Course Name: Peripherals and Interfacing Laboratory

Course No: CSE 3204

Report on '**Arduino Air Drums**'

By:

Jahirul Islam Sourov (1307035)

&

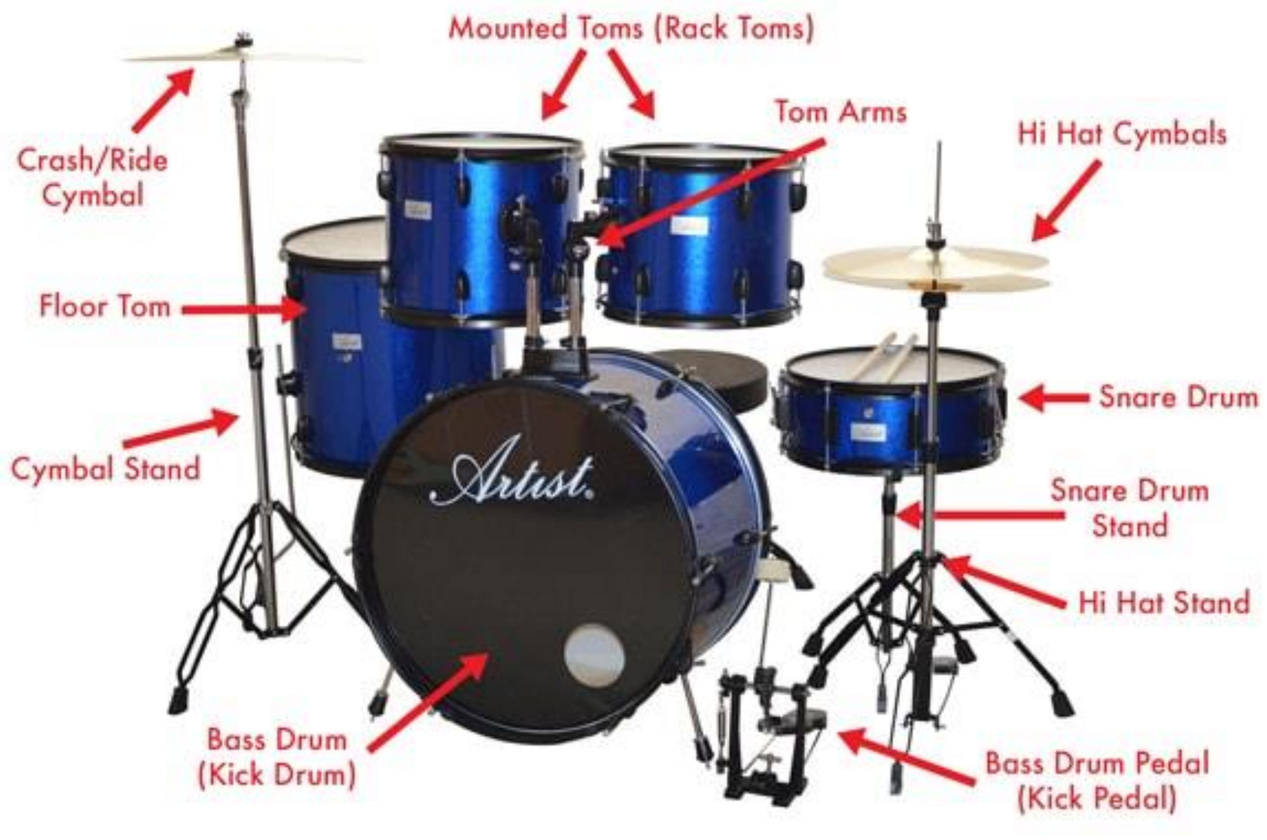
Tamzid Rahman Oronno (1307032)

Contents

<u>Introduction</u>

A Drums kit is one of the most essential part of modern music. Rock, Metal, Country music, Pop, Blues, Jazz – almost every genres of music. Air drums refers to playing drums in the air with the drumsticks in absence of drums kits. We made this Arduino air drums using accelerometer sensors which will generate specific drum kit's sound if drumstick or sandals are triggered.

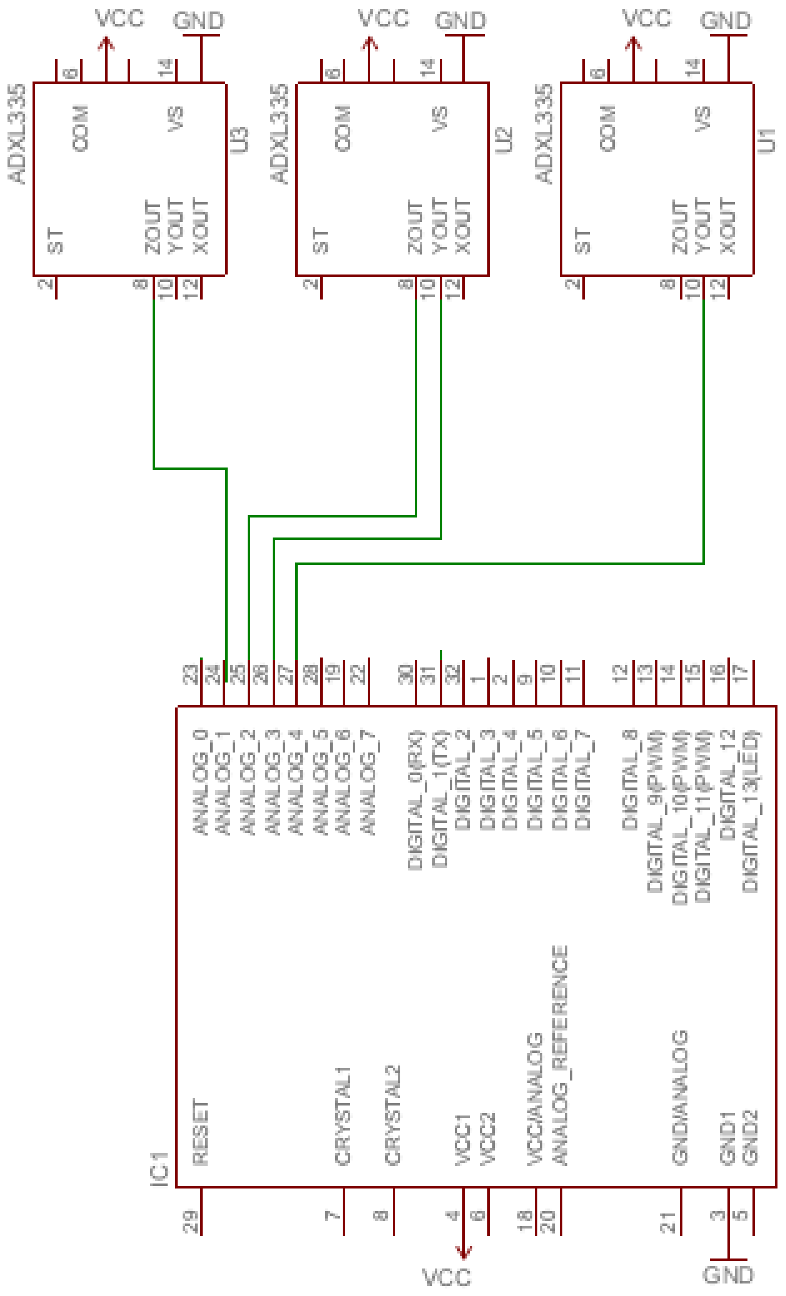
<u>Overview of Drums Set</u>



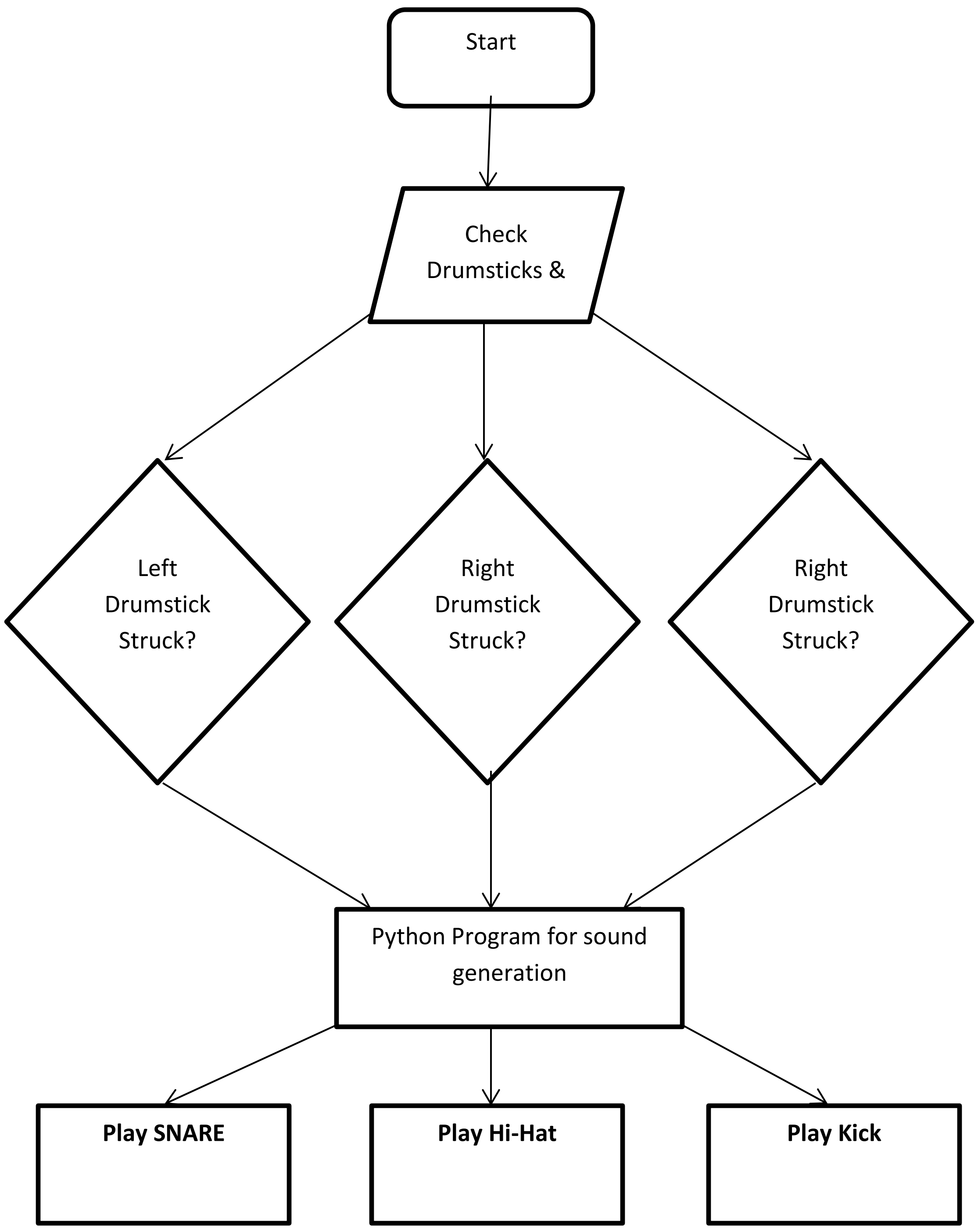
Components

1. Arduino Uno Microcontroller	
2. ADXL 335 accelerometers	(3)
3. USB male connectors	(4)
4. USB female connectors	(4)
5. Breadboard	(1)
6. Drumsticks	(2)
7. Sandals	(2)
8. Male to Male Jumper wires	(6)
9. Long Wires	

Circuit Diagram



Flow Chart


Start
Check Drumsticks &
Left Drumstick Struck?
Right Drumstick Struck?
Right Drumstick Struck?
Python Program for sound generation
Play SNARE
Play Hi-Hat
Play Kick

Description

We interfaced the accelerometers in sandal & two sticks. If we swing them in a specific manner and the velocity exceeds a threshold value, it prints corresponding character in the serial monitor of Arduino.

The python program reads the serial monitor from Arduino.
The program plays snare.wav/kick.wav/hi-hat.wav sounds according to the serial monitor's output using pygame library.

Codes

Arduino code (.ino):

```
void noteOn(byte channel, byte note, byte velocity)
{
   midiMsg(channel+0x90, note, velocity);
}

void midiMsg(byte cmd, byte data1, byte data2)
{
   Serial.write(cmd);
   Serial.write(data1);
   Serial.write(data2);
}

const int numSnareReadings = 10;
const int numHighHatReadings = 2;
const int numKickReadings = 5;
const int numCrashReadings = 10;

int snareReadings[numSnareReadings];
int snareIndex = 0;
int snareTotal = 0;
int snareAverage = 0;
int currentSnareState;
int switchSnareState = 0;

int highHatReadings[numHighHatReadings];
int highHatIndex = 0;
int highHatTotal = 0;
int highHatAverage = 0;
int currentHighHatState;
int switchHighHatState = 0;

int crashReadings[numCrashReadings];
int crashIndex = 0;
int crashTotal = 0;
int crashAverage = 0;
int currentCrashState;
int switchCrashState = 0;

int kickReadings[numKickReadings];
int kickIndex = 0;
int kickTotal = 0;
int kickAverage = 0;
int currentKickState;
int switchKickState = 0;

long previousMillis1 = 0;
long previousMillis2 = 0;
long previousMillis3 = 0;
long previousMillis4 = 0;

long interval1 = 100;
long interval2 = 100;
long interval3 = 100;
long interval4 = 100;

void setup()
{
  Serial.begin(9600);
  for (int thisReading = 0; thisReading < numSnareReadings; thisReading++)
    snareReadings[thisReading] = 0;

  for (int thisReading = 0; thisReading < numHighHatReadings; thisReading++)
    highHatReadings[thisReading] = 0;

  for (int thisReading = 0; thisReading < numKickReadings; thisReading++)
    kickReadings[thisReading] = 0;

  for (int thisReading = 0; thisReading < numKickReadings; thisReading++)
    crashReadings[thisReading] = 0;
```

```
}

void loop()
{
   unsigned long currentMillis = millis();

   snareAverage = snareTotal / numSnareReadings;
   snareTotal = snareTotal - snareReadings[snareIndex];
   snareReadings[snareIndex] = analogRead(A0);
   snareTotal = snareTotal + snareReadings[snareIndex];
   snareIndex = snareIndex + 1;

   if(snareIndex >= numSnareReadings)
      snareIndex = 0;

   snareAverage = snareTotal / numSnareReadings;
   currentSnareState = snareAverage;
   if( currentSnareState > 575 && (currentMillis - previousMillis1) > interval1)
   {
      previousMillis1 = currentMillis;
          //noteOn(9, 38, 120);
      Serial.write('s');
   }


   highHatAverage = highHatTotal / numHighHatReadings;
   highHatTotal = highHatTotal - highHatReadings[highHatIndex];
   highHatReadings[highHatIndex] = analogRead(A1);
   highHatTotal = highHatTotal + highHatReadings[highHatIndex];
   highHatIndex = highHatIndex + 1;

   if(highHatIndex >= numHighHatReadings)
     highHatIndex = 0;

   highHatAverage = highHatTotal / numHighHatReadings;

   currentHighHatState = highHatAverage;
   if( currentHighHatState > 550 && (currentMillis - previousMillis2) > interval2 )
   {
      previousMillis2 = currentMillis;
          //noteOn(9, 46, 120);
      Serial.write('k');//pevious h
   }

   kickAverage = kickTotal / numKickReadings;
   kickTotal = kickTotal - kickReadings[kickIndex];
   kickReadings[kickIndex] = analogRead(A2);
   kickTotal = kickTotal + kickReadings[kickIndex];
```

```
    kickIndex = kickIndex + 1;

    if(kickIndex >= numKickReadings)
      kickIndex = 0;

    kickAverage = kickTotal / numKickReadings;

    currentKickState = kickAverage;
    if( currentKickState > 360 && (currentMillis - previousMillis3) > interval3 )
    {
        previousMillis3 = currentMillis;
            //noteOn(9, 35, 120);
        Serial.write('h');//prev k
    }
}
```

Python code (.py):

```
import pygame
from pygame.locals import *
import pygame.mixer
import serial
import time

portStr = 'COM3'
arduino = serial.Serial(portStr, 9600)

pygame.display.set_mode((120, 120), DOUBLEBUF | HWSURFACE)
pygame.init()

pygame.mixer.init()

snare = pygame.mixer.Sound('snare.wav')
crash = pygame.mixer.Sound('crash.wav')
kick = pygame.mixer.Sound('kick.wav')

while True:
    mainval = arduino.read()
    arduino_byte = chr(mainval[0])
    if arduino_byte == 's':
        snare.play()
    elif arduino_byte == 'h':
        crash.play()
    elif arduino_byte == 'k':
        kick.play()
```

Advantages

- It’s very easy and handy to use.
- Drummers can practice their lessons without real drums.
- No third party DAW applications were used, so its free of complications.
- No external power needed. Sounds are played directly from PC
- Drums kit can be changed easily. Cymbals, Crashes, Toms, Cowbell can be played too.

Limitations

- Each sandal or drumstick allows to play one type of kit only.
- Cables connected to drumsticks make them heavy or sluggish.
- Ride or roll sounds isn’t generated.

Future Plans

- Using gyroscope instead of accelerometer to play different kits in different positions with the components.
- Making it wireless.
- Make an android app to connect via Wi-fi and play drums on phone.