Annealing Lehr : Temperature Controlled Production Oven

Arya Karimi Jafari, Muhammad Mustafa Qaiser, Omar Abdellatif [1]

Contents

1 Motivation **2**

2 System Requirements **3**

3 Hardware Documentation **4**

4 IoT Implementation **5**

5 Team Distribution **8**

6 Future Work **8**

7 Conclusion **9**

8 Declaration of Originality **9**

[1] arya.karimi-jafari@stud.hshl.de/muhammad.mustafa-qaiser@stud.hshl.de/omar.abdellatif@stud.hshl.de

Abstract: An annealing lehr is a tunnel shaped furnace that gradually cools glass products in order to relieve internal stress without introducing new ones. Precise controlled temperature gradients and airflow in each zone allow the glass to not deform or crack during the process. Embedding an Internet of Things (IoT) layer on top of the traditional control system provides continuous remote supervision and data optimization. This project presents a five zone Lehr prototype implemented on an Arduino WiFi Micro-controller. With 3 burner zones to provide the initial heat to the glass objects, a middle zone responsible for thermal neutrality, and a final end zone to handle the cooling and smoke extraction. The Burners, Fan and conveyor belt motor are found in a secure web resource, establishing real time adjustment from an MQTT based web browser.

1 Motivation

Glass objects such as bottles or panels emerging from the lehr carry hidden complications. Rapid or uneven cooling can trap stresses than later result in cracks, deformation, or complete failure. Our motivation was to demonstrate how a traditional five-zone annealing lehr with 3 heating zones, a neutral middle zone, and a cooling zone can be reimagined as a fully networked, data driven system using a simple prototype highlighting the remote and physical capabilities of the Annealing lehr. By representing the burners with RGB LEDs that either follow live temperature readings or respond to setpoints published on a bespoke MQTT server, we show how the operator can manually tweak thermal profiles in real time via the created web browser.

At the same time, the prototype also depicts how safety and material handling blend seamlessly with IoT intelligence, where the conveyor belt's DC motor obeys the speed input dispatched over MQTT, while the exhaust fan's motor is wired directly to a smoke sensor so it turns on instantly when smoke has been detected. This prototype focuses on portraying the general function of Annealing lehrs with the extension of a remote server which can greatly improve flexibility, transparency and safety.

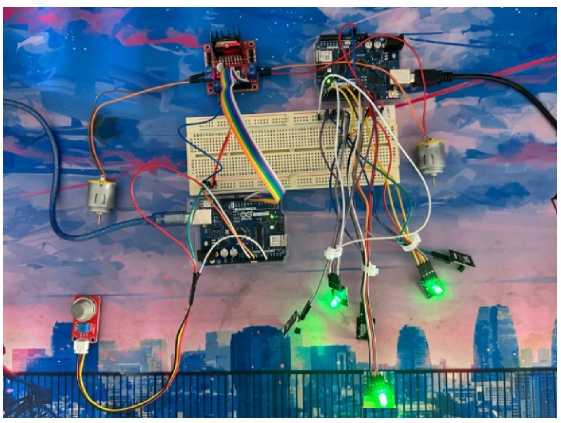

Fig. 1: Prototype Annealing Lehr

2 System Requirements

The system requirements are divided into two interconnected modules: The Physical Environment Control & Remote Web Server Control. In the Physical Environment Control section, once the prototype has been powered on, each temperature sensor continuously samples its zone and drives the corresponding RGB LED according to three distinct thresholds chosen for safe, low temperature demonstration. If the detected temperature is above 30 degrees C, then the LED will turn **Blue**, indicating that the Burner is now off. Readings between 20 and 30 degrees C, result in the RGB turning **Green** showcasing that the Burner is on moderating mode and is only maintaining the temperature. Finally, if the temperature detected is below 20 degrees C, then the RGB will turn **Red** showing that the Burner is now on and heating up the zone. A gas sensor in the 5th zone monitors gas levels, for removing any excess smoke that can be trapped within the Annealing lehr. The gas sensor is directly linked to a miniature DC motor which represents the exhaust fan, and if the gas measurement is greater than the assigned 100 threshold, then the motor will turn on and clear out the smoke. Although a real annealing lehr would assigned critical and calculated temperature setpoints for each zone, our prototype uses unifrom thresholds across all zones due to the limited thermal range which could be achievable with small flames and ambiant heating. When the user wishes to manually control the Burners, and machine belt speeds, disregarding the physical environment, the user can click on the "Web"button to pause sensor driven operation and assume manual command of each zone. In this mode, the user is able to control each individual zone's burners independently, controlling each RGB LEDs color, representing the different Burner modes via an MQTT enabled interface. Simultaneously, the machine belt's DC motor speed cam be set with a value ranging from 0-255, giving complete control over the belt throughput. This override capability is crucial in case a sensor fails, or processing a new product whose thermal profile falls outside the default ranges. Pressing the "Reset"button restores the system to the original Physical Environment Control logic, handing temperature regulation back to the local sensors and returning the lehr to its automated cooling cycle.

```
Sensor 1: 26.44 °C
Sensor 2: 25.87 °C
Sensor 3: 25.75 °C
Sensor 1: 26.44 °C
Sensor 2: 25.87 °C
Sensor 3: 25.75 °C
```

Fig. 2: Environment Control

```
Cmd: red1
Web: override LED 1 → red1
Cmd: green1
Web: override LED 1 → green1
Cmd: blue1
Web: override LED 1 → blue1
Cmd: red2
Web: override LED 2 → red2
Cmd: blue3
Web: override LED 3 → blue3
```

Fig. 3: Web Server Control

3 Hardware Documentation

The following section of the paper dives deeper into the components that were used in order to create this prototype Annealing Lehr.

- **2x Arduino Uno WiFi Rev2:** These boards were chosen for their simplicity and built-in WiFi/MQTT support, making it easy to publish sensor data to a network. Two units are used because a single Uno doesn't have enough I/O pins to handle all sensors and actuators.
- **3× DS18B20 Temperature Sensors:** Digital one-wire sensors that provide precise, two-decimal-place temperature readings and can be daisy-chained on a single data line for easy wiring.
- **3× Common-Cathode RGB LEDs:** Each LED represents one of the three burners in the prototype, with red/green/blue channels driven via PWM to indicate burner state (Red = Heating, Green = Moderating, Blue = Off)
- **2× Small DC Motors:** Low-voltage motors used as simple actuators to demonstrate directional and speed control in response to system events. One motor drives the conveyor belt that carries parts through the lehr, while the other powers the exhaust fan to regulate airflow and remove fumes.
- **MQ2 Gas Sensor:** An analog combustible gas sensor sensitive to LPG, propane, methane, and smoke, enabling basic air-quality or leak detection.
- **L298N Motor Driver Module:** A dual H-bridge driver that provides the necessary current and direction control to run the DC motors from the 5V Arduino outputs.
- **Breadboard & Jumper Wires:** Standard prototyping hardware used to interconnect all components without soldering, allowing quick revisions and troubleshooting.

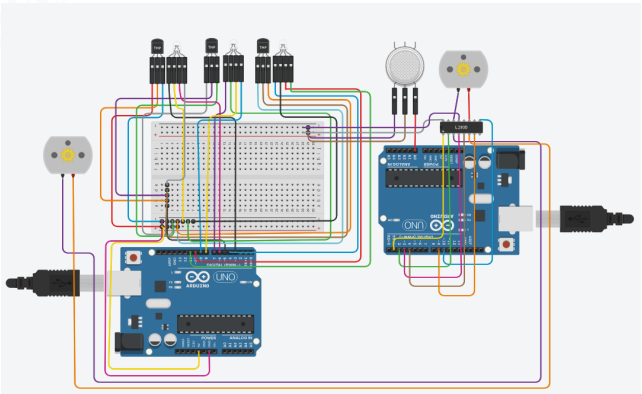

Fig. 4: Project Schematic

4 IoT Implementation

This section provides a detailed technical description of the IoT system architecture, focusing on the integration of the MQTT protocol, the configuration of the communication model, and the deployment strategy of the distributed components.

The IoT system comprises two microcontroller units (MCUs), specifically Arduino-compatible boards equipped with integrated Wi-Fi modules , and a web-based control interface developed using native **HTML**, **CSS**, and **JavaScript**. The system is designed to enable real-time bidirectional communication between the user interface and the embedded devices through the lightweight **MQTT (Message Queuing Telemetry Transport)** protocol.

MQTT is a low-overhead, TCP/IP-based publish-subscribe messaging protocol ideal for resource-constrained devices and unreliable or bandwidth-limited networks, which makes it well-suited for IoT deployments. It employs a central *message broker* to decouple message producers (*publishers*) from consumers (*subscribers*), facilitating a scalable and modular communication infrastructure.

Communication Model

The system utilizes a public MQTT broker (`broker.hivemq.com`) hosted in the cloud. The architecture is defined by the following components:

- **Publisher:** The custom web application running on the client browser acts as the MQTT publisher. It establishes a connection to the broker over WebSocket transport and publishes control messages to a predefined topic (e.g., `/control/burners`).
- **Subscribers:** The two Arduino boards each act as independent MQTT subscribers. Each device connects to the broker via Wi-Fi and subscribes to one or more topics, depending on its assigned functionality.
- **Broker:** The public MQTT broker is responsible for routing messages from publishers to all subscribed clients. It handles session persistence, Quality of Service (QoS) levels, and topic-based filtering.

Device Independence and Parallelism

An important aspect of the design is the logical and physical decoupling of the two Arduino boards. There is no direct inter-device communication between the Arduinos; instead, each device operates as an autonomous subscriber within the MQTT infrastructure. The broker

is the sole point of coordination, ensuring that both devices can operate in parallel without requiring explicit synchronization mechanisms.

The web application sends identical or topic-specific payloads to the MQTT broker. Each Arduino processes incoming messages independently and performs actions such as:

- Activating or deactivating actuators (e.g., motors, burners).
- Updating the state of RGB LEDs configured as visual status indicators.
- Logging or forwarding system state for debugging or monitoring purposes.

Each Arduino executes its logic according to its firmware, which is designed to filter incoming MQTT messages, parse JSON-formatted payloads, and trigger appropriate GPIO-level outputs.

Control Flow and System Synchronization

The control logic in the web application consists of UI elements that allow users to send predefined commands (e.g., “start burner”, “toggle motor”, “reset status”) through MQTT ‘PUBLISH‘ messages. These are dispatched to specific MQTT topics, which the Arduino boards are subscribed to.

The messages use structured payloads (e.g., JSON) to encapsulate control parameters:

```
Topic: /control/burners
Payload: { "burnerID": 1, "status": "on" }
```

Upon receiving a message, each subscriber performs the following steps:

1. Validate message integrity and topic relevance.
2. Parse the JSON payload.
3. Execute device-specific logic depending on the payload content.
4. Update the physical output (e.g., PWM for motor speed or RGB LED color).

This design allows for the dynamic addition of further devices, topic partitioning for load balancing, and potential integration of a feedback channel for monitoring or acknowledgment messages.

Advantages and Scalability

The MQTT-based architecture offers several advantages:

- **Scalability:** New subscribers (e.g., sensors, actuators) can be added without modifying the publisher logic.
- **Asynchronous Communication:** The pub/sub model decouples communication timing between sender and receiver.
- **Low Bandwidth Consumption:** MQTT's binary message format and minimal header overhead reduce network usage.
- **Real-Time Performance:** Rapid delivery of control signals allows near real-time control over embedded devices.

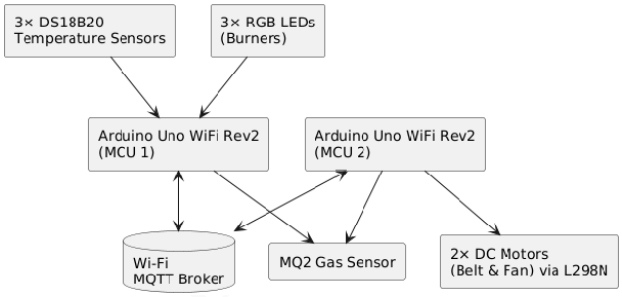

Fig. 5: System Flow

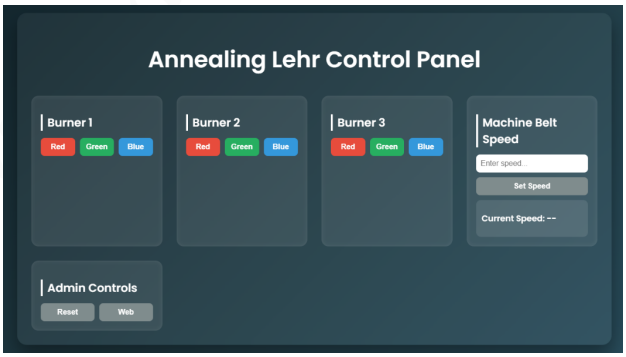

Fig. 6: MQTT Communication Setup

In summary, the integration of MQTT in this system provides a robust, scalable, and efficient method for enabling communication between distributed microcontrollers and a web-based control panel. It demonstrates how modern IoT architectures can leverage cloud-based messaging systems to coordinate independent devices in a synchronized and responsive manner.

5 Team Distribution

Tasks were divided equally among the team members, with each group member's task being describe in the table provided below.

Team Member	Matriculation Nr.	Tasks
Arya Karimi Jafari	2219954	-Concept Designing & Schematic planning. -Temperature Sensors and LED wiring and program writing for Physical Environment.
Muhammad Mustafa Qaiser	222058	-Wiring & connecting DC motors with L298N motor driver & Gas sensor. -Programming Gas sensor and DC motor1 synchronization for Physical Environment.
Omaer Abdellatif	2219957	-Designing & Programming MQTT Web Server. -Implementing MQTT connection from server to all integrated sensors.

Tab. 1: Team members, matriculation numbers and assigned tasks

6 Future Work

Building on the current Annealing lehr prototype, several enhancements could be pursued to create a stronger resemblance to the real life industrial machine system:

- **User Authentication & Access Control:** By introducing a secure login system for the MQTT web interface so that only authorized operators can issue override commands or change setpoints. In addition, a role-based permission system can be added to restrict critical actions to more qualified personnel.
- **Data Logging & Analytics:** By storing every temperature reading, LED state change, and motor action into a database, higher optimization and efficiency can be achieved.
- **Expanded Sensing & Safety:** Incorporating additional sensors such as humidity, CO/CO_2 detectors for more accurate zone measurement. As well as adding hardware interlocks and emergency stop buttons to ensure ultimate safety.

7 Conclusion

In this paper, we have presented the design and implementation of an Arduino-based annealing-lehr prototype, covering our hardware selection (dual Uno WiFi boards, DS18B20 sensors, RGB LEDs, DC motors, gas sensor and motor driver), the key software components in regards to the implementation of an MQTT-enabled web server. Hardware documentation, and general sematic illustrate how each subsystem interconnects to deliver reliable, real-time temperature regulation and visual feedback. Throughout our testing, the system operated continuously and without failure, demonstrating stable temperature readings, correct LED color changes, and consistent motor control. Looking ahead, we plan to introduce secure user authentication, implement comprehensive data logging and analytics, upgrade our sensing capabilities and security through an emergency stop feature, further enhancing both the functionality and resilience of the prototype.

Fig. 7: Annealing Lehr

8 Declaration of Originality

We, Arya Karimi Jafari, Muhammad Mustafa Qaiser, and Omar Abdellatif, herewith declare that we have composed the present paper and work by ourselves and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker.

18-07-2025 - Arya Karimi Jafari, Muhammad Mustafa Qaiser, Omar Abdellatif