Programación Avanzada

Instrumentación Electrónica

Domínguez Chávez José Alfonso

Proyecto Final

"Circuito simulador de pedal de fuzz usando Arduino IDE"

CASTILLO ACOSTA JOSUE MARCELO

DIAZ RODRIGUEZ KAYLEE MICHELLE

HERNÁNDEZ HERNÁNDEZ JULIANA

NAVARRO HERNÁNDEZ HUGO JESUS

SALAS MARTÍNEZ CESAR DAVID

Índice

1. Introducción
2. Descripción general del sistema
3. Componentes del sistema
4. Instalación y armado
5. Configuración del software
6. Instrucciones de uso
7. Exportación de datos o funciones especiales
8. Resolución de problemas comunes
9. Mantenimiento y recomendaciones
10. Créditos y contacto
11. Anexos (códigos fuente, diagramas, referencias)

1. Introducción

Los efectos de sonido son esenciales para modificar y enriquecer el audio en instrumentos eléctricos. Este manual se centra en el efecto “fuzz”, el cual distorsiona la señal para producir un sonido rasposo y saturado, muy utilizado en géneros como el rock. El proyecto consiste en la creación de un pedal de fuzz, controlado mediante un circuito analógico y un ESP32, que permite monitorear y modificar la señal digitalmente.

2. Descripción general del sistema

Para este proyecto se diseñó una simulación de pedal de fuzz armando un circuito analógico que posee un amplificador operacional TL072CP y un ESP32 WROOM-32D programado con Arduino IDE, su finalidad es controlar electrónicamente el efecto y monitorear los parámetros y ondas de salida. El sistema implementa la modulación de señales mediante un circuito analógico y el análisis y conversión de señales por medio del ESP32.

3. Componentes del sistema

Para la elaboración de este proyecto, se utilizaron los siguientes materiales y componentes para el armado del circuito analógico:

- 1 capacitor de poliéster de 4.7u.
- 1 resistencia de 1k.
- 1 resistencia de 10k.
- 1 resistencia de 1.2k.
- 1 resistencia de 150.
- 5 resistencias de 5.6k.
- 2 sockets para Jack 6.3mm.
- 1 opamps TL072CP.
- 1 botón de enganche de 9 pines.

En cuanto al equipo utilizado, tenemos:

- 1 ESP32 WROOM-32D.
- 1 fuente de voltaje.
- 1 amplificador para bajo eléctrico.
- 2 cables tipo Jack 6.3mm Macho-Macho.
- 9 cables tipo caimán.

4. Instalación y armado

- ESP32 DevKit V1
- Transistores (BC108, 2N3904 u otro según diseño de fuzz)
- Resistencias y capacitores según el esquema
- Jack de entrada/salida de audio
- Potenciómetros (ganancia, tono, volumen)
- Fuente de alimentación de 9V o batería
- Protoboard o PCB
- Cables y herramientas básicas (soldador, estaño)

Pasos:

1. **Montaje del circuito analógico:** Soldar o insertar los componentes del fuzz en protoboard o PCB según el esquema. Asegurar la correcta polaridad de los componentes.

2. **Conexión al ESP32:** Conectar la salida del fuzz al ADC del ESP32 y la salida DAC del ESP32 a un buffer o directamente al jack de salida.

3. **Instalación del ESP32:** Montar el ESP32 en su zócalo y asegurarse de que esté alimentado correctamente.

4. **Verificación de conexiones:** Utilizar un multímetro para verificar que no haya cortos ni errores de conexión.

5. Configuración del software

Requisitos Previos:

- Asegúrate de tener instalado el entorno de desarrollo Arduino IDE.
- Asegúrate de tener los controladores necesarios para tu placa ESP32.

2. **Conexiones de Hardware**:

- Conecta el sensor o dispositivo que proporcionará la señal analógica al pin definido como **ENTRADA** (en este caso, el pin 34).
- Conecta el pin DAC (en este caso, el pin 25) a la carga o circuito que recibirá la señal analógica.

3. **Configuración del Código**:

- Abre el Arduino IDE.
- Crea un nuevo sketch y copia el siguiente código:

```
//V.1.1.0
#define ENTRADA 34  // Pin analógico conectado al circuito offset
#define pinDAC1 25   //Definimos el pin de lectura de datos del sensor

//Mapeo del valor del rango con numeros flotantes
float mapFloat(float x, float in_min, float in_max, float out_min, float out_max) {
return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
void setup() {

}

void loop() {
float e = 2.71828; //Definir constante
float lectura = analogRead(ENTRADA); //Leer datos del analogico
float x = mapFloat(lectura, 0, 4096, -1.0, 1.0); //Mapear valor ya leido en el rango -1.0 y 1.0
```

```
float Resultado = (0.4 / (1 + pow(e, 10 * (x + 0.03)))) - 0.2;
float Resultado2 = (0.6 / (1 + pow(e, 90 * (x + 0.02)))) - 0.3;
float Resultado3 = (0.8 / (1 + pow(e, 90 * (x + 0.01)))) - 0.4;
float Resultado4 = (1.0 / (1 + pow(e, 90 * (x + 0.01)))) - 0.5;
float Resultado5 = (1.2 / (1 + pow(e, 70 * (x + 0.02)))) - 0.6;
float Resultado6 = (1.4 / (1 + pow(e, 60 * (x - 0.03)))) - 0.7;
float Resultado7 = (1.6 / (1 + pow(e, 65 * (x - 0.04)))) - 0.8;

//Parte para aumentar ganancia si es necesario
float ResultadoAmplificado = Resultado * 1.0;  // Ganancia ajustable
ResultadoAmplificado = constrain(ResultadoAmplificado, -1.0, 1.0);


// Convertimos a 8 bits (0-255) centrado para ser más eficiente
uint8_t muestra8bit = map(Resultado, -10, 10, 0, 255);


dacWrite(pinDAC1, muestra8bit);//Escribir al DAC (0-255)

delayMicroseconds(45); // Aproximadamente 22.2 kHz pausa para lograr una frecuencia
requerida
}
```

4. **Subida del Código**:

- Conecta tu placa ESP32 a la computadora.
- Selecciona el tipo de placa y el puerto correcto en el menú "Herramientas" del Arduino IDE.
- Haz clic en el botón de "Subir" para cargar el código en la placa.

5. **Verificación**:

- Una vez que el código esté cargado, verifica que el sistema esté funcionando correctamente. Puedes utilizar un osciloscopio o un multímetro para medir la salida del DAC y asegurarte de que los valores se están generando como se

Notas Adicionales:

- Asegúrate de que las conexiones de hardware sean correctas para evitar daños en la placa o en los componentes.
- Puedes ajustar la ganancia en la línea **float ResultadoAmplificado = Resultado * 1.0;** según sea necesario para tu aplicación específica.
- espera.

6. Instrucciones de uso

1. Conecta tu guitarra eléctrica al jack de entrada del circuito.
2. Alimenta el circuito mediante la fuente de voltaje o con el ESP32 conectado a una computadora.
3. Conecta la salida del circuito a un amplificador de guitarra.
4. Utiliza el botón de cambio para alternar entre el canal limpio y el canal con efecto fuzz.
5. Si está conectada la pantalla OLED, observarás la forma de onda en tiempo real.

7. Exportación de datos o funciones especiales

- El sistema puede exportar los datos de audio procesados en formato .csv para su análisis con Python.
- Utilizando el script en Python provisto (v1.0.3), es posible grabar audio desde el ESP32 y guardarlo en formato .wav.
- La señal procesada también se puede visualizar en la pantalla OLED conectada.

8. Resolución de problemas comunes

Problema	Solución
Solo se escucha ruido al activar el efecto	Asegúrate de no usar simultáneamente la salida DAC y el puerto serial.
El ESP32 no recibe bien la señal	Verifica el offset del circuito y la etapa de amplificación.
La pantalla OLED no muestra nada	Comprueba conexiones I2C y dirección 0x3C; prueba sin enviar datos por el DAC.

9. Mantenimiento y recomendaciones

- Verifica las conexiones del circuito regularmente, especialmente jacks y pines de ESP32.
- No alimentes el sistema con más voltaje del necesario (recomendado: 5V o 3.3V para ESP32).
- Almacena el circuito en un contenedor plástico o caja de proyecto para evitar cortocircuitos.
- Realiza pruebas de software antes de cargar códigos nuevos al ESP32.

Anexos

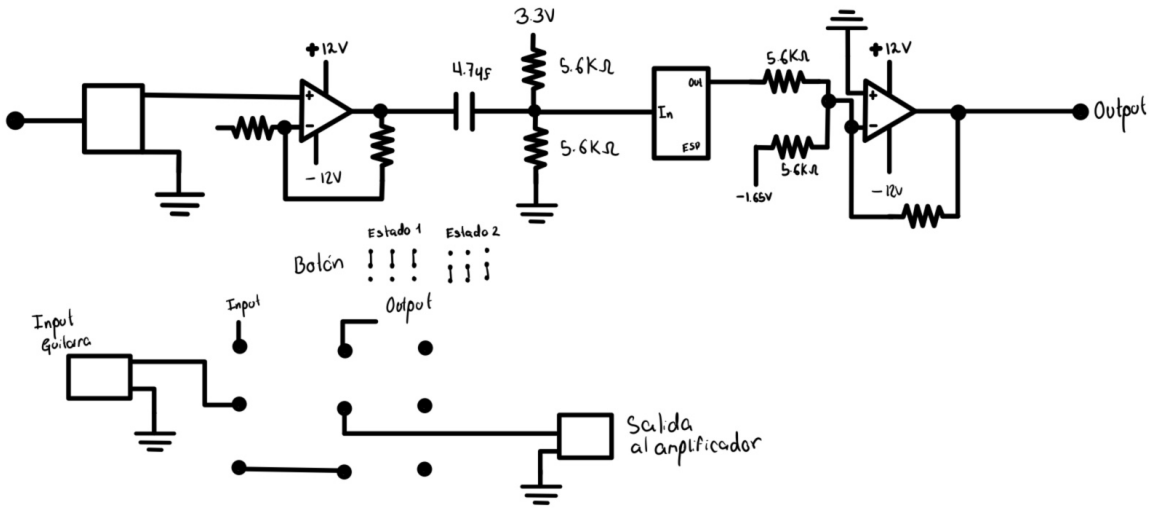

Diagrama eléctrico del circuito.