Design and Implementation of an IoT-Integrated Automated Beverage Dispensing Table

Max Kotas

December 19, 2024

Abstract

This paper presents the comprehensive design, fabrication, and operational methodology of an automated beverage dispensing table, integrating Internet-of-Things (IoT) technologies with precision mechanical and electrical components. Leveraging MQTT-based communication, voice assistant integration, stepper motor-driven linear motion, peristaltic pumping, and a custom electronics stack, this project exemplifies a holistic approach to smart home automation. The system converts voice commands into actionable control signals, utilizes a linear actuator to position a cup beneath dispensing nozzles, and accurately measures and delivers liquid volumes from multiple reservoirs. The following sections detail the engineering challenges, component selections, fabrication processes, and control software. Furthermore, this paper discusses the transition from a Raspberry Pi-based prototype to an ESP32-centric architecture and the preliminary design of a dedicated PCB that consolidates wiring and improves reliability. Insights into future improvements and advanced calibration techniques are also presented.

1 Introduction

The growing interest in smart home technologies and IoT-driven convenience has stimulated the development of automated beverage dispensing systems. While such concepts have been explored in commercial bar environments and robotics demonstrations, the present work focuses

on a home-friendly, furniture-grade implementation. The primary motivation behind this project was to bridge the gap between typical demonstration-scale IoT experiments and a genuinely useful household device. By integrating voice commands from widely used digital assistants (e.g., Amazon Alexa) into a mechanical and electrical infrastructure, the system provides both educational value for engineers and tangible utility for end-users.

From a pedagogical perspective, the project offered a controlled environment to study motor control systems, real-time signal processing, MQTT communication, and the interaction between mechanical design and electronic instrumentation. In practical terms, the table transforms user intent—expressed verbally—into a precise set of mechanical actions, delivering a ready-to-consume beverage. This paradigm elevates the typical 'smart home' narrative to include not only environmental controls (lights, temperature, security) but also interactive culinary and social experiences.

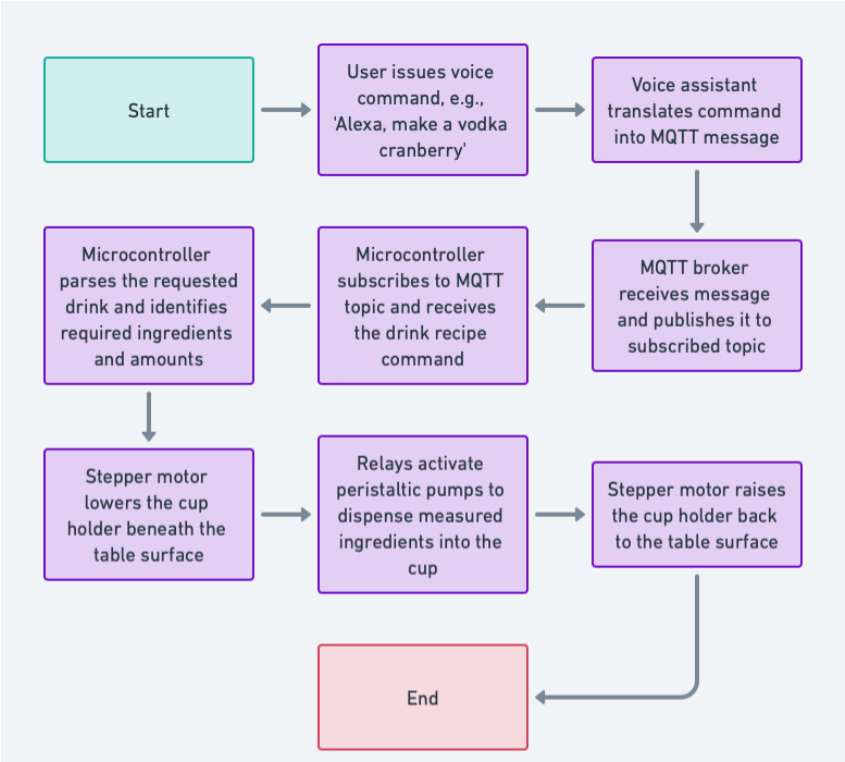

Figure 1: High-level schematic of the automated beverage dispensing table. The system integrates MQTT-based communication, a stepper-driven linear actuator, and peristaltic pumps.

At a high level, the system seamlessly combines voice commands, network communication, and precise actuation to produce a fully mixed drink. To illustrate this more concretely, Figure 2 presents a collage demonstrating the different states of the device in operation. From left to

right (or top to bottom), the sequence shows:

(a) Cup at the table surface, empty. (b) Cup lowered beneath the table. (c) Cup raised with dispensed drink. (d) Final view showing the prepared drink.

Figure 2: A collage of the automated beverage dispensing table in operation. The sequence shows the cup starting empty at the surface, lowering for ingredient dispensing, returning to the top with a measured portion of cranberry juice, and finally presenting a fully prepared beverage to the user. (Placeholder Images)

2 Core Components and Architecture

The system consists of several interdependent components, carefully selected for accuracy, reliability, and ease of integration. Each subsystem contributes to the overarching functionality, from receiving and interpreting user commands to delivering precise quantities of liquids.

2.1 Control Electronics

Initially, the system's logic was driven by a Raspberry Pi Zero 2W, chosen for its compact form factor, robust Wi-Fi connectivity, and accessible GPIO. The Raspberry Pi subscribes to an MQTT broker, receiving structured JSON messages triggered by voice commands. Future iterations integrate an ESP32 microcontroller onto a custom PCB for improved electrical noise immunity, simplified wiring, and potentially lower latency I/O operations.

2.2 Linear Actuation and Stepper Motor

Vertical motion is achieved using a NEMA17 stepper motor coupled to a lead screw mechanism. This arrangement ensures highly repeatable and smooth linear translation of the cup holder. A DRV8825 stepper driver module, known for its microstepping capabilities and ease of interface,

manages the motor's current and stepping signals. The mechanical components—an aluminum extrusion frame, linear bearings, and custom 3D-printed mounts—guarantee stable, low-friction movement.

Figure 3: The Early NEMA17 stepper motor and lead screw assembly working in a test run. 3D-printed brackets ensure correct alignment and rigidity.

2.3 Dispensing System and Peristaltic Pumps

This system uses 12V DC motors driving peristaltic pump heads. The volumetric flow rate Q (ml/s) relates to the motor's rotational frequency f (Hz) and the volume displaced per rotation V_r (ml/rev):

$$Q = f \cdot V_r.$$

For a motor speed of RPM = 3000:

$$f = \frac{3000}{60} = 50\,\text{Hz}.$$

If the measured flow rate is $Q = 33$ ml/s, then:

$$V_r = \frac{Q}{f} = \frac{33\,\text{ml/s}}{50\,\text{s}^{-1}} = 0.66\,\text{ml/rev}.$$

Over a dispensing interval t (s), the total volume is:

$$V(t) = Q \cdot t.$$

These relationships allow precise time-based control of the dispensed volume.

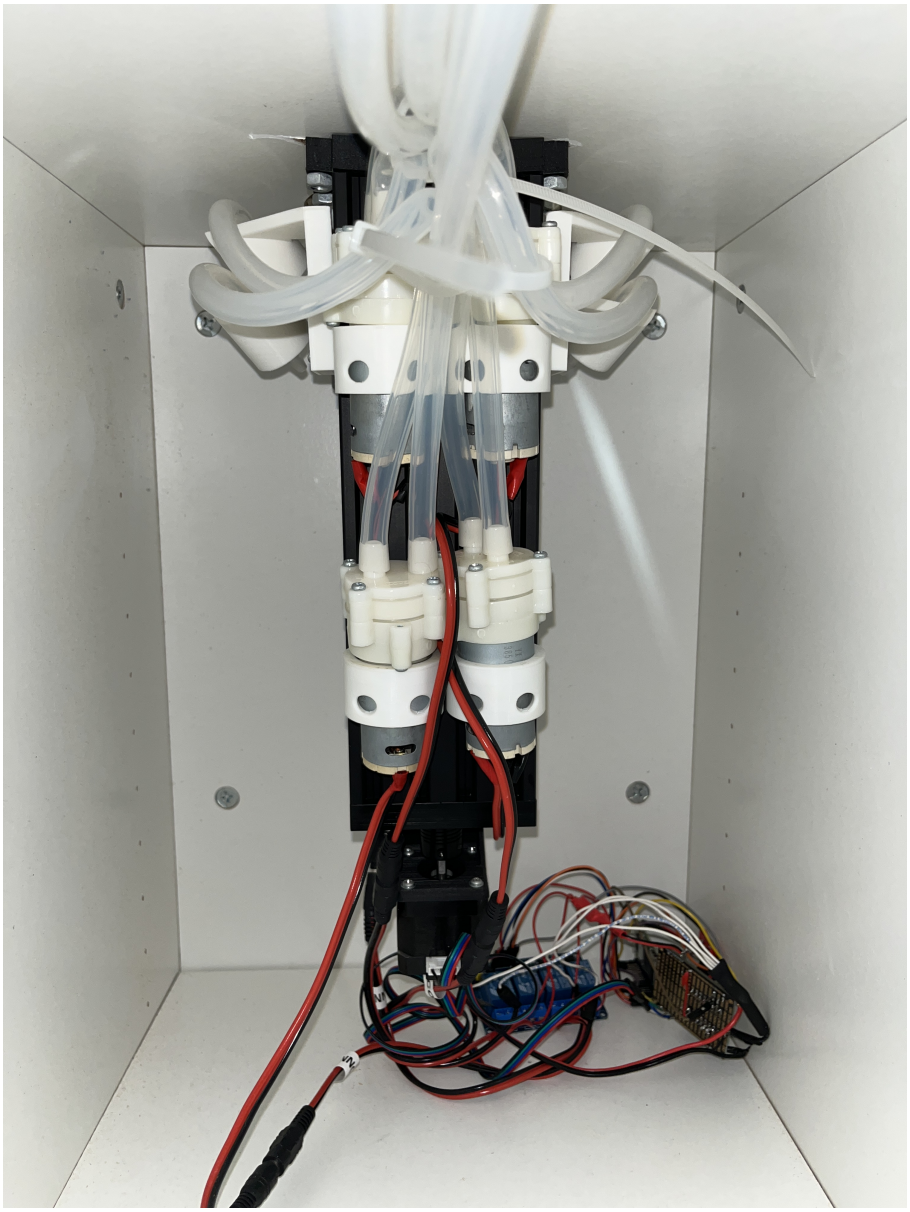

Figure 4: Peristaltic pump assembly with food-grade silicone tubing and check valves.

2.4 3D-Printed Structural Components

Numerous custom 3D-printed parts hold the system together, including motor mounts, pump brackets, valve holders, and the cup holder platform. Iterative design and prototyping allowed for optimal tolerances, accessibility for fasteners, and clearance for wiring. Each iteration incorporated feedback from mechanical testing, leading to robust parts that could withstand operational stresses and environmental conditions within a residential setting.

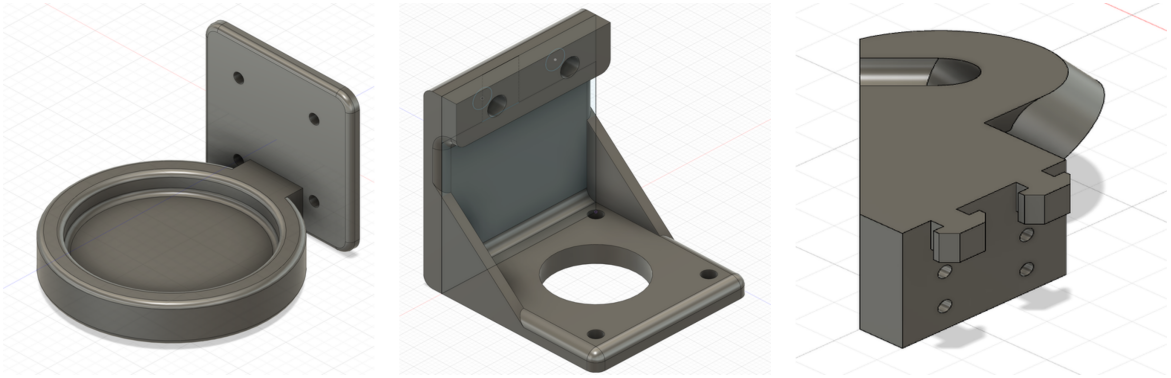

Figure 5: A selection of 3D-printed mounting brackets and fixtures that support motors, pumps, and tubing.

3 PCB Design and Schematic Integration

To minimize wiring complexity and increase system reliability, a dedicated PCB was designed to consolidate the ESP32 controller, stepper motor driver, power regulation circuitry, and connector interfaces into a single, compact board. The PCB design phase began after preliminary testing with a Raspberry Pi Zero 2W and loose wiring confirmed the feasibility of the concept. By transitioning to an ESP32-based PCB, the objective is to achieve a cleaner assembly, improved signal integrity, and more straightforward scaling or modification of the system.

A key starting point in this process was developing a clear, annotated schematic to capture the system's functional requirements and ensure that all components interact correctly.

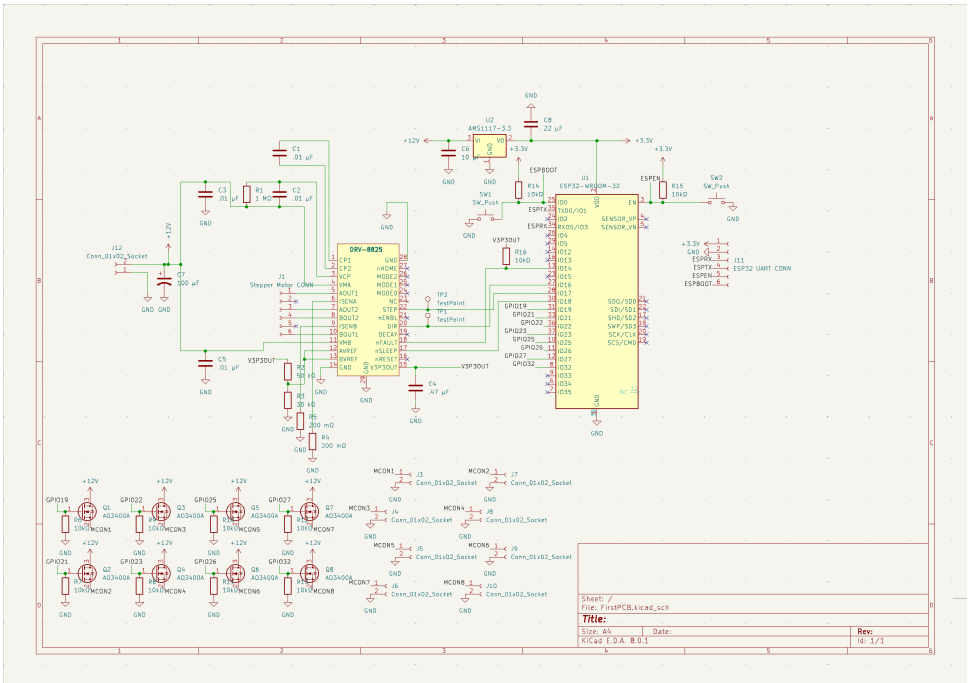

Figure 6: Portion of the project's schematic showing the ESP32-WROOM-32 module, the DRV8825 stepper motor driver, an AMS1117-3.3 voltage regulator, and MOSFET-based outputs for pump control. The schematic clearly distinguishes power domains and includes test points for easier troubleshooting.

As illustrated in Figure 6, the schematic details the core functional blocks:

- **ESP32-WROOM-32 Module:** Serves as the main microcontroller with integrated Wi-Fi and Bluetooth, providing MQTT connectivity and real-time control over peripherals.
- **DRV8825 Stepper Driver:** Dedicated pins for direction, step, enable, and microstepping configuration facilitate precise control of the NEMA17 motor via simple GPIO signals.
- **On-Board Voltage Regulation:** An AMS1117-3.3 (or equivalent) regulator provides the stable 3.3 V required by the ESP32 and logic-level signals. Bulk capacitors and decoupling capacitors are strategically placed to reduce noise.
- **Pump Control Stages:** Transistor-based output channels (shown as MOSFET placeholders) drive the 12 V peristaltic pumps. These are routed to connectors along the PCB's edges for easy attachment of the tubing and pump assemblies.

Once the schematic was finalized, careful consideration went into component placement and PCB layout. High-current traces for the stepper motor and pumps were routed to minimize

voltage drops and electromagnetic interference. Sensitive analog or high-speed signals (e.g., communication lines) were kept short or isolated from noisy sections. The antenna keep-out zone required by the ESP32 module's datasheet was also respected to ensure stable wireless performance.

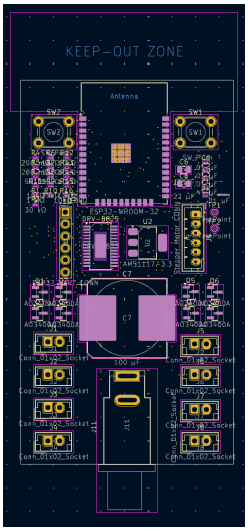

Figure 7: Top-down PCB layout view. The ESP32 module occupies the top-central area, with the DRV8825 driver, connectors for pumps, and test points arranged strategically. Bulk capacitors, decoupling capacitors, and voltage regulators are placed close to their respective loads to maintain signal integrity.

Figure 7 showcases the board's top-down layout. Notice how the ESP32 module sits horizontally centrally, reducing the length of control signal traces to the stepper driver and pump outputs. Connectors are grouped for related functionalities, and test points are included for in-circuit measurements. The antenna keep-out region ensures that metal components and copper pours do not interfere with wireless transmissions. Proper PCB grounding techniques, ground planes, and careful routing reduce noise, improving both stepper motor performance and sensor readings.

To further validate the mechanical fit, component heights, and general assembly considerations, a 3D model of the PCB was generated. This helps visualize the final product, ensuring that connectors are accessible, the largest components (e.g., electrolytic capacitors, stepper connectors) will not obstruct each other, and any enclosure design can accommodate the board's form factor.

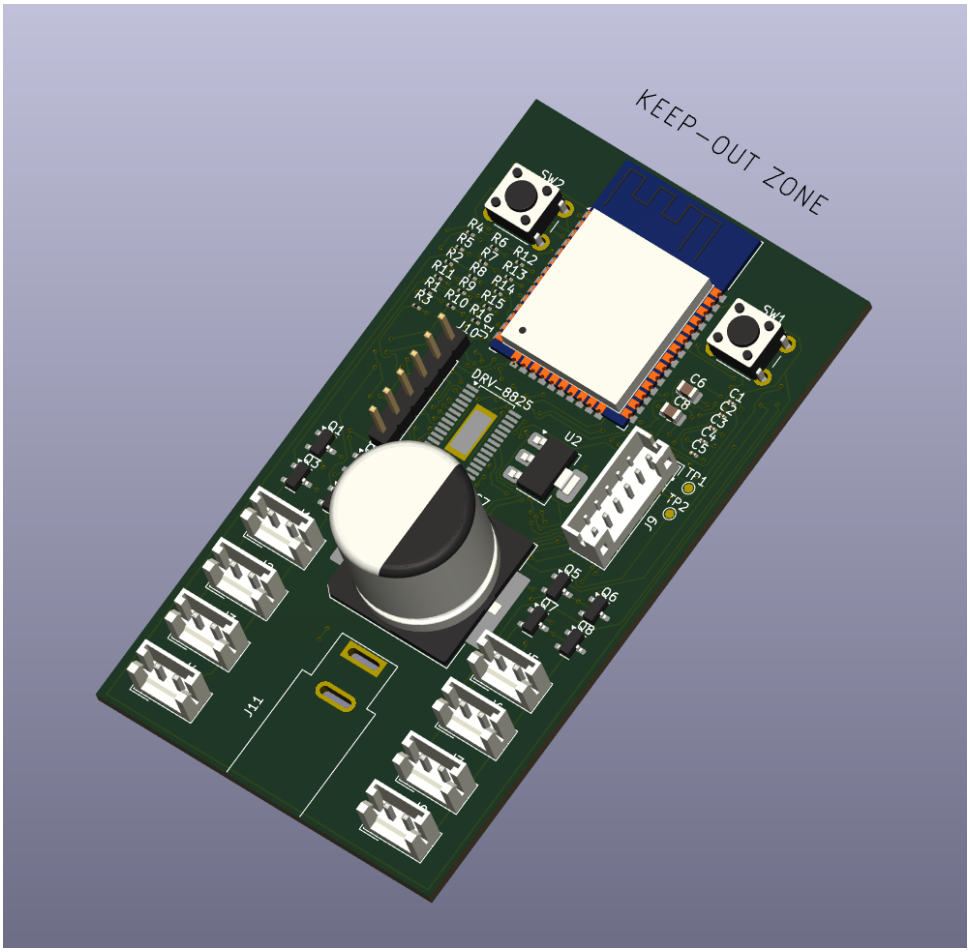

Figure 8: 3D rendering of the assembled PCB. This perspective reveals component heights, connector orientations, button placements, and the overall form factor. It aids in early mechanical verification before ordering physical boards.

Figure 8 provides an intuitive, near-photorealistic representation of the final PCB assembly. The visualization ensures that any mechanical interference issues, such as the placement of a large capacitor overshadowing a pump connector or the orientation of programming buttons, are addressed prior to fabrication. Additionally, visual inspection in 3D reduces surprises during final assembly and integration with the table's mechanical frame.

In summary, the PCB design and schematic integration steps mark a transition from a working prototype to a refined, manufacturable device. The careful schematic planning, methodical PCB layout, and 3D modeling contribute to a robust, serviceable platform for the automated beverage dispensing system. Once fabricated, the PCB can be tested and refined, further streamlining the entire assembly and enhancing the system's overall reliability.

4 Build Process and Assembly

The build process involved multiple stages of prototyping, iterative refinement, and integration testing before arriving at a functional system ready for demonstration and eventual transition to a custom PCB. Initially, development centered on proving core concepts using off-the-shelf boards, rough mechanical assemblies, and basic test scripts. As confidence grew in the underlying principles—voice command interpretation, MQTT message passing, stepper control, and pump activation—focus shifted toward refining the mechanical and structural aspects of the design.

4.1 Iterative 3D-Printed Parts

A pivotal aspect of the build process was the iterative design and refinement of numerous 3D-printed components. Early attempts at fabricating the cup holder, pump mounts, and valve holders often revealed unforeseen issues. For example:

- **Cup Holder Geometry:** The initial cup holder design did not account for the vertical clearance required when retracting the cup fully below the table surface. Early prototypes collided with stationary frame members or protruded above the table line. Subsequent iterations reduced the holder's height profile, adjusted its supporting arms, and ensured that when the linear actuator retracted, the cup and holder remained fully concealed.

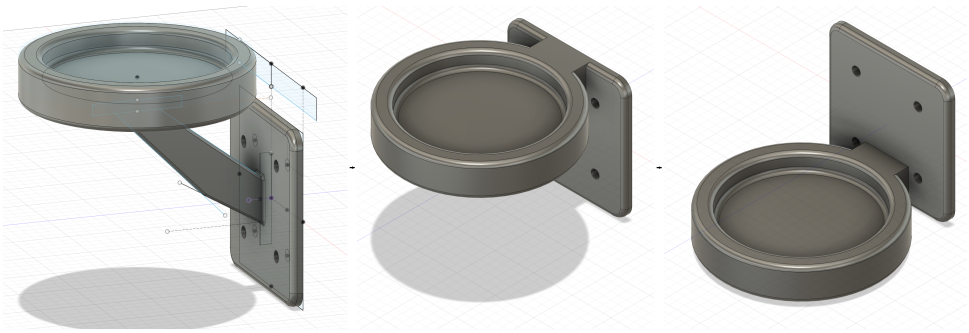

Figure 9: A collage of cup holder prototypes from early bulky designs (left) to more streamlined, low-profile versions (right). Adjusted dimensions and improved arm geometry ensured smooth retraction beneath the tabletop.

- **Spout Holder and Tubing Guides:** Aligning the tubing so that liquids dispensed accurately into the cup proved challenging. Initial valve holders misaligned the output

streams, causing spills or uneven mixing. Iterations introduced angular adjustments, one-way valve mounts, and contoured channels to guide silicone tubing securely. This evolution transformed a bulky halo of nozzles into streamlined, individually adjustable holders that precisely directed each ingredient's flow.

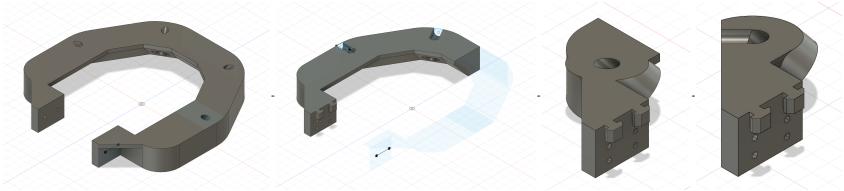

Figure 10: A collage showing the evolution of the spout holder from a single-piece halo design (left) to modular, individually adjustable holders (right). Iterative testing improved alignment, reduced leakage, and simplified maintenance.

- **Stepper Motor Bracket Refinements:** Ensuring the stepper motor remained rigidly mounted was crucial for consistent, precise vertical motion. Early bracket designs flexed or interfered with the lead screw assembly. Subsequent iterations introduced thicker walls, strategically placed support ribs, and cutouts that accommodated the motor's bearing lip. Adjusting tolerances and adding space for tools to tighten T-nuts and screws made installation and maintenance far easier. Over time, the bracket evolved into a robust component capable of maintaining proper alignment and load distribution.

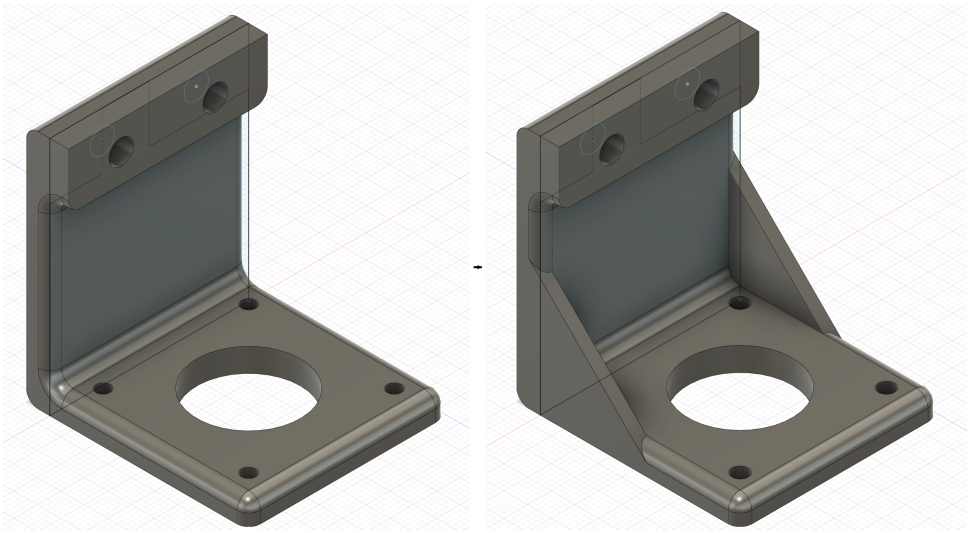

Figure 11: A collage illustrating the progression of stepper motor bracket designs. Early versions (left) lacked rigidity and tool access, while later models (right) featured reinforced walls, integral support ribs, and strategically placed openings for secure mounting and easy adjustments.

Printing these parts and testing them in situ revealed the inherent trial-and-error nature of mechanical prototyping. Slight dimensional inaccuracies, unexpected warping, or print bed size limitations required on-the-fly design changes. Over time, careful tolerance adjustments, refined support strategies, and considered material selection (e.g., PLA for quick iteration, PETG for improved strength) resulted in components that were both robust and functional.

This iterative approach to mechanical design not only improved individual parts but also ensured that the entire mechanical assembly became more cohesive and reliable. By iterating rapidly and incorporating feedback from each test, the final design achieved a balance between complexity, functionality, and manufacturability.

Printing these parts and testing them in situ revealed the inherent trial-and-error nature of mechanical prototyping. Slight dimensional inaccuracies in prints, unexpected warping, or limitations in available print bed size required on-the-fly design changes. Over time, careful tolerance adjustments, support strategies, and material selection (e.g., PLA for quick iteration, PETG for higher durability) improved the final printed parts' quality and reliability.

4.2 Difficulties and Solutions in Integration

Beyond 3D printing challenges, other integration steps posed unique difficulties:

- **Wiring Complexity:** Initially, a tangle of jumper wires connected the Raspberry Pi, stepper driver, relays, and pumps. This complicated troubleshooting and risked intermittent connections. Color-coded wiring schemes and bundling into organized looms helped manage this complexity, though it ultimately inspired the move toward a custom PCB.

- **Stepper Motor Tuning:** Achieving reliable vertical motion for the cup holder required adjusting stepper driver current limits, selecting appropriate microstepping modes, and fine-tuning acceleration parameters. Early tests produced jittery motion or occasional missed steps, leading to repeated trial runs to find stable settings. Using an oscilloscope and experimenting with motor current and step pulse timing resulted in smoother, more consistent vertical travel.

- **Pump Calibration and Leak Prevention:** Since the dispensing process relies on timed

pump operation, accurately mapping pump run time to liquid volume proved challenging. Early tests revealed that viscosity differences, tubing length variations, and minor suction losses caused inconsistencies in output. Logging test data, adjusting timings empirically, and adding one-way check valves to prevent backflow gradually honed the calibration process.

- **Structural Rigidity and End-Stop Considerations:** The linear actuator's lead screw mechanism relies on stable anchor points. Early brackets flexed slightly under load, introducing positional error. Reinforcing critical mounting points, thickening printed walls, and adding mechanical end-stops (or limit switches in future revisions) improved positional repeatability and system safety.

4.3 Towards a Production-Ready Assembly

By meticulously documenting each iteration and refining designs after every setback, the project evolved from a loose collection of prototypes into a cohesive, reliable system. Each printed part, wiring harness, and driver setting contributed to a growing body of knowledge. Lessons learned from each failed print, wiring snag, or unstable bracket guided the project toward an increasingly elegant and maintainable final design.

This iterative approach ensured that by the time the system neared a more production-ready state, many of the fundamental mechanical and integration challenges had been addressed. The forthcoming incorporation of a custom PCB will streamline the wiring even further, consolidating insights from these iterative stages into a polished, user-friendly device.

5 System Operation

At the highest level, the system operates as follows:

1. **Voice Command:** A user issues a command, such as "Alexa, make a vodka cranberry." The voice assistant translates the request into a structured message and publishes it to an MQTT topic.

2. **MQTT Reception:** The microcontroller, subscribed to the relevant MQTT topic, receives the command payload. The payload includes the drink recipe data—essentially a set of ingredients and proportions.

3. **Cup Lowering:** The stepper motor driver energizes the motor to lower the cup holder beneath the table's surface, positioning it under the dispensing nozzles.

4. **Dispensing Sequence:** The microcontroller calculates each ingredient's pump run time from calibrated flow rates. It activates the corresponding relay or transistor switch, allowing the peristaltic pump to dispense a measured volume of liquid into the cup.

5. **Cup Raising:** Once all ingredients are dispensed, the stepper motor reverses direction, raising the cup holder back to the table surface for retrieval.

6 Future Improvements and Calibration Strategies

Although the current prototype demonstrates a functioning concept, several avenues remain for refinement:

- **Closed-Loop Flow Control:** Employing flow sensors or integrating load cells beneath the cup could provide feedback for more accurate volumetric control. Closed-loop PID systems could then adjust pump run times in real-time.

- **Safety and Error Handling:** Incorporating limit switches or optical sensors on the linear actuator would ensure the system halts if the cup holder encounters an obstruction. Additional checks, such as a pressure sensor, could prevent overflow or detect missing cups.

- **PCB Iteration and Testing:** After the first PCB prototype is fabricated, thorough testing will identify potential issues in layout, grounding, or EMI suppression. Subsequent PCB revisions may include improved connectors, decoupling strategies, or integrated sensors.

- **Enhanced Recipe Management:** Storing recipes in a local database or querying a cloud-based recipe service could introduce dynamic recipe adjustments and more complex

mixing profiles.

7 Conclusion

This work outlines the development of an IoT-integrated, automated beverage dispensing table, from conceptual design through prototype assembly and operational validation. By combining voice commands, MQTT messaging, stepper-driven linear motion, and peristaltic pump-based fluid handling, the system presents a compelling case study in smart home automation applied to a culinary setting.

While the prototype is fully functional, the proposed PCB design and future calibration techniques promise a more robust, reliable, and user-friendly solution. Overall, this project serves as a foundation for future research and development in automated food and beverage preparation, potentially informing commercial products, hospitality applications, and advanced home-automation solutions.

References

- **Microcontrollers and Boards:**
  - Raspberry Pi Zero 2W: `https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/`
  - ESP32-WROOM-32 Module: `https://www.espressif.com/en/products/socs/esp32/overview`
- **Motor and Driver Components:**
  - NEMA17 Stepper Motor (Generic Specifications): `https://www.omc-stepperonline.com/nema-17-stepper-motor`
  - DRV8825 Stepper Driver Datasheet: `http://www.ti.com/lit/ds/symlink/drv8825.pdf`
- **Peristaltic Pumps and Valves:**

  - 12V DC Peristaltic Pump (Generic Specifications): https://www.adafruit.com/product/1150 (Example link)
  - Food-Grade Silicone Tubing (Manufacturer Specifications): https://www.mcmaster.com/silicone-tubing/
  - One-Way Check Valves (Generic Inline Check Valve Specs): https://www.mcmaster.com/check-valves/

- **Power and Regulation:**
  - AMS1117-3.3 Regulator Datasheet: https://www.advanced-monolithic.com/pdf/ds1117.pdf
  - Buck Converter Module (MP1584EN-based): https://www.digikey.com/en/products/detail/monolithic-power-systems-inc/MP1584EN/

- **MQTT and Software Libraries:**
  - Paho MQTT Client Library for Python: https://www.eclipse.org/paho/
  - umqtt.simple for MicroPython (ESP32): https://docs.micropython.org/en/latest/library/umqtt.simple.html
  - RPi.GPIO Library for Raspberry Pi: https://pypi.org/project/RPi.GPIO/

- **3D Printing and CAD Resources:**
  - Fusion 360 (CAD Software) Documentation: https://knowledge.autodesk.com/support/fusion-360
  - PrusaSlicer Documentation: https://help.prusa3d.com/category/prusaslicer_204

- **General Electronics Reference:**
  - Arduino GPIO and PWM Examples (Conceptual Reference): https://www.arduino.cc/en/Reference/HomePage

- Raspberry Pi Documentation on GPIO: https://www.raspberrypi.com/documentation/computers/configuration.html#gpio