Article

Home Automation Using Arduino with Microsoft Visual Studio

By **Kishore Chowdary** on **Sep 30 2016**

Requirements

1. Visual Studio.
2. Arduino IDE.
3. Connecting wires (10 numbers or more).
4. Breadboard.
5. DC motors which can run by just 5 volts power supplly (2 numbers).
6. LED light (1 number).

Introduction

We people expect advancements in our day to day life. That is why I have thought of implementing a voice control technology using Arduino to take control over the electrical appliances at my home. Hence, in this article, I will explain to you how to create an application which helps you control the devices in your home, with your voice commands. So, to make it simple, I will explain how to connect a door which can be opened and closed and a fan along with a light which can be turned on and off, using the voice commands. Let me explain the process step by step.

Note 1 - If you feel that you are unable to understand this program, just click on the following link (Turning LED on and off using Voice commands) to learn a basic programming for Arduino. Then, it will become very easy for you to implement this technique.

Step 1- Programming the Arduino

- Open the Arduino IDE if already installed (or download and install it from this link).
- Now, open the IDE and enter the following code in it.

```
char incomingdata;
void setup() {
    pinMode(2, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(12, OUTPUT);
    pinMode(13, OUTPUT);
    Serial.begin(9600);
}
void loop() {
    incomingdata = Serial.read(); {
        if (incomingdata == 'a') {
            digitalWrite(2, HIGH);
        } else if (incomingdata == 'b') {
            digitalWrite(2, LOW);
        } else if (incomingdata == 'c') {
            digitalWrite(4, HIGH);
        } else if (incomingdata == 'd') {
            digitalWrite(4, LOW);
        } else if (incomingdata == 'e') {
            digitalWrite(12, HIGH);
            digitalWrite(13, LOW);
            delay(3500);
            digitalWrite(12, LOW);
        } else if (incomingdata == 'f') {
            digitalWrite(12, LOW);
            digitalWrite(13, HIGH);
            delay(3500);
            digitalWrite(13, LOW);
        }
    }
}
```

- In this program, I have used a delay of 3.5 seconds in the last two conditions. This is because of the length of the door which I have used in my prototype. You can change the delay length according to your prototype.
- Save this sketch at any location and verify the code.
- Now, connect your Arduino board and upload this sketch into the board.

Step 2 - Programming in Visual Studio

- Open Visual Studio and create new Windows Form application with any name you want.
- Once you have created a new form application, open the designer window of Form1.
- Now, from the tools box, drag and drop the SerialPort tool into the form1.
- This will help you communicate your program with the Arduino board.
- Now, you have to make your PC recognize your voice so that it can act as a command for the system.
- For this, we need to include a reference file called "System.Speech".
- This will now help you to give speech as an input to the computer.

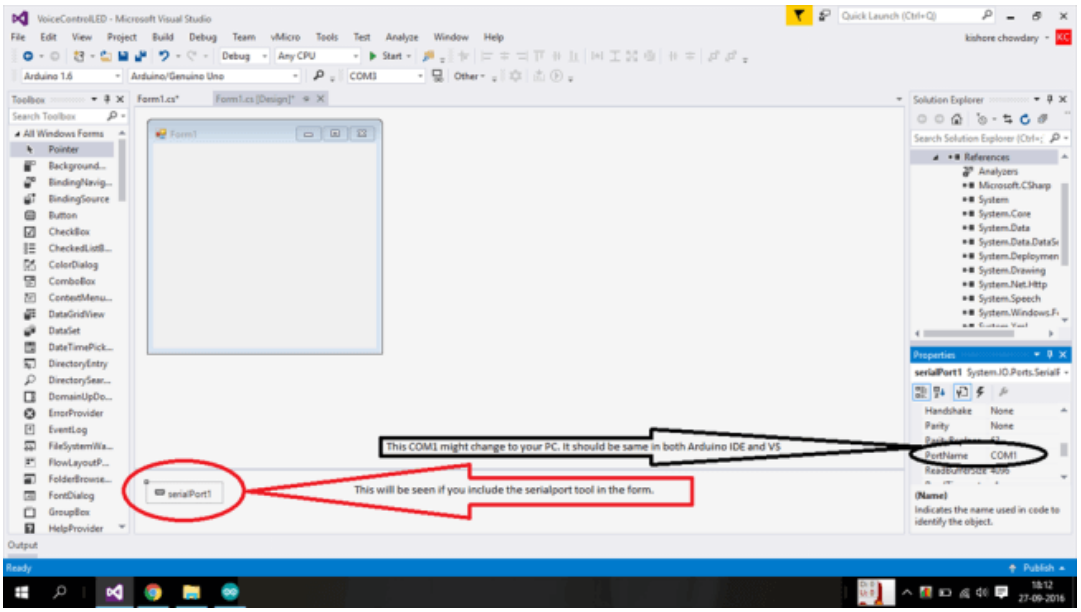
Step 3 - Writing code for home automation

- We need to write a piece of code which will help the computer to recognize our voice input.
- So, first we need to set a few predefined commands like LIGHT ON, LIGHT OFF, FAN ON, FAN OFF, DOOR OPEN and DOOR CLOSE.
- These commands will work as they are if you build your prototype by connecting a DC motor with a small fan wing to work like a fan and another DC motor with a door which can be moved up and down like a shutter door and also an LED which can work like a light.
- Remember again that you are building a prototype for an automated house and not a real house.
- Double click on the From1 and paste the following code in there.

```
using System;
using System.Speech.Recognition;
using System.Speech.Synthesis;
using System.Windows.Forms;
namespace VoiceAutomation {
    public partial class Form1: Form {
        public Form1() {
            InitializeComponent();
        }
        SpeechSynthesizer speechSynthesizerObj;
        private void Form1_Load(object sender, EventArgs e) {
            speechSynthesizerObj = new SpeechSynthesizer();
            speechSynthesizerObj.SpeakAsync("Welcome to the world of voice. Hope you are doing well. Give the commands.");
            SpeechRecognitionEngine sRecognize = new SpeechRecognitionEngine();
            Choices sList = new Choices();
            sList.Add(new string[] {
                "light on",
                "light off",
                "fan on",
                "fan off",
                "door open",
                "door close"
            });
            Grammar gr = new Grammar(new GrammarBuilder(sList));
            try {
                sRecognize.RequestRecognizerUpdate();
                sRecognize.LoadGrammar(gr);
                sRecognize.SpeechRecognized += sRecognize_SpeechRecognized;
                sRecognize.SetInputToDefaultAudioDevice();
                sRecognize.RecognizeAsync(RecognizeMode.Multiple);
                sRecognize.Recognize();
            } catch {
                return;
            }
        }
        private void sRecognize_SpeechRecognized(object sender, SpeechRecognizedEventArgs e) {
            if (e.Result.Text == "lights on") {
                serialPort1.Open();
                serialPort1.Write("a");
                serialPort1.Close();
            } else if (e.Result.Text == "lights off") {
                serialPort1.Open();
                serialPort1.Write("b");
                serialPort1.Close();
```

```
45.             } else if (e.Result.Text == "fan on") {
46.                 serialPort1.Open();
47.                 serialPort1.Write("c");
48.                 serialPort1.Close();
49.             } else if (e.Result.Text == "fan off") {
50.                 serialPort1.Open();
51.                 serialPort1.Write("d");
52.                 serialPort1.Close();
53.             } else if (e.Result.Text == "door open") {
54.                 serialPort1.Open();
55.                 serialPort1.Write("e");
56.                 serialPort1.Close();
57.             } else if (e.Result.Text == "door close") {
58.                 serialPort1.Open();
59.                 serialPort1.Write("f");
60.                 serialPort1.Close();
61.             }
62.         }
63.     }
64. }
```

- If any error occurs when you copy and paste your code, then it would be just because of the serialPort tool which you could have forgoten to include in the form.
- For this issue, have a look over step 2 again.

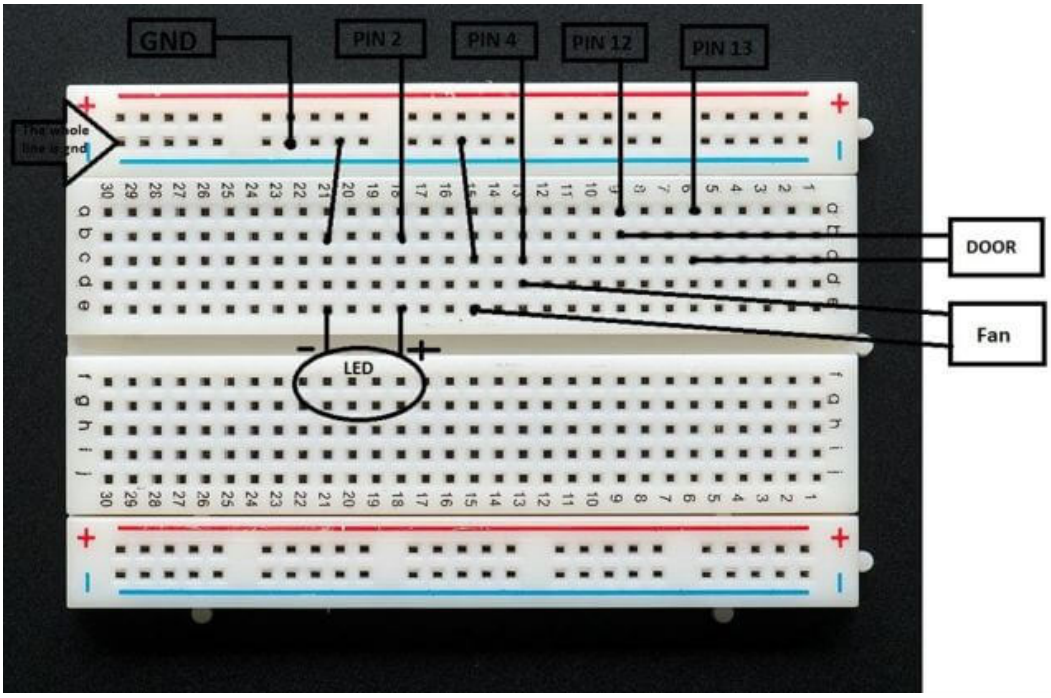


Step 4 - Giving the connections in the breadboard

- According to the program, I have used pin 2 of the Arduino for light, pin 4 for the fan and pin 12, 13 for the door.
- Since the door needs to move in both the directions, we need to make the dc motor of the door to run in both directions.
- Hence, door is give with two pin connections.

Connecting the setup

1. Take a ground connection from the Arduino and connect it to the breadboard.
2. Take pin 2 and connect it to the LED along with the ground connection.
3. Take pin 4 and connect it to the DC motor along with the ground connection which will act as a fan.
4. Take pins 12, 13 and connect each of them to one end of the motor so that it can run in both directions.
5. Take a look over the image for clarification.



Working of the setup

1. When you say the command light on, lights will turn on and when you say the command light off, light will turn off.
2. When you say fan on, fan will turn on and when you say fan off, fan will turn off.
3. When you say door open, the motor of the door will rotate in a direction and then turns off after a few seconds.
4. When you say door close, the motor will rotate in another direction and then turns off after few seconds.

Final step

- If you are unable to build a simulated model, you can just give connections for the LEDs and the DC motors and check the output.
- Here, for the door, the motor will just rotate in bi-directions. Hence, you need to create the door such that it can move up and down.
- Once again, check up all your code and the connections.
- Now, run the program and say the commands like,
    - Light on
    - Light off
    - Fan on
    - Fan off
    - Door open
    - Door close
- Thus, you have finally created your program and created an automated home.

Note - I also have few other concepts for this automation technique. They include the password, security alarm, scheduled power on and off of the gadgets, and so on. Hope you will understand and like this concept. Thank you.

Thank you for using C# Corner