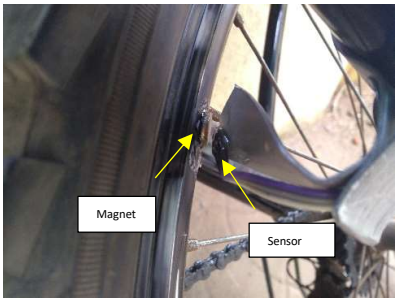
COMPONENTS

1) Sensor-

The input is captured with a magnetic and an hall effect sensor. The magnet is mounted on the wheel and the every time the magnet crosses the sensor, Arduino recognises the event and makes a note of time at which the event took place. Thus acquiring 3 time events, the angular velocity and acceleration of the wheel can be calculated.

(The magnet on the rim)



(Magnet near the hall sensor)

The sensor used is a A3144 hall effect sensor. This sensor pulls its output low when ever it encounters a strong enough magnetic field. However it does require a 10k ohm pullup resistor. However this resistor is replaced with the integrated 20k ohm resistor of the microcontroller. Also the polarity of the magnetic field matters and the sensor won't work if the magnet is placed in the wrong orientation.

(Hall Effect Sensor A3144)

2) Microcontroller-

The project is based around the Arduino Nano microcontroller with Atmel Atmega328P as the brains of the project.

(The microcontroller and the power bank. The Arduino receives power from the power bank through a USB to Mini USB cable)

The iBall PC-2204 power bank as the power source with a battery capacity of 2200mAh and a maximum current draw of 1A.

3) The display-

The display is an a generic Adafruit 128x32 monochrome display. The display uses I2C as communication protocol to communicate with the microcontroller.

{Display in three modes:

a) Just after booting the system, START in the lower right corner indicating the system is ready.

b) After a ride where the display show the distance travelled (accurate to ≈ 2 meters)

c) The speed at which the cycle is travelling (in km/hr) }

4) Power-

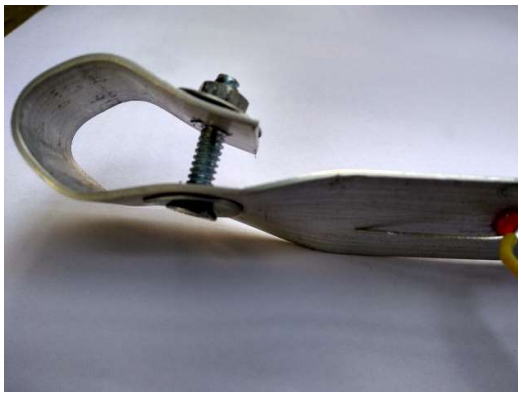
The system does not have a integrated power supply as the charging with power outlet would not have been possible and adding a dynamo was out of the scope. Hence a simple commercial power bank was chosen as it is easy to find one, they have descent capacity, can be charged with normal charger, are portable and the most importantly, they have a charge controller as well as the output is regulated and protected. Hence the battery is protected against any circuit flaws which my lead to short circuit and thus the chances of fire or explosions in battery are zero.

The power source used for the project is a iBall Power bank (Model: PC-2204). The model was chosen as it had geometry for which a holder could be easily be made with aluminum strips.

5) Fixtures-

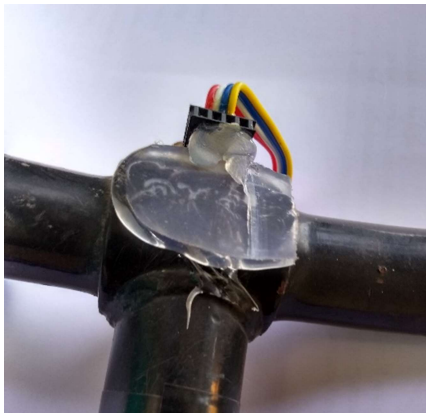
The battery, the hall effect sensor and the tail lights were secured on the cycle body with fixtures that were made by bending and twisting aluminum strip having a thickness of 1.34mm and a width of 18.59mm. The strips were bent in a semicircular shape. Thus, nuts and bolts were passed through the strip and secured on the cycle chassis.

Nut and bolt method was chosen as I did not want to make any permanent change to the cycle and with nuts and bolts, the system can be easily be removed or modified when needed.



Tail-light and the fixture on the frame.

The fixture used for mounting of tail lights along with the nut and the flat headed bolt.



However, for mounting the display for speed and distance, the female headers acted as the support for display.

As these were too small and delicate for screws and nuts, they were simply pasted with some hot glue.

CODE

This program was written on Arduino IDE

```
#include <Adafruit_SSD1306.h>
#include <Wire.h>
```

Arduino libraries for the OLED display

```
#define pi 3.14
#define brakelight 8
#define diameter .66
#define displaychangeafter 5000

#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);

const int chipSelect= 9;
```

Assigining values for constants like π, the diameter of the wheel, the digital pin connected to the brake light and the time after which the display is supposed to show the distance instead of the speed once the cycle has stopped.

```
int n=0;
int count=0;
double instant=0;
double previnstant=0;
double prevprevinstant=0;
int wheelcount=0;
float velocity=0;
float accleration=0;
float distance=0;
int temp;
bool unit1=0;

bool unit3=0;
int unitmillis1=0;
int unitmillis2=0;
int unitmillis3=0;
```

Declaring global variables

```
void setup()
{
 pinMode(2,INPUT_PULLUP);
 pinMode(brakelight,OUTPUT);
 attachInterrupt(digitalPinToInterrupt(2),cross,FALLING);

 display.begin(SSD1306_SWITCHCAPVCC , 0x3C);
 display.setTextColor(WHITE);
 display.setTextSize(3);

}

void loop()
{

 if(0>accleration)
 {
  digitalWrite(brakelight,HIGH);
 }

 if(0<accleration)
 {
  digitalWrite(brakelight,LOW);
 }
 {
  if(unit1!=1&&unit2!=1&&unit3!=1)
  {
    display.clearDisplay();
    display.setCursor(0,0);
    display.print(velocity*18/5);
    display.setTextSize(2);
    display.setCursor(67,18);
    display.print("km/hr");
    display.display();
    display.setTextSize(3);
```

Setting up the display(font size and colour) the brakelight pin and the hall effect sensor

Checks if the cycle is speeding up or slowing down.

If the cycle is decelerating the brake lights will turn on irrespective if the brakes or pressed or not.

Part of code responsible for displaying the speed of the cycle.

The system calculates the speed in m/s.

So a constant of 18/5 has to be multiplied to get the speed in km/hr.

```
  display.display();
 }
 else
 {
  display.clearDisplay();
  display.setCursor(0,0);
  display.print(distance);
  display.setTextSize(1);
  display.setCursor(90,25);
  if(distance==0){
   display.print("START");
  }
  else{
   display.print("meters");
  }
  display.display();
  display.setTextSize(3);
 }
}
```

This part of the code is responsible for displaying the distance the cycle has travelled in the current session.

The distance is only displayed once the cycle has stopped and might take up to 5 seconds after the cycle has stopped

```
if(unitmillis1!=millis()/displaychangeafter+1)
{
 unitmillis1=millis()/displaychangeafter+1;
 unit1=1;
}
if(unitmillis2!=millis()/displaychangeafter+2)
{
 unitmillis2=millis()/displaychangeafter+2;
 unit2=1;
}
if(unitmillis3!=millis()/displaychangeafter)
{
 unitmillis3=millis()/displaychangeafter;
 unit3=1;
}
```

The part responsible for counting 5 seconds once the cycle has stopped.

```
}

void cross()
{

 prevprevinstant=previnstant;
 previnstant=instant;
 instant=micros()/1000.0;
 wheelcount++;


 if(wheelcount>2)
 {
  distance=diameter*wheelcount*pi;
  velocity=diameter/(instant-previnstant)*pi;
  accleration=2*diameter*pi*(1/(instant-previnstant)-1/(previnstant-prevprevinstant))/(instant-prevprevinstant);
  velocity=velocity*1000;
  accleration=accleration*100000000;

  unit3=0;
  unit2=0;
  unit1=0;
    }
}
```

Records the time at which the magnet crosses the hall effect sensor.

Calculates the distance, speed and the acceleration based on the data obtained above.

The part which says 5 seconds have not yet passed and hence asks the Arduino to display speed and not the distace