Control de LED utilizando Arduino y Visual C#: Introducción a la Programación Básica

Nombre de integrantes: Eli Emmanuel Flores Blanco #23580416

Greisy Margarita Lima Silverio #23580378

Escuela: Instituto Tecnologico de Reynosa

Carrera: Ing. Mecatronica

Semestre: 2do

Fecha de entrega: 12/May/25

Resumen

Esta práctica tuvo como finalidad demostrar la interacción entre hardware y software mediante el uso de una placa Arduino y un programa en Visual C# con reconocimiento de voz. Se diseñó un circuito electrónico básico que permitía controlar tres LEDs de diferentes colores mediante comandos hablados. La programación incluyó el uso de la biblioteca System.Speech y comunicación serial. Los resultados fueron exitosos, ya que los LEDs reaccionaban correctamente a los comandos emitidos, mostrando la funcionalidad de la interfaz desarrollada.

Introducción

El avance de la tecnología ha facilitado el desarrollo de sistemas interactivos que permiten una comunicación más natural entre el ser humano y las máquinas. Uno de los recursos más accesibles para estudiantes y desarrolladores es la plataforma Arduino, que permite aprender y experimentar con conceptos de electrónica y programación de forma sencilla.

En esta práctica se utilizó Arduino en conjunto con Visual Studio y el lenguaje C# para desarrollar un sistema capaz de responder a comandos de voz y realizar acciones específicas en un circuito. Esto permitió integrar conceptos teóricos y prácticos, favoreciendo el aprendizaje activo.

Arduino es una plataforma abierta basada en hardware y software libre que facilita el aprendizaje de la programación y la electrónica (Banzi, 2011).

Materiales y Métodos

Materiales:

- 1 placa Arduino UNO
- 1 protoboard
- 3 LEDs (rojo, azul y verde)
- 3 resistencias de 220 ohmios
- Cables de conexión
- Computadora con Visual Studio Community
- Biblioteca de reconocimiento de voz (System.Speech)
- Cable USB

Método:

Se construyó un circuito donde cada LED fue conectado a los pines digitales 2, 3 y 4 del Arduino. Cada LED tenía en serie una resistencia de 220 ohmios para evitar sobrecorriente. El sistema fue programado para encender o apagar los LEDs de acuerdo con comandos recibidos desde una aplicación en C#, utilizando comunicación serial.

Fragmento de código Arduino:

```cpp
if (Serial.available() > 0) {
  char comando = Serial.read();
  if (comando == 'R') {
    digitalWrite(ledRojo, HIGH);
  }
}
```

Fragmento de código en C#:

```csharp
if (comando == "encender rojo") {
    arduino.Write("R");
}
```

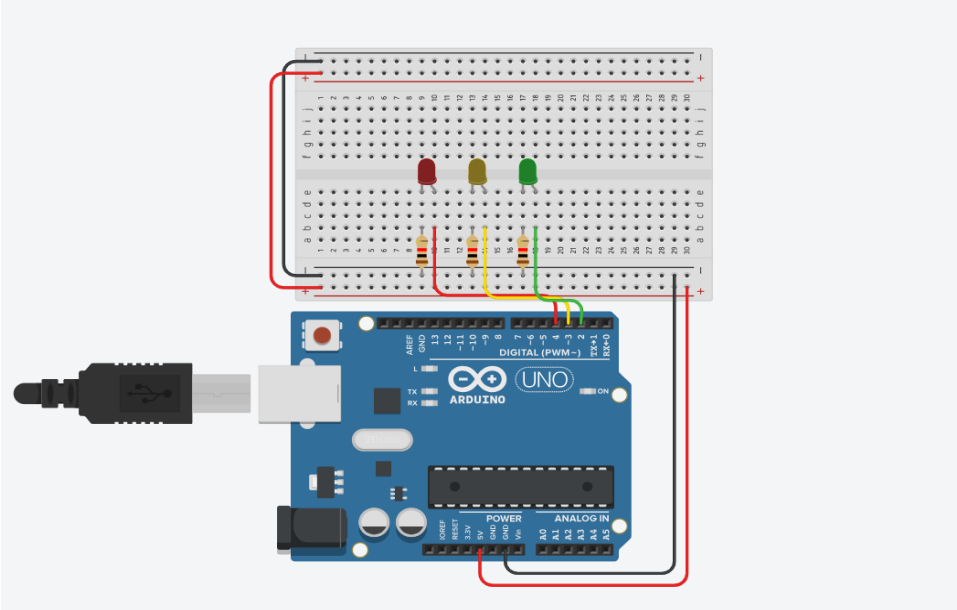
Resultados

Durante la ejecución de la aplicación, se observó que el sistema respondía de manera adecuada a los comandos de voz previamente programados. Los LEDs se encendían o apagaban según las instrucciones dadas verbalmente, mostrando una buena integración entre hardware y software.

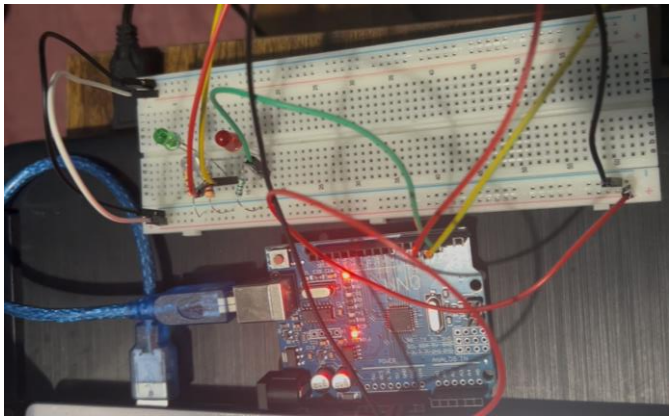
Tabla: Comandos y funciones

Comando de voz	Función realizada
encender rojo	Enciende LED rojo
apagar rojo	Apaga LED rojo
encender azul	Enciende LED azul
apagar azul	Apaga LED azul
encender verde	Enciende LED verde
apagar verde	Apaga LED verde
encender todos	Enciende todos los LEDs
apagar todos	Apaga todos los LEDs

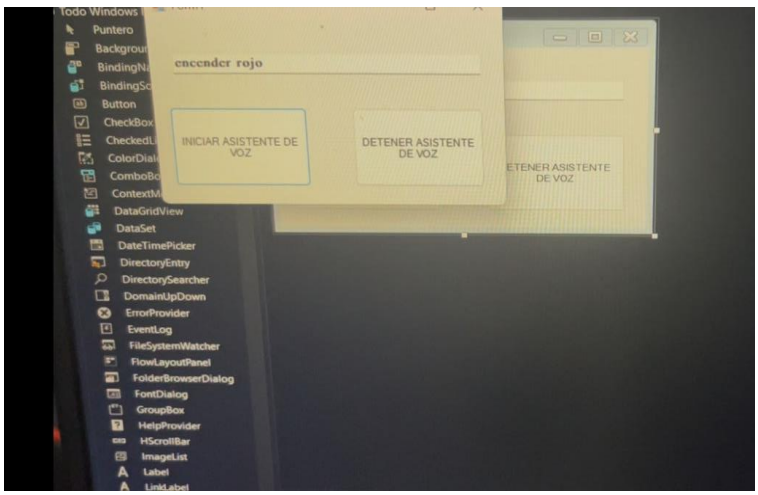
Simulacion en tinkercad



Circuito



Interfaz Visual Studio



Discusión

Los resultados obtenidos demostraron que es posible implementar un sistema de control por voz usando herramientas relativamente simples y económicas. Aunque el reconocimiento de voz puede verse afectado por ruidos ambientales, en condiciones normales el sistema funcionó adecuadamente. Esta práctica también facilitó la comprensión de cómo establecer una comunicación serial entre una aplicación en C# y un microcontrolador.

Entre las dificultades encontradas se destacan la configuración del puerto COM y la sensibilidad del reconocimiento de voz. Sin embargo, una vez ajustados estos parámetros, el sistema fue funcional y robusto dentro de las condiciones del entorno controlado.

Conclusiones

La práctica permitió integrar conocimientos de electrónica básica, programación de microcontroladores y desarrollo de interfaces en C#. Se logró implementar un sistema funcional que responde a comandos de voz, lo cual tiene diversas aplicaciones en el ámbito de la domótica y accesibilidad tecnológica. Se refuerza así la importancia de aprender herramientas de desarrollo multiplataforma como Arduino y Visual Studio para prototipado rápido e interacción hombre-máquina.

Referencias

Banzi, M. (2011). *Getting started with Arduino* (2nd ed.). O'Reilly Media.

Microsoft. (2023). *System.Speech Namespace*. https://learn.microsoft.com/en-us/dotnet/api/system.speech

Anexos

Código de Visual C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Speech.Recognition;
using System.Speech.Synthesis;
```

```
namespace proyectoasistente
{
    public partial class Form1 : Form
    {
        SpeechRecognitionEngine escucha = new SpeechRecognitionEngine(new
System.Globalization.CultureInfo("es-ES"));
        SpeechSynthesizer hablae = new SpeechSynthesizer();
        System.IO.Ports.SerialPort arduino;

        public Form1()
        {
            InitializeComponent();
            arduino = new System.IO.Ports.SerialPort();
            arduino.PortName = "COM3"; // Asegúrate de que sea el puerto
correcto
            arduino.BaudRate = 9600;
            arduino.Open();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            // Configuración adicional si es necesario
        }

        private void button1_Click(object sender, EventArgs e)
        {
            escucha.SetInputToDefaultAudioDevice();
            escucha.SpeechRecognized += Deteccion;

            // Lista de comandos reconocibles
            Choices alternativas = new Choices();
            alternativas.Add(new string[] {
                "encender rojo", "apagar rojo",
                "encender azul", "apagar azul",
                "encender verde", "apagar verde",
                "encender todos", "apagar todos"
            });

            // Crear la gramática
            GrammarBuilder gb = new GrammarBuilder();
            gb.Append(alternativas);
            Grammar g = new Grammar(gb);
            escucha.LoadGrammar(g);

            // Iniciar reconocimiento
            escucha.RecognizeAsync(RecognizeMode.Multiple);

            hablae.SpeakAsync("reconocimiento de voz activado");
        }

        private void Deteccion(object sender, SpeechRecognizedEventArgs e)
        {
            // Mostrar comando en el textbox
            textBox1.Text = e.Result.Text;
            string comando = e.Result.Text.ToLower();

            // Comandos para cada color
            if (comando == "encender rojo")
```

```
            {
                arduino.Write("R");
            }
            else if (comando == "apagar rojo")
            {
                arduino.Write("Z");
            }
            else if (comando == "encender azul")
            {
                arduino.Write("A");
            }
            else if (comando == "apagar azul")
            {
                arduino.Write("Y");
            }
            else if (comando == "encender verde")
            {
                arduino.Write("V");
            }
            else if (comando == "apagar verde")
            {
                arduino.Write("X");
            }
            else if (comando == "encender todos")
            {
                arduino.Write("T");
            }
            else if (comando == "apagar todos")
            {
                arduino.Write("F");
            }
        }

        private void Form1_FormClosing(object sender, FormClosingEventArgs e)
        {
            if (arduino.IsOpen)
            {
                arduino.Close(); // Cierra el puerto cuando se cierre el
formulario
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            escucha.RecognizeAsyncStop(); // Detiene el reconocimiento
        }
    }
}
```

Código de Arduino IDE

```
// Pines de los LEDs
const int ledRojo = 2;
const int ledAzul = 3;  // <- antes era ledAmarillo
const int ledVerde = 4;

void setup() {
  // Inicializar los pines como salidas
  pinMode(ledRojo, OUTPUT);
  pinMode(ledAzul, OUTPUT);
  pinMode(ledVerde, OUTPUT);

  // Iniciar la comunicación serial
  Serial.begin(9600);
}

void loop() {
  if (Serial.available() > 0) {
    char comando = Serial.read();

    switch (comando) {
      case 'R':
        digitalWrite(ledRojo, HIGH);
        break;
      case 'Z':
        digitalWrite(ledRojo, LOW);
        break;
      case 'A':  // Encender azul
        digitalWrite(ledAzul, HIGH);
        break;
      case 'Y':  // Apagar azul
        digitalWrite(ledAzul, LOW);
        break;
      case 'V':
        digitalWrite(ledVerde, HIGH);
        break;
      case 'X':
        digitalWrite(ledVerde, LOW);
        break;
      case 'T': // Encender todos
        digitalWrite(ledRojo, HIGH);
        digitalWrite(ledAzul, HIGH);
        digitalWrite(ledVerde, HIGH);
        break;
```

```
      case 'F': // Apagar todos
        digitalWrite(ledRojo, LOW);
        digitalWrite(ledAzul, LOW);
        digitalWrite(ledVerde, LOW);
        break;
    }
  }
}
```

Circuito terminado