin definition for LCD

const int numReadings = 10; /\* Define the number of samples to keep track of. The higher the number, the more the readings will be smoothed, but the slower the output will respond to the input. Using a constant rather than a normal variable let use this value to determine the size of the readings array.\*/

int buffer[20]; //Buffer needed to store data packet for transmission

int data1;

int readings[numReadings]; // the readings from the analog input

int index = 0; // the index of the current reading

int total = 0; // the running total

int average = 0; // the average

int inputPin = A2;

byte heart[8] = { //Custom LCD character

0b00000,

0b01010,

0b11111,

0b11111,

0b11111,

0b01110,

0b00100,

0b00000

};

byte smiley[8] = {

0b00000,

0b00000,

0b01010,

0b00000,

0b00000,

0b10001,

0b01110,

0b00000

};

byte block[8] = {

0b11111,

0b11111,

0b11111,

0b11111,

0b11111,

0b11111,

0b11111,

0b11111

};

void setup()

{

lcd.createChar(1, heart);

// create a new character

lcd.createChar(2, smiley);

// create a new character

lcd.createChar(3, block);

pinMode(A4, OUTPUT);

digitalWrite(A4, LOW); //Adxl’s Gnd pin is connected to A4

pinMode(A5, OUTPUT);

digitalWrite(A5, HIGH); //Adxl’s Vcc pin is connected to A5

lcd.begin(16, 2);

lcd.setCursor(0,0);

lcd.print(" Digital Spirit");

lcd.setCursor(0,1);

lcd.print(" Level Project");

lcd.setCursor(15,1);

lcd.write(1);

delay(2000);

lcd.clear();

lcd.setCursor(0,0);

lcd.print(" Made By");

lcd.setCursor(14,0);

lcd.write(2);

lcd.setCursor(0,1);

lcd.print("\* SAJID SHAIKH \*");

delay(3000);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Using Arduino &");

lcd.setCursor(0,1);

lcd.print("Accelerometer");

delay(3000);

lcd.setCursor(13,1);

lcd.print(".");

delay(1000);

lcd.setCursor(14,1);

lcd.print(".");

delay(1000);

lcd.setCursor(15,1);

lcd.print(".");

delay(2000);

lcd.clear();

digitalWrite(13,LOW);

lcd.setCursor(0,0);

lcd.print("Let's Begin!!!");

delay(1000);

lcd.clear();

// initialize serial communication with computer:

Serial.begin(9600);

// initialize all the readings to 0:

for (int thisReading = 0; thisReading < numReadings; thisReading++)

readings[thisReading] = 0;

}

void loop() {

int lcd\_Cursor\_Position = 0;

lcd.clear();

// subtract the last reading:

total= total - readings[index];

// read from the sensor:

readings[index] = analogRead(inputPin);

// add the reading to the total:

total= total + readings[index];

// advance to the next position in the array:

index = index + 1;

// if we're at the end of the array...

if (index >= numReadings)

// ...wrap around to the beginning:

index = 0;

// calculate the average:

average = total / numReadings;

/\* since I am using 16 x 2 LCD, I am mapping the sensor values to 16 distinct values\*/

int avg=map (average, 280, 420, 0,15 );

data1 = avg;

plot(data1);

lcd\_Cursor\_Position =15- avg; // calculation to position the lcd cursor

lcd.setCursor((lcd\_Cursor\_Position), 1);

lcd.write(3); /\*this shows character on lcd which moves like bubble as in spirit level.\*/

lcd.setCursor((lcd\_Cursor\_Position), 0);

lcd.write(3);

delay(20);

}

/\*This function plots the sensor data so It can be seen as a waveform on PC using Simplot tool.\*/

//It can be neglected

void plot(int data1)

{

int pktSize;

buffer[0] = 0xCDAB; //SimPlot packet header. Indicates start of data packet

buffer[1] = 1\*sizeof(int); //Size of data in bytes. Does not include the header and size fields

buffer[2] = data1;

pktSize = 2 + 2 + (1\*sizeof(int)); //Header bytes + size field bytes + data

//IMPORTANT: Change to serial port that is connected to PC

Serial.write((uint8\_t \* )buffer, pktSize);

}