- **1. Basic Breadboard Prototype**
- **Objective:** Test the core functionality of the system with essential components.
- **Components:** Temperature sensor, microcontroller (e.g., Arduino), motor driver IC, small DC motor (as fan), and display.
- **Setup:** Assemble components on a breadboard. Program the microcontroller to adjust fan speed based on temperature readings.
- **Outcome:** Validate basic temperature-to-fan-speed control, identify any immediate issues in circuit logic or component compatibility.

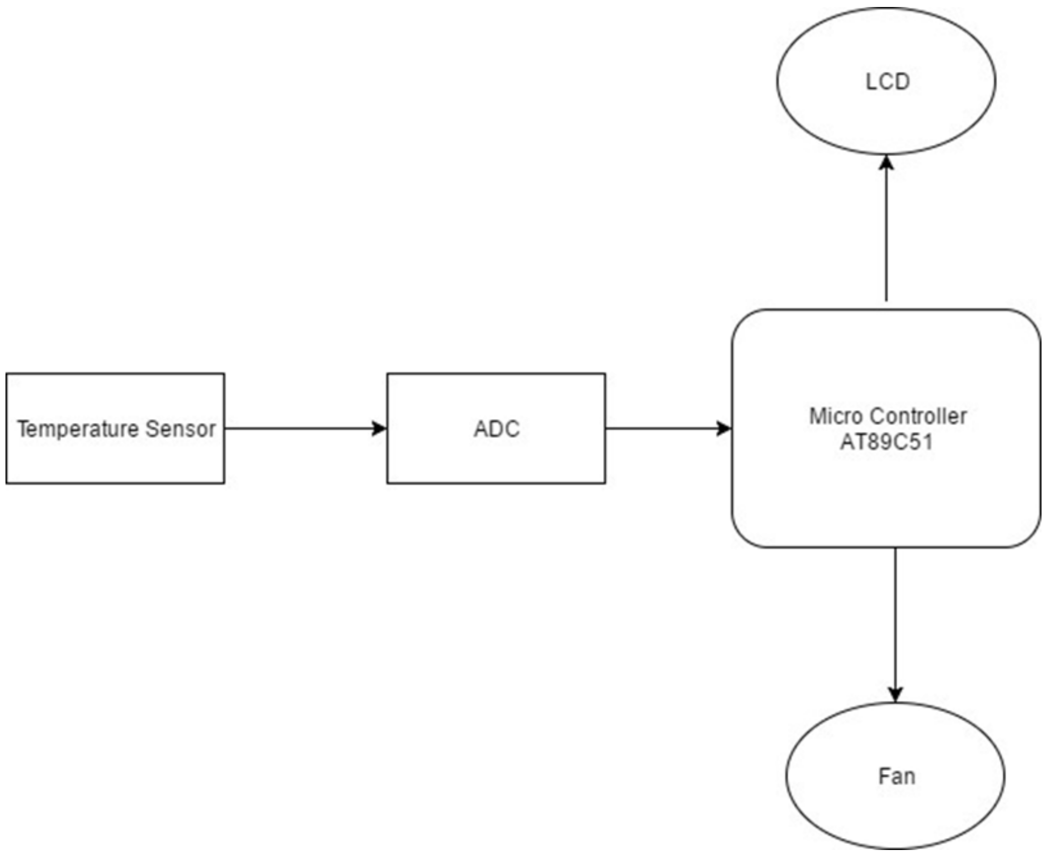
2. Perf Board/Custom PCB Prototype

- **Objective:** Transition from the breadboard to a more durable setup to improve stability.
- **Components:** Same as breadboard prototype but assembled on a perf board or custom PCB.
- **Setup:** Move components to a perf board or fabricate a custom PCB. Solder connections for durability.
- **Outcome:** Achieve a stable, portable prototype that reduces issues from loose connections and allows extended testing.

3. Final Functional Prototype with Casing

- **Objective:** Create a finished version with a protective casing to simulate a complete
- **Components:**
- **Custom PCB:** Houses all essential components (temperature sensor, microcontroller, motor driver IC, fan control circuitry).
- **Temperature Sensor:** Placed externally on the casing for accurate ambient readings.
- **Microcontroller:** Manages temperature readings, fan speed control, and display updates.
- **Motor Driver IC:** Controls the speed of the fan based on signals from the microcontroller.
- **Fan (DC Motor):** Provides the cooling function, adjusting speed according to ambient temperature.
- **Display Module (e.g., LCD):** Shows real-time temperature and fan speed for user feedback.
- **Power Supply:** External adapter or rechargeable battery for standalone functionality.
- **Protective Casing:** Heat-resistant, compact casing to hold and protect all components securely.
- **Setup:**
- Mount all components on the custom PCB and install them inside the casing.
- Attach the temperature sensor externally for accurate ambient measurement.
- Secure the display in a visible location on the casing for easy access.
- Connect to a reliable power source for continuous operation.
- **Outcomes:**
- A fully functional, user-ready prototype with accurate temperature-based fan control and real-time display.
- Durable, compact, and portable, ideal for extended testing or potential real-world use.
- Enhanced user experience with a polished appearance and user feedback display.

Working Principle

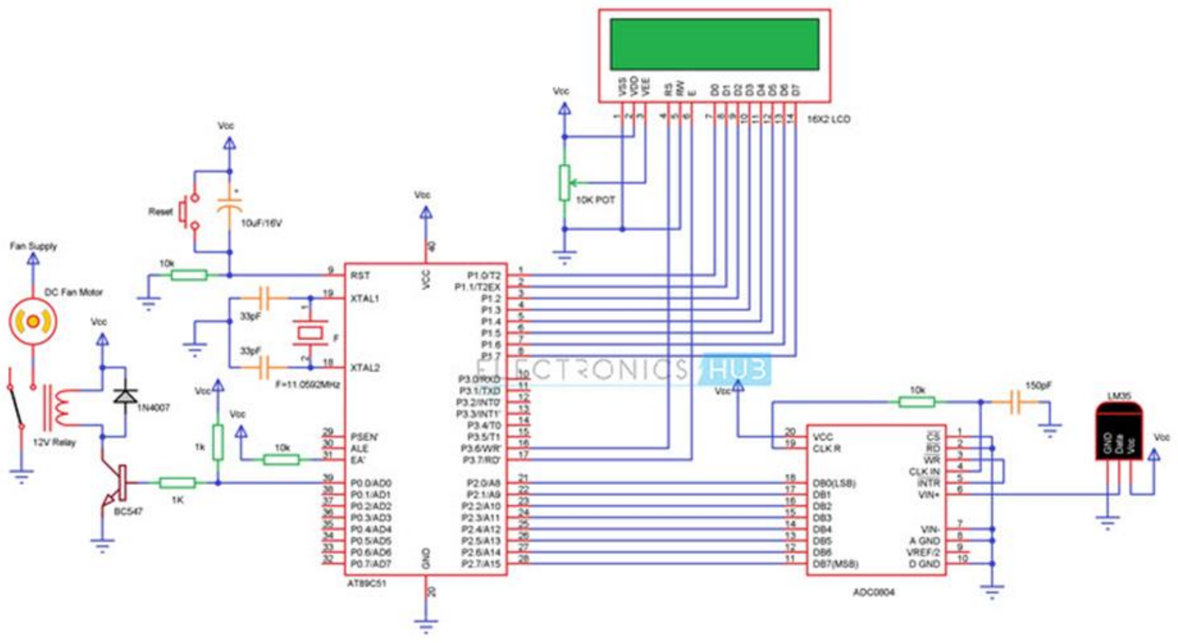


-
-

2. **Temperature Sensor (Input):** Reads ambient temperature and sends data to the microcontroller.
3. **Microcontroller (Processor):** Processes temperature data and sends appropriate control signals to adjust fan speed.
4. **Fan Speed Controller (Output):** Adjusts the fan speed based on signals from the microcontroller.
5. **Display Unit:** Shows temperature and fan speed for user feedback.

- **Process Flow:** The temperature sensor provides input to the microcontroller, which then processes this data and, based on preset temperature thresholds, controls the fan speed. This is displayed on the interface for the user.

<u>Circuit diagram:</u>



- The **LM35** sensor detects the ambient temperature and outputs an analog voltage proportional to the temperature.
- This analog voltage is converted into a digital signal by the **ADC0804**.
- The **8051 microcontroller** reads the digital temperature data, processes it, and compares it to preset temperature thresholds.
- Based on the temperature reading, the microcontroller sends a signal to the **relay driver circuit** (BC547 transistor and relay) to control the fan speed.
- The **LCD display** shows the real-time temperature for user reference.

Algorithm:

1 **Initialization**:

- Set Port 0 (P0) to high (0xFF).
- Configure Port 1, specifically set P1.5 to control an output.
- Initialize the LCD by sending commands for 8-bit mode, 2-line display, turning on the display, and clearing it.

2 **Write to LCD**:

- Display "MICROCONTROLLER" on the first line of the LCD.
- Move the cursor to the second line and display "PROJECT".
- Clear the display and then write "TEMP CONTROL FAN" on the LCD.

3 **Wait for User Input**:

- Monitor Port 0 (P0) to get user input (e.g., temperature or control signal).

- Store this input for further processing.

4 **Compare Input**:

- If the input is greater than or equal to 35:
    - Set the fan to high speed by calling the GREATER subroutine.
- If the input is between 25 and 35:
    - Set the fan to low speed by calling the LOWER subroutine.
- If the input is less than 25:
    - Turn off or maintain current fan speed.

5 **Temperature Conversion**:

- Convert the input value into two separate digits for the temperature (tens and units).
- Prepare these digits for display by adding 0x30 to convert them into ASCII format.

6 **Display Temperature on LCD**:

- Write the converted temperature digits to the LCD, followed by the letter "C" to indicate Celsius.

7 **Handle Timer Interrupt**:

- On each timer interrupt, toggle the state of P1.5 (e.g., for indicating the fan's status).
- Stop and restart the timer during the interrupt process to manage timing control.

8 **Loop**:

- The program continuously monitors the user input, updates the LCD display, and adjusts the fan speed based on the input value.

9 **End**

Limitations of the Project

- Limited temperature range adaptability of the sensor.
- System power consumption may be higher than anticipated without optimization.
- Lack of advanced predictive algorithms for proactive cooling.

Modern Engineering Tools, Techniques, and Resources for Improvement

- **Advanced Microcontrollers:** Consider using more efficient microcontrollers with built-in temperature sensing.
- **IoT Connectivity:** Use IoT modules to enable remote monitoring and control.
- **Machine Learning:** Implement predictive algorithms for better cooling based on usage patterns.
- **Renewable Energy Integration:** Add solar panels or other renewable sources for sustainable operation.

Alternative Design Solution

- Integrate a smart thermostat and IoT-enabled fan control system. This could allow users to adjust temperature settings from a smartphone app and monitor energy usage. Additional sensors could provide more precise environmental data for control.

Impact on Public Health, Safety, Society, and Environment

- **Public Health and Safety:** Automated cooling can prevent overheating, ensuring a safer and more comfortable environment.

- **Societal Impact:** Enhances the convenience and comfort of residential and public spaces, contributing to quality of life.
- **Environmental Impact:** By reducing unnecessary energy use, the project promotes environmental sustainability, which aligns with modern green initiatives.

Comment on the ability to function effectively as a team member

Division of Roles and Responsibilities: Assigning specific roles based on individual strengths (e.g., circuit design, coding, testing, documentation) can streamline the workflow. For example, one team member could focus on programming the microcontroller, while another handles hardware assembly and testing.

- **Communication and Coordination**: Regular meetings to discuss progress, address challenges, and exchange feedback will be essential. Sharing insights and clarifying technical details can prevent errors and align everyone toward the same goal.

- **Problem Solving and Innovation**: As a team, brainstorming alternative approaches to challenges (like optimizing fan control or improving temperature accuracy) can lead to innovative solutions that a single person might overlook.

- **Quality Assurance and Testing**: Collaboration in testing and debugging will ensure a more robust and reliable system. Team members can double-check each other's work, reducing the likelihood of missed issues.

- **Documentation and Presentation**: Dividing documentation tasks allows team members to capture details in areas they’ve worked on directly, ensuring a comprehensive and accurate project report.

Bill of Materials