ESP32-Based Heater Control System Design

1. Minimum Sensors Required

To detect temperature and control the heating process, the following minimum sensors are recommended:

- DHT22 Temperature Sensor
- Measures ambient temperature with ±0.5°C accuracy.
- Digital output makes interfacing simple with microcontrollers like ESP32.

- (Optional) Current Sensor (ACS712/INA219)
- Monitors the power consumption of the heater.
- Useful for protection and efficiency.

2. Recommended Communication Protocol

Protocol: UART (Serial Communication)

Justification:

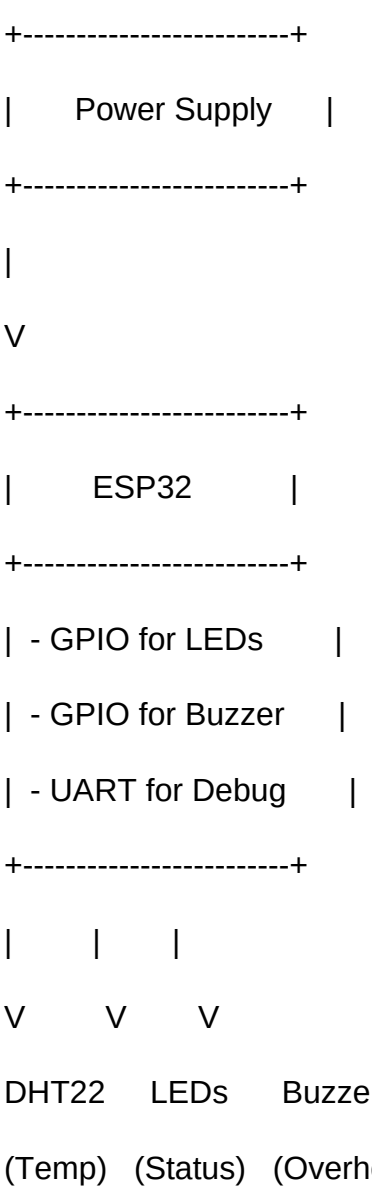
- Simplicity: UART is easy to implement and debug on ESP32.
- Compatibility: Ideal for logging and interfacing with PCs or other controllers.
- Speed: Sufficient for periodic temperature/status updates.
- Debug-friendly: Directly viewable via Serial Monitor or terminal.

Future Extension:

- BLE (Bluetooth Low Energy) for wireless control/status updates from a mobile app.

- I2C/SPI for connecting additional modules (like OLED, RTC, EEPROM).

3. Block Diagram (Text-based)



4. Future Roadmap

Overheat Protection:

- Set a maximum temperature limit (e.g., 40°C).

- Automatically cut off heater and trigger buzzer/LED alert.

- Add hardware relay for physical disconnection of heater in extreme cases.

Multiple Heating Profiles:

- Define temperature targets like:

- Profile A: 28-32°C (Default)

- Profile B: 32-35°C (Warm)

- Profile C: 35-38°C (Hot)

- Switch profiles using:

- Push button cycles

- BLE mobile app

- Web UI (via Wi-Fi)

Data Logging & Dashboard:

- Store temperature logs on SD card or cloud.

- Show graphs via web interface or mobile app using HTTP/BLE.

Smart Enhancements:

- Use PID control instead of threshold switching.

- Add humidity sensing for climate applications.

- Incorporate machine learning to adapt to user behavior.

Wokwi Simulation:

https://wokwi.com/projects/437516644786341889

Prepared by:

Jesu Raj J

Embedded Systems Intern

Date: 26 July 2025