Frontier Semiconductor Bangladesh Ltd.

EE Recruitment (Embedded Firmware Development)

Title: Temperature-Based Fan Speed Control System.

Submitted By:

Shahid Ahamed Hasib

BSc in Mechatronics Engineering

Rajshhai University of Engineering & Technology (RUET)

Email: Shahidhasib586@gmail.com

Phone: +8801632552003

Project Title: Temperature-Based Fan Speed Control System.

Introduction: An automated temperature control system is developed for a room. The room has a controllable fan for that purpose. Both manual and automatic controls is available. A keypad for the system inputs and a display for interface purposes are implemented with an RGB LED to show the current state of the room's temperature.

Objectives:

- To develop a system that monitors the ambient temperature.
- To implement automatic fan control.
- To implement a manual control for the fan.
- To show the current state using RGB led.
- To use keypad and display for the features and setup options.

Components:

The following components are used for the project implementation.

i. Platform: The “Proteus 8 professional” software is used for the project simulation.
ii. Controller: Arduino Uno “MCU” is used for the project implementation
iii. Temperature Sensor: “LM35” temperature sensor is used to sense the temperature.
iv. Display: A “16x2” LCD display is used to show the temperature, fan speed and anticipated manus.
v. Keypad: A “4x4” keypad is used for the menu navigation.
vi. Fan: A “DC motor” is used as fan.
vii. Motor Driver: “L293D” motor driver module is used for fan control.
viii. LED: An “RGB LED” is used to show the system states.
ix. Others: Resistors, and some power sources.

Methodology:

Temperature Sensing: The project utilizes a temperature sensor (LM35) to measure the ambient temperature. Analog readings from the sensor are converted to Celsius and displayed on the LCD screen. A 4x4 keypad provides user interaction for configuration. The LCD screen displays temperature readings, fan speed, and menu options. Fan Speed Control: Fan speed is adjusted based on the following temperature conditions: Normal temperature range (20°C to 25°C): Low fan speed, High temperature range (25°C to 30°C): Gradual increase in fan speed, Exceeding 30°C: Maximum fan speed, below 20°C: Fan turned off. Menu System: Users can access a menu system using the keypad. Menu options include setting upper and lower temperature thresholds.

Circuit Diagram:

The circuit diagram for this project is as follows:

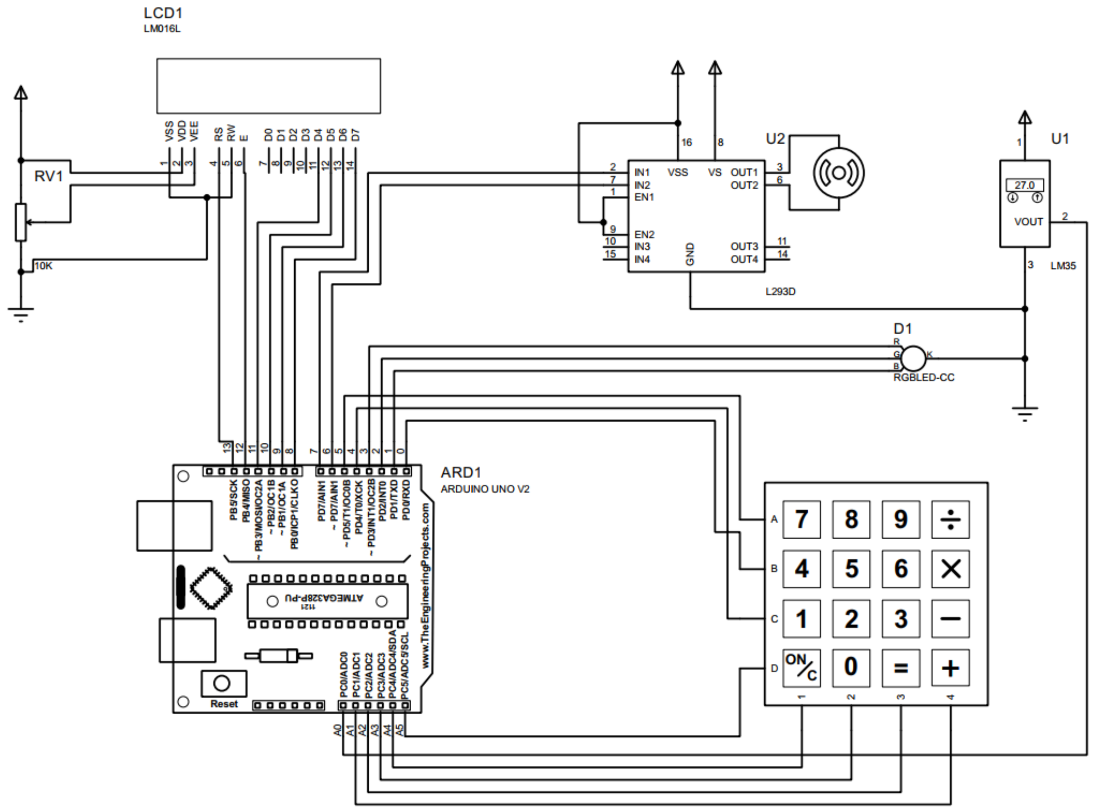

Fig. 1: Circuit Diagram.

Challenges Faced:

Keypad Integration: Configuring and integrating the 4x4 keypad required careful attention to wiring and library usage. Ensuring reliable keypad input was challenging due to hardware variations. LCD Display: Initializing and displaying information on the LCD screen involved addressing challenges with the Liquid Crystal library. State Management: Managing the various states of the system (e.g., SETUP, AMBIENT, HEAT) required meticulous coding to avoid unexpected behavior.

Algorithmic Overview:

In the code for the project “Temperature-Based Fan Speed Control System”, the following strategies are heeded:

Initialization:

Set up the Arduino pins for the LCD, keypad, and fan speed control. Initialize the LCD, custom characters, and other necessary peripherals. Define constants for various system states (HOME, SETUP, AMBIENT, HEAT, COLD, SET_UPPER_THRESHOLD, SET_LOWER_THRESHOLD) and initialize the current state to HOME.

Main Loop: Continuously read a character from the keypad to check for user input. Based on the current state and user input, transition to the appropriate state (e.g., from HOME to SETUP, AMBIENT, HEAT, or COLD).

Temperature Sensing (Readtemp Function):

Read the analog value from the LM35 temperature sensor. Convert the analog value to temperature in Celsius. Display the temperature on the LCD, including a custom degree Celsius symbol.

Fan Speed Control: Implement fan speed control logic based on the temperature reading: Normal Temperature Range (20°C to 25°C): Set the fan speed to a minimum value (e.g., 2). Display green LED as an indicator. High Temperature Range (25°C to 30°C): Gradually increase the fan speed as the temperature rises. Display red LED as an indicator. Exceeding 30°C: Set the fan speed to the maximum value (e.g., 255). Display red LED as an indicator. Below 20°C: Turn off the fan. Display blue LED as an indicator.

User Interface and Menu System:

Implement the HOME state: Check for user input (e.g., '0' key) to enter the SETUP menu. Implement the SETUP state: Handle user input to navigate between different setup options (e.g., '1' for AMBIENT, '2' for HEAT). Display relevant information on the LCD for each option. Implement the AMBIENT and HEAT states: Display specific information and descriptions related to these modes. Allow users to set upper and lower temperature thresholds in the SET_UPPER_THRESHOLD and SET_LOWER_THRESHOLD states. Save the user-configured thresholds to EEPROM when the '#' key is pressed._Display Management: Continuously update the LCD to display real-time temperature and fan speed information. Use custom characters for the degree Celsius symbol.

Challenges Handling: Implement error handling and state management to address challenges related to the keypad, LCD, and user input.

Sudo Code:

```
Initialize:
  Setup pins for LCD, keypad, and fan speed control
  Initialize LCD and custom characters
  Define constants for system states (HOME, SETUP, AMBIENT, HEAT, COLD, SET_UPPER_THRESHOLD, SET_LOWER_THRESHOLD)
  Initialize currentState to HOME
  Initialize upperThreshold and lowerThreshold to default values (e.g., 30°C and 20°C)


Main Loop:
  Repeat indefinitely:
    Read user input character from keypad


    // State Transition Logic
    If currentState is HOME:
      If user presses '0':
        Set currentState to SETUP
        Call displaySetupMenu()


    Else If currentState is SETUP:
      Call handleSetupMenu(userInput)
 // Add logic for other states (AMBIENT, HEAT, COLD, SET_UPPER_THRESHOLD, SET_LOWER_THRESHOLD)
// Temperature and Fan Control Logic
    Call Readtemp()
    If temperature falls within specific ranges:
      Adjust fan speed accordingly
      Display temperature and fan speed on LCD
      Control LEDs based on temperature


    Delay for stability (e.g., 1 second)
Readtemp Function:
  Read analog value from temperature sensor (e.g., LM35)
  Convert analog value to temperature in Celsius
  Display temperature on LCD using a custom degree Celsius symbol
handleSetupMenu Function:
  Accept user input as parameter (userInput)
  Switch (userInput):
    Case '1':
      Set currentState to AMBIENT
      Clear LCD and display relevant information about AMBIENT mode
    Case '2':
      Set currentState to HEAT
      Clear LCD and display relevant information about HEAT mode
    // Add cases for other setup options
displaySetupMenu Function:
  Clear LCD
  Display setup menu options on the LCD
  Example:
  Set cursor to (0, 0)
  Print "1. Operation Mode"
  Set cursor to (0, 1)
  Print "2. Ambient"
setUpperThreshold Function:
  Accept user input as parameter (key)
  If key is numeric ('0' to '9'):
    Update upperThreshold value
  Else If key is '#':
    Save upperThreshold to EEPROM
    Set currentState to SETUP
setLowerThreshold Function:
  Accept user input as parameter (key)
  If key is numeric ('0' to '9'):
    Update lowerThreshold value
  Else If key is '#':
    Save lowerThreshold to EEPROM
    Set currentState to SETUP
Fan Speed Control Logic:
  If temperature is within normal range:
    Set fan speed to minimum
    Display green LED
  Else If temperature is high:
    Gradually increase fan speed
    Display red LED
  Else If temperature exceeds maximum:
    Set fan speed to maximum
    Display red LED
  Else (temperature is low):
    Turn off fan
    Display blue LED
// Initialize pins, setup LCD and custom characters, etc.
Setup:
  ...
// Main program loop
Loop:
  ...
// Functions for handling user input, displaying menus, and controlling fan speed
Functions:
  ..
```

Results and Achievements:

The project successfully controls the fan speed based on temperature conditions. Users can navigate through the menu system to set upper and lower temperature thresholds. Real-time temperature readings are displayed on the LCD screen. The system provides visual indicators (LEDs) for temperature conditions. The custom degree Celsius symbol is displayed on the LCD.

Conclusion:

The Temperature-Based Fan Speed Control System effectively demonstrates the integration of temperature sensing, user interaction through a keypad, and real-time display using an LCD screen. While facing challenges in keypad and LCD integration, the project achieved its objectives of providing a functional fan speed control system.