**Sistema de Control de Temperatura con Arduino Mega ADK y ESP32**

**Descripción**

Este proyecto implementa un sistema de control de temperatura en tiempo real utilizando un **Arduino Mega ADK** y un **ESP32**. El sistema regula la temperatura en un conducto de aire usando sensores de temperatura LM35 y celdas Peltier para generar aire caliente o frío según sea necesario.

El ESP32 se conecta a una base de datos para recibir las configuraciones deseadas y las envía al Arduino Mega ADK a través de comunicación serial. El sistema se encarga de medir la temperatura actual y ajustarla a un setpoint especificado, controlando las celdas Peltier y los ventiladores de manera eficiente.

**Características**

* **Comunicación Serie:** Interacción entre el Arduino y el ESP32 a través de pines TX2 y RX2.
* **Control de Temperatura:** Tres modos de operación: todo apagado, generación de calor y generación de frío.
* **Lectura de Sensores:** Utiliza sensores LM35 para medir la temperatura.
* **Control de Actuadores:** Controla celdas Peltier y ventiladores para ajustar la temperatura.
* **No se usan Delays:** Uso de la función millis() para evitar bloqueos y mejorar la eficiencia.

**Componentes Requeridos**

* **Arduino Mega ADK**
* **ESP32**
* **Sensores de Temperatura LM35** (2 unidades)
* **Celdas Peltier IRF440N** (para calor y frío)
* **Cartucho de Calor**
* **Ventiladores de Aire Caliente y Frío**
* **Ventilador Disipador de Calor**
* **Cables de Conexión**
* **Fuente de Alimentación**

**Esquema de Conexiones**

**Entradas**

* **Sensor de Temperatura LM35 (Calor):** Conectado a PIN A0.
* **Sensor de Temperatura LM35 (Frío):** Conectado a PIN A1.

**Salidas**

* **Celda Peltier (Calor):** Conectada a PIN DIGITAL 5.
* **Celda Peltier (Frío):** Conectada a PIN DIGITAL 6.
* **Cartucho de Calor:** Conectado a PIN DIGITAL 7.
* **Ventilador de Aire Caliente:** Conectado a PIN PWM 8.
* **Ventilador de Aire Frío:** Conectado a PIN PWM 9.
* **Ventilador Disipador de Celda Frío:** Conectado a PIN DIGITAL 10.

**Comunicación**

* **RX2:** PIN 17 (Arduino) conectado a TX del ESP32.
* **TX2:** PIN 16 (Arduino) conectado a RX del ESP32.

**Instalación y Configuración**

1. **Conectar los componentes:** Sigue el esquema de conexiones para conectar todos los sensores, actuadores y el ESP32 al Arduino Mega ADK.
2. **Subir el Código:** Usa el IDE de Arduino para subir el código al Arduino Mega ADK.
3. **Configurar el ESP32:** Asegúrate de que el ESP32 esté configurado para conectarse a la base de datos y enviar los comandos adecuados al Arduino.
4. **Alimentación:** Provee la alimentación adecuada para el Arduino, el ESP32 y los actuadores.

**Funcionamiento del Código**

El código está dividido en las siguientes secciones:

1. **Inicialización:**
   * Configura los pines de entrada y salida.
   * Establece la comunicación serial entre el Arduino y el ESP32.
2. **Loop Principal:**
   * comunicacion(): Lee los datos enviados desde el ESP32 (estado del sistema y setpoint de temperatura).
   * stateMachine(): Gestiona la máquina de estados para controlar las salidas del sistema.
3. **Funciones Auxiliares:**
   * setpoint(String texto): Convierte el valor de temperatura recibido como string a float.
   * caso1(): Apaga todas las salidas.
   * caso2(): Controla la generación de calor ajustando las salidas según la temperatura medida.
   * caso3(): Controla la generación de frío ajustando las salidas según la temperatura medida.
   * controlTemperatura1() y controlTemperatura2(): Miden la temperatura y ajustan las celdas Peltier y ventiladores para alcanzar el setpoint deseado.

**Ejecución**

1. **Inicialización:**
   * El sistema inicia en estado apagado (caso 1) con todas las salidas desactivadas.
2. **Control de Temperatura:**
   * En caso 2 (generación de calor), el sistema activará las celdas y ventiladores para calentar el conducto hasta alcanzar el setpoint.
   * En caso 3 (generación de frío), el sistema activará las celdas y ventiladores para enfriar el conducto hasta alcanzar el setpoint.
3. **Monitoreo:**
   * El sistema imprime por el puerto serial la temperatura medida y el estado actual para fines de depuración.

**Contribuciones**

Las contribuciones son bienvenidas. Si tienes ideas o mejoras, por favor abre un *issue* o envía un *pull request*.