

**TABLE OF CONTENTS:**

1. Synopsis/Abstract/Summary………………………………1
2. List of Illustrations………………………………………………2
3. Introduction……………….………………………………………5
4. Main text……………………………………………………………6

4A. Findings……………………………………………………………...9

4B. Future Scope/Future works……………………………….9

4C. Conclusions………………………………………………………...9

1. Attachments……………………………………………………….10
2. Glossary………………………………………………………………11



**SYNOPSIS:**

The common problem faced in any house is the cleaning of dust present on top of the fan blades. Due to this dust a wheeze I created which results in most of the respiratory diseases like asthma, dust allergies etc. Timely cleaning of fans requires manpower and are time consuming.

There are many products in market which are used for fan cleaning like the Fullhouz fan cleaner stick and dust blowers which require man power to operate. Reputed fan companies like Crompton have launched new fans with a tilted blade which are dust repellent and repel about 50% of germs.

Our project works under three phases namely Set, Clean and Retract. The project is affixed to the top of the fan and is controlled using a Bluetooth based app. The basis fan cleaning equipment like cleaner stick prize ranges from 300-800 INR and the blower cost us 800 INR. The dust repellent fans prize ranges from 2500-4000 INR based on the size of the fan. Our project addresses these problems in an effective and cost-efficient manner and is fully automated.

**ABSTRACT:**

We have come up with a simple and cost-effective solution for solving the problem of clean the dust off the top of the fan blades. The prototype is affixed to the fan and is operated using a Bluetooth based android application. The cleaner works under three phases namely: -

1. Set
2. Clean
3. Retract

When ‘Set’ option is selected the cleaning set is rotated so as to set the cleaner parallel to the blades. The blades position is detected using an ultrasonic sensor and the cleaner’s rotation is stopped after the blade is detected.

When the ‘Clean’ option is selected the cleaning, mechanism is activated and the dust from top of the fan blades is cleaned by a duster which moved to and fro using a motor and a screw affixed to the motor.

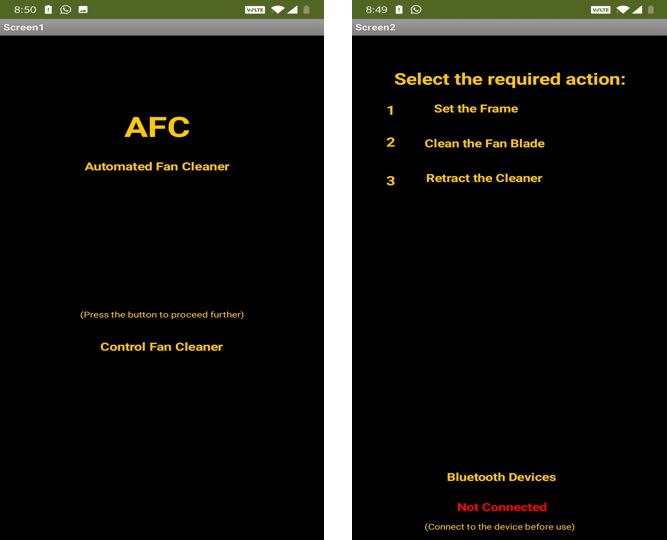
When the ‘Retract’ option is selected the duster used for cleaning the blades is folded using a servo motor so as to not obstruct the fan blades movement when the fan is switched on.



**List of Illustrations:**

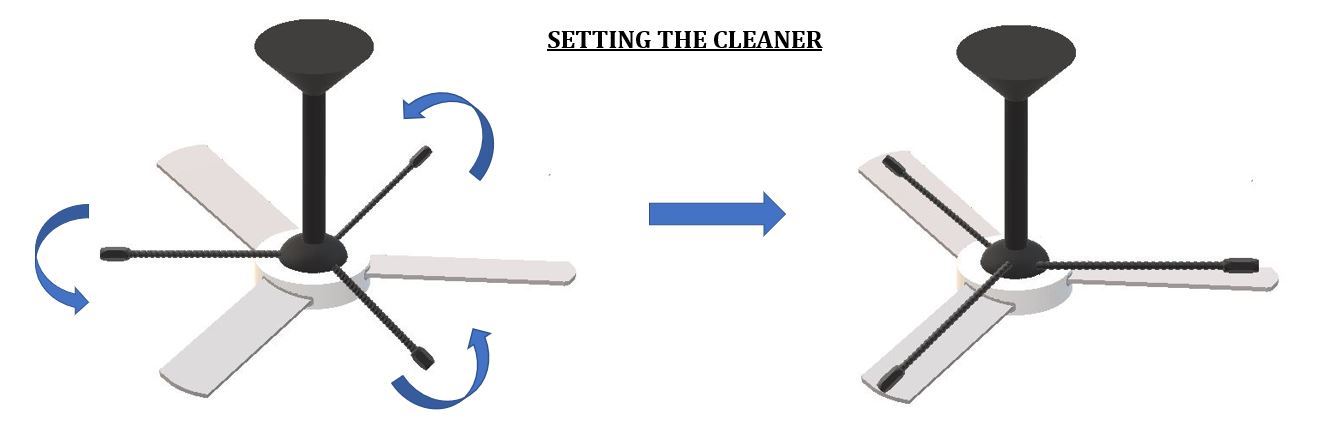
1. **Options Selection:**

The required working of the cleaner system is selected by the user through an app.



1. **Setting the Fan Blades:**

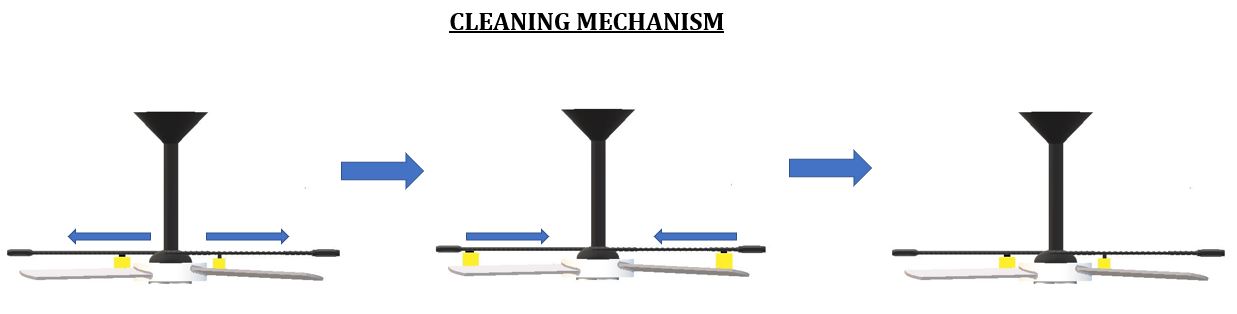
When the set option is selected the setup is rotated about the rod of the fan and is set to the fan blades position by detecting the fan blade using an ultrasonic sensor.When the cleaning option is selected the rod is rotated moving the duster front and back which in turn cleans the top of the blade.





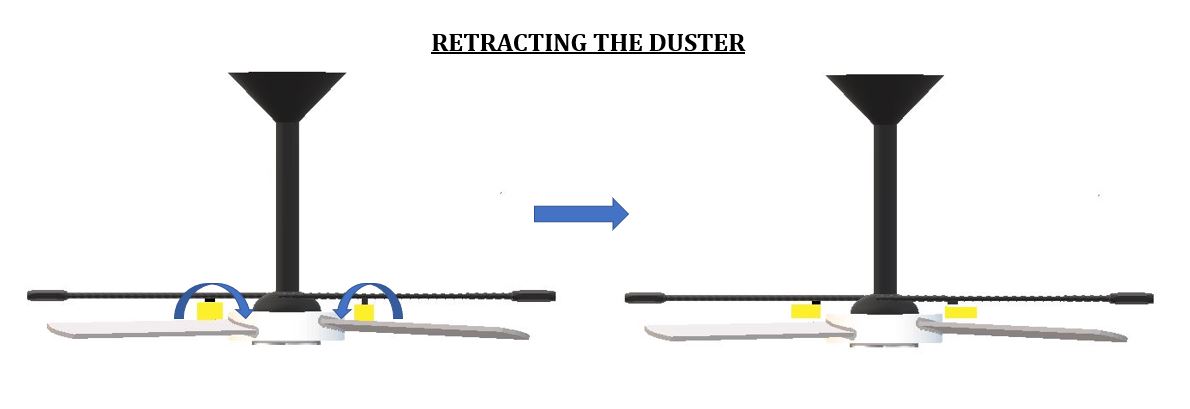
1. **Cleaning the Blades:**

When the cleaning option is selected the rod is rotated moving the duster front and back which in turn cleans the top of the blade.



1. **Retracting the Duster:**

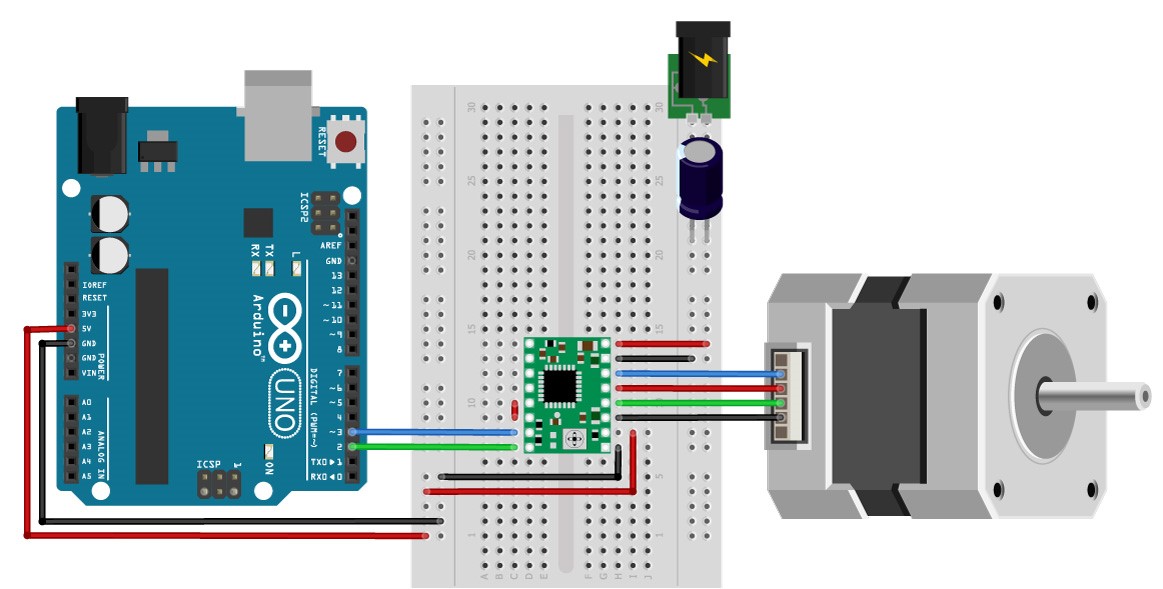
When the retract option is selected the duster is folded using a servo motor so as to not obstruct the flow of fan blades while the fan is running.



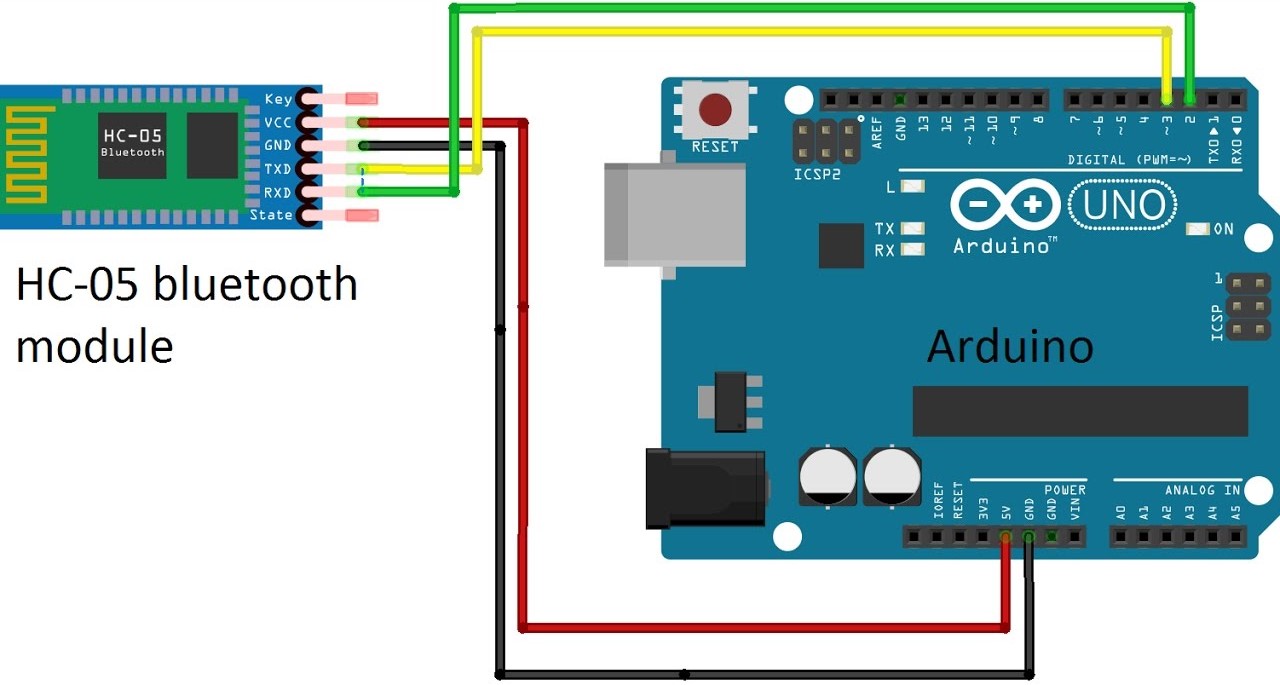


1. **Wiring Setup of the Prototype:**

**1. Wiring of Arduino Uno with Nema 17 using A4988:**

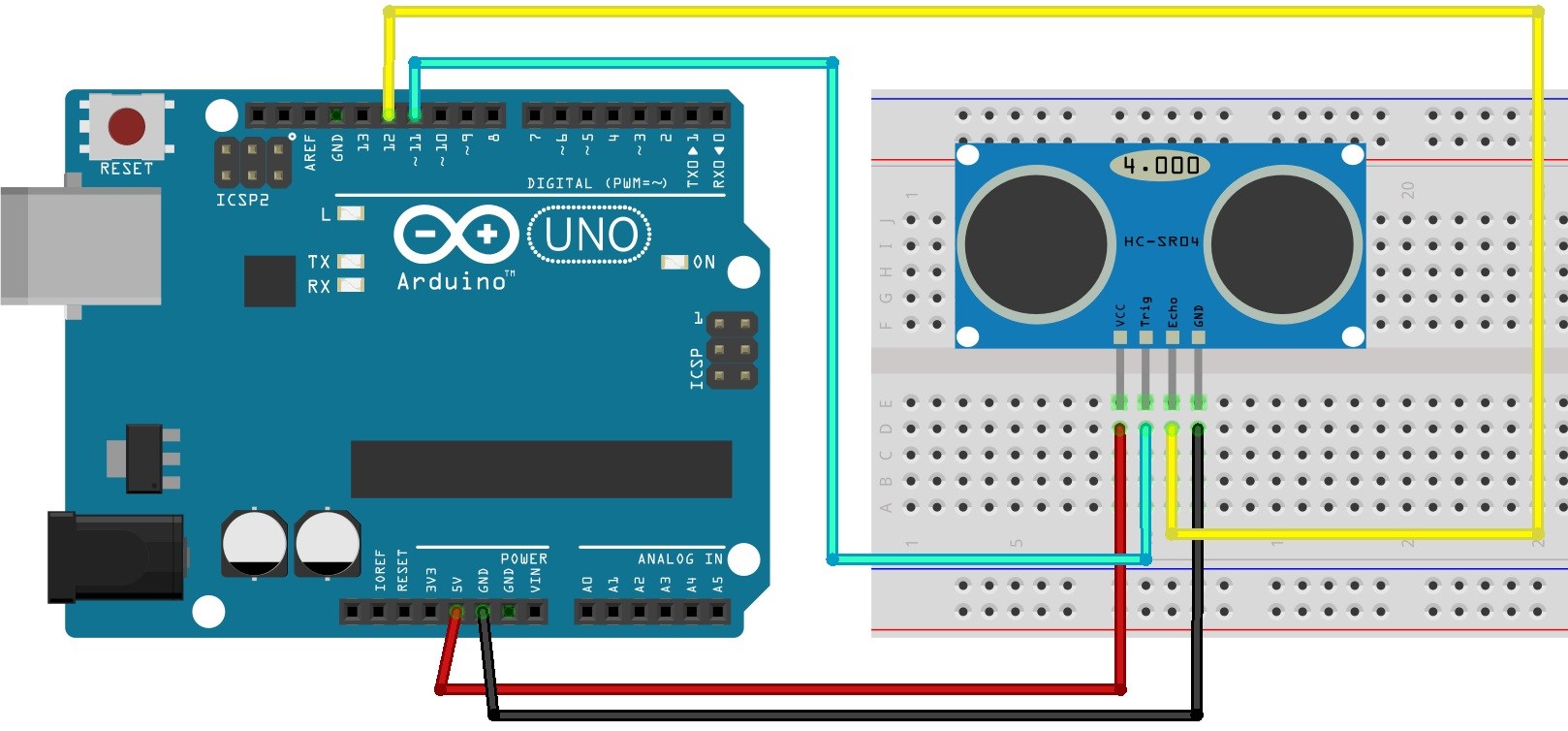
****

**2. Wiring of Arduino with HC-05:**

****



**3. Wiring of Arduino with HCSR04:**

****

**Introduction:**

The common problem faced in any house is the cleaning of dust present on top of the fan blades. Due to this dust a wheeze I created which results in most of the respiratory diseases like asthma, dust allergies etc. Timely cleaning of fans requires manpower and are time consuming. Our project works under three phases namely Set, Clean and Retract. The project is affixed to the top of the fan and is controlled using a Bluetooth based app.

Our project addresses these problems in an effective and cost-efficient manner and is fully automated.

**Main Text:**

The components used in this project are:

1. **Arduino Uno:**

* The **Arduino UNO** is an [open-source](https://en.wikipedia.org/wiki/Open-source) [microcontroller board](https://en.wikipedia.org/wiki/Microcontroller_board) based on the [Microchip](https://en.wikipedia.org/wiki/Microchip_Technology) [ATmega328P](https://en.wikipedia.org/wiki/ATmega328P) microcontroller and developed by [Arduino.cc](https://en.wikipedia.org/wiki/Arduino). The board is equipped with sets of digital and analog [input/output](https://en.wikipedia.org/wiki/Input/output) (I/O) pins that may be interfaced to various [expansion boards](https://en.wikipedia.org/wiki/Expansion_board) (shields) and other circuits. The board has 14 Digital pins, 6 Analog pins, and programmable with the [Arduino IDE](https://en.wikipedia.org/wiki/Arduino#Software) (Integrated Development Environment) via a type B [USB cable](https://en.wikipedia.org/wiki/USB_cable).



It can be powered by the USB cable or by an external [9-volt battery](https://en.wikipedia.org/wiki/9-volt_battery), though It accepts voltages between 7 and 20 volts.

* **Specifications:**
* Microcontroller: Microchip ATmega328P
* Operating Voltage: 5 Volts
* Input Voltage: 7 to 20 Volts
* Digital I/O Pins: 14 (of which 6 provide PWM output)
* Analog Input Pins: 6
* DC Current per I/O Pin: 20 mA
* DC Current for 3.3V Pin: 50 mA
* Flash Memory: 32 KB of which 0.5 KB used by bootloader
* SRAM: 2 KB
* EEPROM: 1 KB
* Clock Speed: 16 MHz
* Length: 68.6 mm
* Width: 53.4 mm
* Weight: 25 g

1. **HC-05 Bluetooth Module:**

* The Bluetooth module HC-05 is a MASTER/SLAVE module. By default, the factory setting is SLAVE. The Role of the module (Master or Slave) can be configured only by AT COMMANDS. The slave modules cannot initiate a connection to another Bluetooth device, but can accept connections. Master module can initiate a connection to other devices. The user can use it simply for a serial port replacement to establish connection between MCU and GPS, PC to your embedded project, etc.

### Hardware Features

* + Typical ‐80dBm sensitivity.
  + Up to +4dBm RF transmit power.
  + 3.3 to 5 V I/O.
  + PIO (Programmable Input/Output) control.
  + UART interface with programmable baud rate.
  + With integrated antenna.
  + With edge connector.

### Software Features

* + Slave default Baud rate: 9600, Data bits:8, Stop bit:1, Parity: No parity.
  + Auto‐connect to the last device on power as default.
  + Permit pairing device to connect as default.
  + Auto‐pairing PINCODE:”1234” as default



### A4988 Motor Driver:

### The A4988 is a complete micro stepping motor driver with built-in translator for easy operation. It is designed to operate bipolar stepper motors in full-, half-, quarter-, eighth-, and sixteenth-step modes, with an output drive capacity of up to 35 V and ±2 A. The A4988 includes a fixed off-time current regulator which has the ability to operate in Slow or Mixed decay modes.

### Features:

* + Simple step and direction control interface
  + Five different step resolutions: full-step, half-step, quarter-step, eighth-step, and sixteenth-step
  + Adjustable current control lets you set the maximum current output with a potentiometer, which lets you use voltages above your stepper motor’s rated voltage to achieve higher step rates
  + Intelligent chopping control that automatically selects the correct current decay mode (fast decay or slow decay)
  + Over-temperature thermal shutdown, under-voltage lockout, and crossover-current protection
  + Short-to-ground and shorted-load protection

1. **Nema 17 Stepper Motor:**

* This hybrid stepping motor has a 1.8° step angle (200 steps/revolution). Each phase draws 1.2 A at 4 V, allowing for a holding torque of 3.2 kg-cm. The motor has six color-coded wires terminated with bare leads that allow it to be controlled by both unipolar and bipolar stepper motor drivers. When used with a unipolar stepper motor driver, all six leads are used. When used with a bipolar stepper motor driver, the centre-tap yellow and white wires can be left disconnected (the red-blue pair gives access to one coil and the black-green pair gives access to the other coil).
* **Specifications:**
  + Size: 42.3 mm square × 48 mm, not including the shaft (NEMA 17)
  + Weight: 350 g (13 oz)
  + Shaft diameter: 5 mm “D”
  + Steps per revolution: 200
  + Current rating: 1.2 A per coil
  + Voltage rating: 4 V
  + Resistance: 3.3 Ω per coil
  + Holding torque: 3.2 kg-cm (44 oz-in)
  + Inductance: 2.8 MH per coil
  + Lead length: 30 cm (12″)

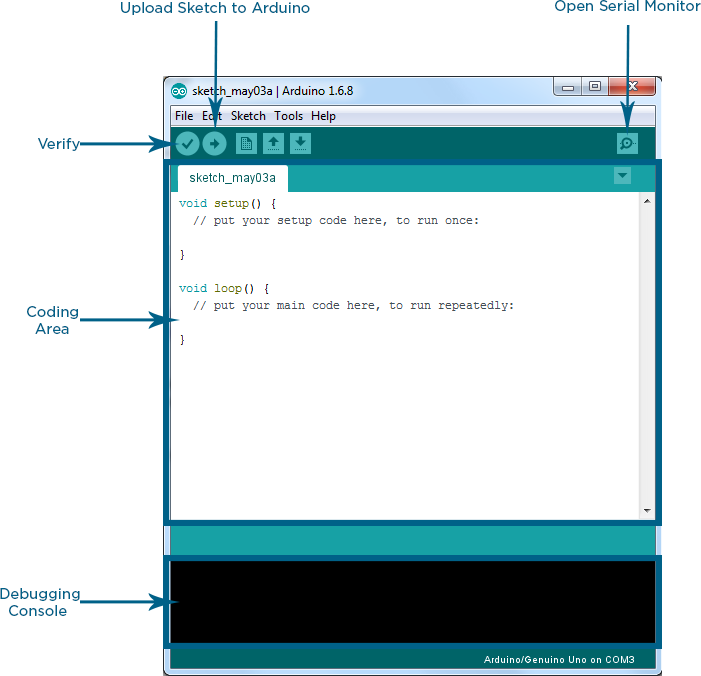


**Software:**

1. **Arduino IDE:**

The **Arduino integrated development environment (IDE)** is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main ()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.



**Working of the Prototype:**

Our prototype is programmed and controlled using Arduino – a microprocessor. It’s working is controlled by a Bluetooth based Android application. The prototype mainly works under 3 phases i.e. Set, retract, Clean. The require action is selected from the android app based on which the prototype works as coded. The commands from the app are received using HC-05 -Bluetooth module.

When the ‘Set’ option is selected the frame of the prototype is rotated about the axis of the fan using a DC motor. The motor rotates it in a range of angles and stops when a fan blade is detected. The fan blade’s position is detected using an ultrasonic sensor which are placed at the end of the frame rod. When the programmed distance is detected i.e. fan blade is detected the motor stops in turn aligning the cleaner parallel to the blades.

When the ‘Retract’ option is selected the duster responsible for cleaning the blade is folded using a servo motor such that it is parallel to the fan blade so as to get obstruction free movement of the blades when the fan is switched on.

When the ‘Clean’ option is selected the duster is brought into position i.e. perpendicular to the blade by using a servo motor. The cleaner rods are then rotated using a Nema 17 Stepper Motor which is interfaced and controlled by Arduino through A4988-motor driver. When the rods are rotated the nuts to which the duster is attached is moved front and back based on the direction of rotation of the motor. This movement of the duster in turn cleans the top of the fan blade.

**Prototype Code in Arduino IDE:**

#include<Servo.h>

Servo myServo1;

const int trigPin = 5;

const int echoPin = 6;

long duration;

int distance;

const int stepPin = 3;

const int dirPin = 4;

void setup()

{

pinMode(stepPin,OUTPUT);



pinMode(dirPin,OUTPUT);

Serial.begin(9600);

pinMode(trigPin,OUTPUT);

pinMode(echoPin,INPUT);

myServo1.attach(12);

}

void loop() {

if (Serial.available()>0)

switch(Serial.read())

{

case 'a':

SetTheBlade();

break;

case 'b':

CleanTheBlade();

break;

case 'c':

Retract();

break;

}

}

void SetTheBlade()

{

int i=0;

while(i==0){

digitalWrite(dirPin,HIGH);

for(int x = 0; x < 10000 ;x++) {

digitalWrite(stepPin,HIGH);



digitalWrite(stepPin,LOW);

delayMicroseconds(500);

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance= duration\*0.034/2;

Serial.print("Distance: ");

Serial.println(distance);

if(distance<=10)

{

x=20000;

i++;

}}

}}

void CleanTheBlade()

{

myservo1.write(0);

delay(1000);

digitalWrite(dirPin,HIGH);

for(int x = 0; x < 10000 ;x++) {

digitalWrite(stepPin,HIGH);

delayMicroseconds(500);

digitalWrite(stepPin,LOW);

delayMicroseconds(500);

}



digitalWrite(dirPin,LOW);

for(int x = 0; x < 10000 ;x++) {

digitalWrite(stepPin,HIGH);

delayMicroseconds(500);

digitalWrite(stepPin,LOW);

delayMicroseconds(500);

}

}

void Retract()

{

myServo1.write(180);

delay(1000);

}

**Future Scope:**

The project can be further developed to attain more stability and increase its efficiency. We can make further additions to the cleaning mechanism so as to even clean the bottom part of the fan blades.

**Conclusion:**

By this project we aim to solve the problem of cleaning the top of fan blades so as to prevent the formation of wheeze and also develop his project to clean the bottom part of the fan as well.

**Glossary:**

* Arduino Uno
* Nema 17 Stepper Motor
* HC-05 - Bluetooth Module
* HCSR04 - Ultrasonic sensor
* Servo Motor
* A4988 – Motor Driver

















