Voice Activated Geographic Reference Globe

By

Team 44
Mahathi Jayaraman
Varsha Mullangi
Rijul Roy

Final Report for ECE 445, Senior Design, Spring 2026
TA: Chihun Song
Professor: Joohyung Kim
May 6, 2026

Abstract

This document will detail the purpose, design, verification, and results of the Voice Activated Geographic Reference Globe. The Voice Activated Geographic Reference Globe is a mechanical device that accepts user input of a country and then automatically rotates the globe to bring that country to the forefront, after which a laser will activate and highlight the country. This product is intended to be used by children, either in classrooms or at home to help them improve their knowledge of geography. At the end of the Spring 2026 semester, this project was successfully completed and currently functions as intended.

Table of Contents

1. Introduction

1.1 Problem

It is a widely known fact that American children and adults struggle with geography. Many of them cannot point out countries on a map or identify the locations of major world events. This lack of geographical knowledge can limit people's global awareness and make it more difficult for them to understand global culture, current events, and international relations. We believe that this gap in knowledge starts at school, where students are not fully engaged. When geography is taught through memorization instead of connection, students may not retain the information in the long term. In addition, as learning becomes more and more digital, children are spending the majority of their time on screens and online, which is taking them out of the real world. While technology can be educational, learning centered around screens can be overstimulating, overwhelming, and distracting. Fortunately, globes already offer a strong foundation for screen-free geographical education. However, traditional globes aren't always ideal. Globes often have very small text that can be difficult for children to read clearly. It can also be challenging to locate certain countries, especially if a child is unfamiliar with that particular region of the globe. Clearly, a more interactive and intuitive approach to teaching and learning geography is needed.

1.2 Solution

To solve this problem, our group has designed and created an engaging product to help students learn geography without being attached to a screen. That product is the Voice-Activated Geographic Reference Globe. Our voice-activated globe solves this problem by making learning fun and interactive using a speech detection mechanism. This mechanism will recognize a spoken country as input and automatically rotate the globe, ensuring that the specified location ends up at a predefined center point. This country will then be identified by a laser pointer, illuminating the region that the user is searching for. This product makes using a globe much simpler. The speech-based interaction means that the system is intuitive and easy to use for young children. The combination of spoken country names and visual identification using the globe and laser pointer helps users form stronger connections between countries and their locations. Using a globe rather than just a screen that displays each country also helps with 3D spatial positioning of the countries in the user's mind, which lets them remember countries in relation to each other. While engaging, this globe will not overwhelm children the way a computer program will.

1.3 Functionality

This section will outline the high-level requirements that we have defined for this project.

Our vocabulary (available commands) will be limited to the 197 countries on Earth, plus a "Reset" function with a target accuracy of 75%. The vocabulary of our system, the commands, allows our globe to be used for its main purpose: identifying the countries of the world. This requirement sets the bar for our command library.

The status LEDs will light up red if a valid command is not recognized within 10 s, blue after the Wake Word is detected and the microphone is accepting input, and green when the command is recognized and matched with a command in our database. The status LEDs are designed to provide real-time feedback to the user about the status of the input process. As the only form of input feedback our system offers, the status LEDs are an integral part of our design.

Once given a valid input command, the globe correctly rotates to either display the country or the reset position at the predefined center reference. This requirement ensures our rotation functions correctly, which is essential to identifying countries.

The laser pointer accurately points to the predefined center reference point. It turns on automatically once the globe is done rotating and turns off when the Wake Word is detected, remaining off until rotation is complete. This requirement describes the “reference” part of our system. Without the system portion described in this high-level requirement, the globe would not be able to identify countries.

When the reset command is detected, the globe will rotate to the reset position: 0˚ Latitude, 0˚ Longitude. This command not only allows the globe to go to a neutral position, but it also allows the user to recalibrate the globe, ensuring that it performs optimally.

1.4 Subsystems Overview

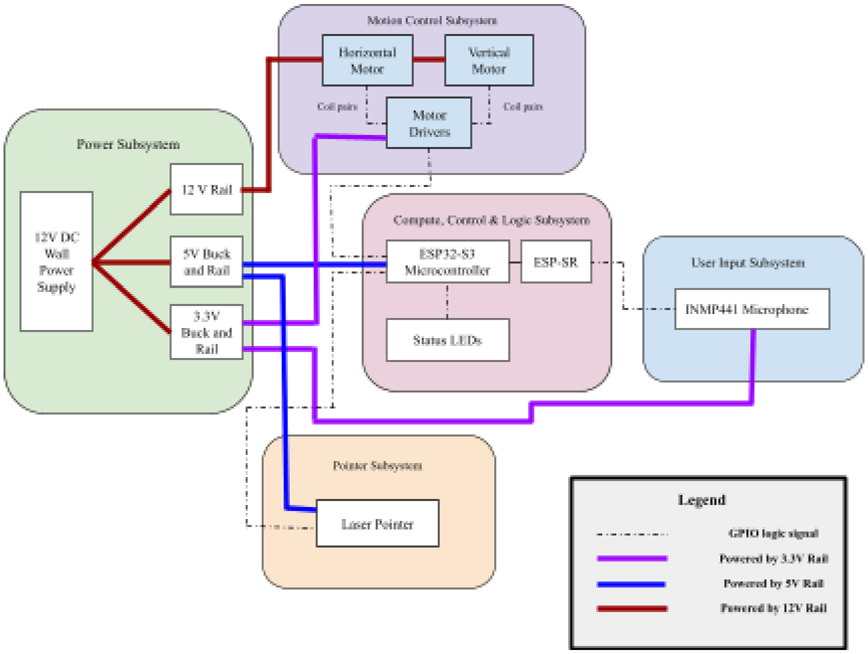


Figure 1: Block Diagram

1.4.1 User Input

This subsystem captures the audio from the user for the Wake Word and command, then sends the information to the Compute subsystem to be processed. The User Input subsystem consists of INMP441 I2S Digital Microphone which will be connected to the GPIO pins of the ESP32-S3 MCU. When the Wake Word is detected, the microphone will listen for a command from the user. The command must match one of the 198 recognized commands in order to count as successful audio detection. The microphone will stream audio to the ESP32-S3 over the I2S interface using a sampling format compatible with ESP-SR, Espressif's Speech Command Recognition Framework, which requires a 16 kHz sampling rate and a mono channel format [2]. The MCU then processes this audio and transfers it to the Compute, Control, and Logic subsystem, where it uses ESP-SR to help determine which country the user wants to find.

1.4.2 Compute, Control, and Logic

This subsystem is responsible for processing audio and calculating motor movements. It takes the audio input collected by the User Input Subsystem and extracts the command using ESP-SR. This subsystem will then match the recognized command with a command from the database to find the target location. We will be using a coordinate system similar to the latitude and longitude system to identify each country on the globe, scaled to steps. Each step is 1.7˚. After that, the subsystem will calculate the movement along each axis in number of steps needed to get from the current position to the target position. It will then send that information over to the Motion Control subsystem to begin rotation.

In addition to telling the Motion Control Subsystem how and where to move the globe, this subsystem provides real-time feedback to the user through the Status LEDs. Based on the current state in the speech recognition pipeline, this subsystem controls the status LEDs to communicate what state system is currently in. While audio input is actively being collected, the subsystem turns on the blue status LED to indicate input collection. If the subsystem fails to identify a valid command or fails to match the identified input with the command in the country database within 10 s, it turns the red LED on. When a command is correctly identified and matched in the country database, the green LED is turned on to indicate a success. This will help the user understand if a command needs to be repeated, if the command was correctly recognized, or if they need to try again.

1.4.3 Motion Control

The Motion Control Subsystem consists of two Nema-17 Stepper Motors and two Pololu Breakout A4988 Motor Driver Modules. This subsystem will implement the motion control of the globe This refers to the rotation of the globe along the z-axis (the actual Earth's axis of rotation) and the y-axis (the axis perpendicular to the frame of view). After a command is recognized as input from the User Input subsystem and movement is calculated in the Compute, Control & Logic Subsystem, this subsystem will receive movement information from the Compute, Control & Logic Subsystem. The GPIO pins of the ESP32-S3 MCU are connected to the motor drivers, and send movement information to the motor drivers. After receiving movement information, the z-axis stepper motor driver and the y-axis stepper motor driver will drive their respective motors to rotate the globe to the target position. The motors have the capability to calculate exact rotation, so we can use this to make sure the globe rotates to exactly the desired position.

1.4.4 Pointer

This subsystem will implement the pointer subsystem. This is the subsystem that controls when the laser pointer, which identifies the specified country, will turn on and off. When the globe has rotated to the target position, the ESP32-S3 MCU will send a signal to the laser pointer, turning it on. The laser pointer will remain on until the Wake Word is detected or the system is turned off.

1.4.5 Power Subsystem

The power subsystem is responsible for powering the full system (MCU, microphone, LED, motors, laser) by providing regulated DC power using an external wall adapter. The wall adapter converts the AC power from the wall outlet to a regulated 12 V DC power supply, which serves as the main input into the system. Voltage regulators will then convert the 12 V power supply into 5 V and 3.3 V rails to power the various devices in our project. The 12 V → 5 V buck converter will generate a regulated 5 V rail for components such as the status LEDs and laser pointer, and the 12 V → 3.3 V buck converter will generate a regulated 3.3 V rail for the motor drivers and other low-voltage components. This subsystem also includes a master on/off switch between the wall adapter and the PCB, enabling the user to safely turn the system on and off.

2. Design

This section presents the final design of the Voice-Activated Geographic Reference Globe, including the completed physical product and the major design decisions made across each subsystem and during the overall design process. To see the final physical PCB, see Appendix A.3 and A.4. Figure 2 shows the overall final completed product.

Figure 2: Final Product Image

2.1 Design procedure

Our design process was split into five main parts across our subsystems: User Input, Compute, Control, and Logic, Motion Control, Pointer, and Power. Over the course of the semester, we made design adjustments to ensure optimal performance of our globe. Considering these changes, we had to update the requirements and verifications, and the high-level requirements to better reflect the current design of the system.

User input saw the largest change, with the removal of the push-to-talk button. This change was necessary because using the push-to-talk button meant we would have to do custom audio sampling, which felt unnecessary considering the ESP-SR framework was built to be seamlessly integrated with Espressif's Wakenet Wake Word framework. Because of this change, the high-level requirement pertaining to the status LEDs was modified to reference the Wake Word instead of the push-to-talk button.

Additionally, we originally intended to use the ESP32-S3-WROOM1-N16 SMD Chip for the purposes of our project. During research we noticed that Multinet 5 Q8, which is the Speech Command Recognition model we are using, consumes 2310 KB or 2.31 MB of PSRAM [2]. The N16 model however, has 0 MB of PSRAM, and wasn't compatible with our project. We pivoted to using the ESP32-S3-WROOM-1-N8R8 SMD Chip instead, since it has 8 MB of PSRAM and is much more suitable. However, due to routing issues, we were not able to get the SMD ESP to function properly. Prioritizing our final product, we decided to use the ESP32-S3-Devkit-C1-N8R8 instead, which still uses the right model of the chip, allowing us to still utilize the speech recognition model.

The motor subsystem got a similar switch. Due to wiring issues on our PCB, the SMD motor driver system including the A4988 motor driver didn’t function correctly. Due to this, we used the Pololu Breakout A4988 Modules instead, which integrated seamlessly with our design and functioned exactly as intended, providing us with a clean, working final product.

The last subsystem that was modified was the power subsystem. In the original design, the 12 V input had multiple protections, including a fuse and a diode. In our final system, we decided to remove these protections because they were causing problematic voltage drops. It was safe to do so since the 12 V wall power adapter was being used already and had all of these protections included. Since this was a last minute decision, we were not able to order new PCBs reflecting this change. To combat this issue, we bridged the whole 12 V input system without the components included.

Aside from the changes to each subsystem, we also changed the mechanical design of our project. Our initial design included a cradle with a rail system similar to rollercoaster rails that the globe would glide along. We switched to our current design because we determined that the motor would not be able to hold the globe steady in the rail system, and that our current design would be much more stable. Skee Alderich at the machine shop then built the mechanical portion of the globe.

2.2 Hardware Design

As mentioned earlier, our hardware design was split across our five subsystems, which will be discussed in the following sections. To view the final overall schematic and PCB routing, see Appendix B.

2.2.1 User Input

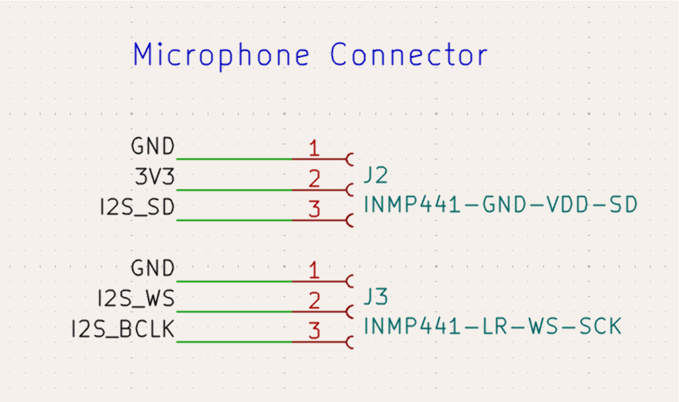


Figure 3: Microphone Connector Schematic

Figure 3 shows the schematic for the User Input subsystem. This subsystem consists of an INMP441 microphone. We chose this microphone, since ESP-SR requires an omnidirectional MEMS microphone [2]. This is connected to the ESP via three I2S signals: SD, WS, and BCLK. Two other pins are connected to GND, while the last pin is connected to the 3.3 V power rail.

2.2.2 Compute, Control, and Logic

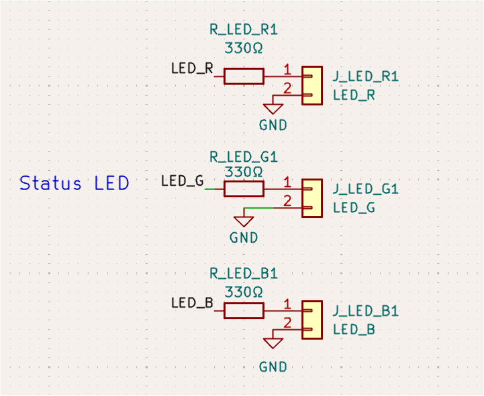


Figure 4: Status LED Schematic

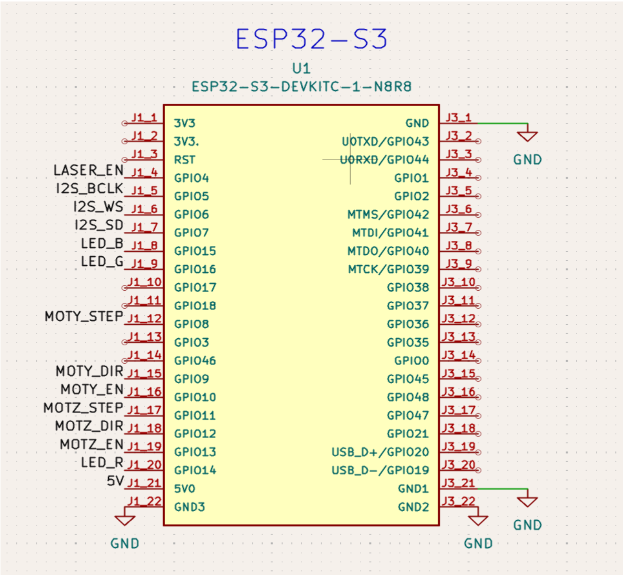


Figure 5: ESP32-S3 Schematic

Figures 4 and 5 show the schematic for the Compute, Control, and Logic subsystem. This subsystem consists of two parts, the ESP and the status LEDs. The ESP32-S3 part is composed of the ESP32-S3-Devkit-C1. The devkit routes all the signals for our design and contains the programming code to execute the project as desired. The status LEDs part consists of three 330 Ω resistors, as well as three

LEDs, one red, one green, and one blue. The three resistors are responsible for protecting the LEDs from current overload. The three LEDs are responsible for lighting up to indicate to the user the status of the input acceptance, as described earlier.

2.2.3 Motion Control

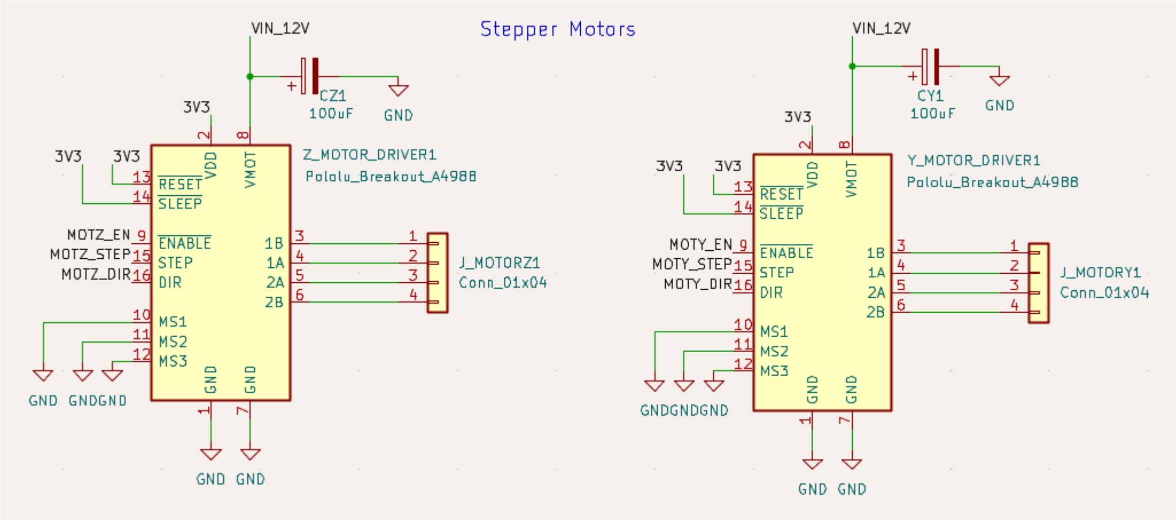


Figure 6: Stepper Motor Drivers Schematic

Figure 6 shows the schematic for the Motion Control Subsystem. This subsystem consists of two A4988 motor breakout motor modules, two 100 µF capacitors, and two NEMA-17 Stepper Motors (not pictured in the schematic). The 100 µF capacitors serve as protection against voltage spikes to protect the motor drivers. In addition, they can help counteract voltage power fluctuations from the 12 V source. The Enable, Step, and Dir signals all come from the ESP, and the VDD, Reset, and Sleep signals come from the 3.3 V power source. The rest are all grounded, except for the 1A, 1B, 2A, and 2B signals, which are the two pairs of signals sent to the stepper motor, which instructs the motor on how much to turn and in which direction.

2.2.4 Pointer

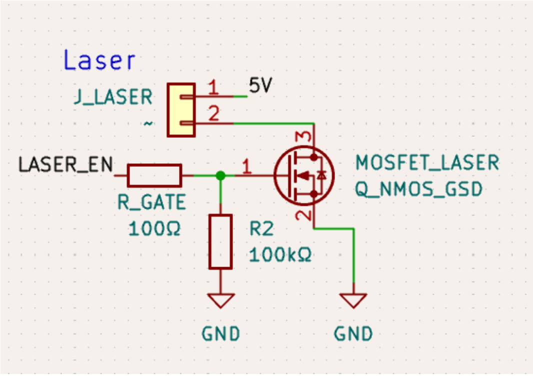


Figure 7: Laser Schematic

Figure 7 shows the schematic for the Pointer subsystem. This subsystem consists of two resistors, one 100 Ω and one 100 kΩ, along with a MOSFET and a connector for the laser. The R_GATE, the 100 Ω resistor, protects the gate when powered to ensure the gate doesn't constantly flip between open and closed. The 100 kΩ resistor is the pull-down resistor, which ensures that the MOSFET stays off and doesn't float when the LASER_EN signal is off. The MOSFET controls the grounding signal of the laser via the gate connected to LASER_EN. When LASER_EN is high, the MOSFET connects J_LASER to ground, which closes the circuit and allows the laser to activate. When LASER_EN is low, the MOSFET disconnects the circuit, disabling the laser.

2.2.5 Power Subsystem

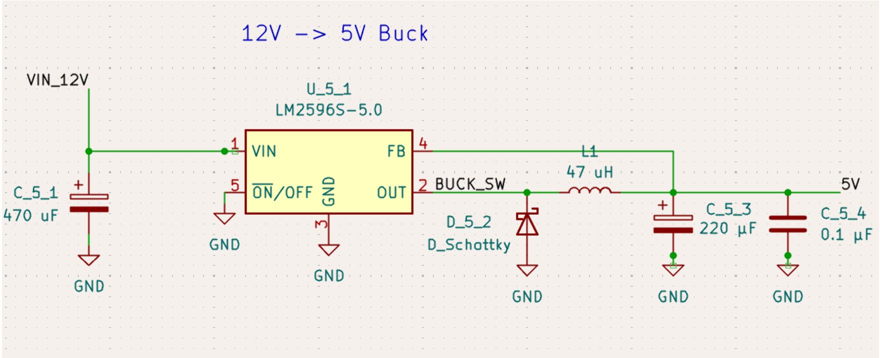


Figure 8: 12 V ⟶ 5 V Buck Schematic

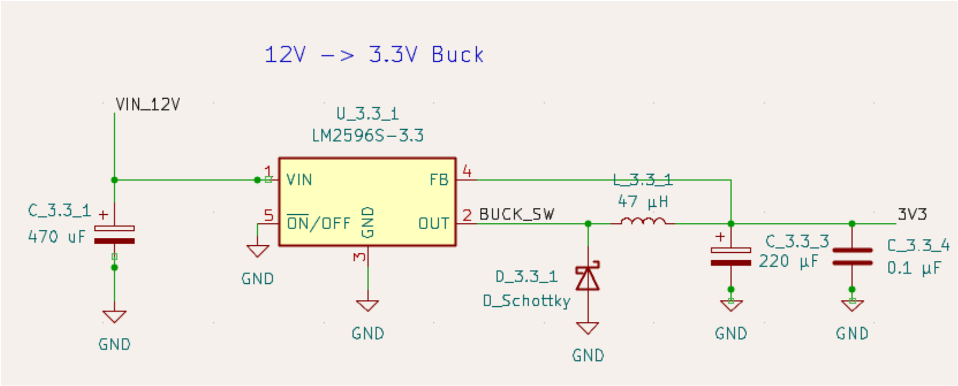


Figure 9: 12 V ⟶ 3.3 V Buck Schematic

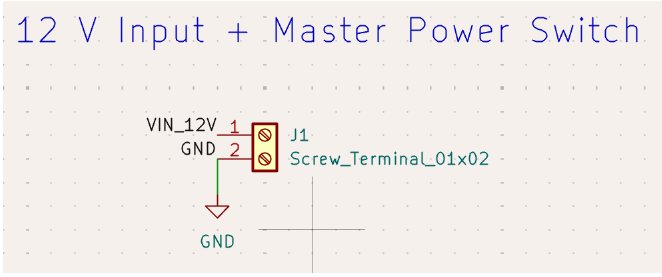


Figure 10: 12 V Input + Master Power Switch Schematic

Figures 8,9, and 10, show the schematic for the Power subsystem. This subsystem consists of three parts: the input, the 5 V buck, and the 3.3 V buck. The input system consists of one screw terminal, which accepts the 12 V input from the wall terminal, and a ground pin. Each buck consists of a power buck, 470 µF capacitor, a Schottky diode, a 220 µF capacitor, a 0.1 µF capacitor, and a 47 µH inductor. The buck converter converts the 12 V input to the desired output, either 5 V or 3.3 V as specified by the system. The 470 µF capacitor acts as protection from voltage surges as well as to smooth out voltage fluctuations in the 12 V input. The Schottky diode acts as a drain for the current when the buck is not active. The 47 µH inductor smooths the output voltage to produce a constant voltage. The 220 µF capacitor helps to stabilize the output voltage, and the 0.1 µF capacitor helps filter noise.

2.3 Software Design

Software design was split among output control speech recognition, coordinate calculation, and motor communication. Output behavior, including the status LEDs and laser pointer, was controlled using simple gpio_set_level() commands to turn each GPIO output on or off at the appropriate point in the system sequence. This section will describe how speech recognition, coordinate calculation, and motor communication were implemented in more detail.

2.3.1 Speech Recognition

The speech recognition software was built from an official Espressif English ESP-Skainet example that already provided the main ESP-SR framework for WakeNet wake-word detection and MultiNet 5 Q8 English Speech Command Recognition. We chose to work off a branch that already supported the ESP32-S3 DevKit configuration, since the main branch only allows support for boards with an internal microphone [1]. To allow MultiNet to recognize the project vocabulary, each country command had to be converted into phonemes using the multinet_g2p.py tool [3] and added to the SDK configuration file. For an example of how these phonemes are stored in the SDK configuration file and how they appear compared to the country name itself, see Appendix A.1 and A.2. This allowed the model to listen for the full set of country commands, along with the reset command, which is the word “start”.

During operation, the system first waits for the wake word, “Hi, ESP”. When WakeNet detects the wake word and verifies the audio channel, the software sets a detection flag to one. When the flag is set to one, the fetched audio is passed into MultiNet so it can compare the spoken input against the stored country vocabulary. When MultiNet detects a valid command, the software reads the command ID that corresponds to the detected command. Since the country database is stored in the same order as the phonemes stored in the SDK configuration file, the country index is equal to the command ID and can be easily used to retrieve coordinates for motion control.

After a valid country is recognized, the detection flag is set back to zero. This prevents the software from passing additional audio into MultiNet for new country-command detection while the globe is already moving. This is necessary because, as specified in our Requirements and Verification Table in Appendix C.1, the globe system should respond to only one country at a time. Otherwise, a second spoken country could interrupt the first movement before the original target reaches the predefined center point. Once the globe finishes rotating to the selected country, speech recognition is re-enabled so the system can accept the next wake word and country command.

2.3.2 Coordinate Calculation and Motor Communication

The coordinate calculation software translates each recognized country into the motor positions needed to center that country on the globe. Each country is stored in the database with an x_steps value for horizontal rotation and a y_steps value for vertical rotation, both measured as offsets from the reset position of (0,0). Our reset function, the word “start”, is stored after all of the 197 countries in the database and in the SDK configuration file. Figure 11 shows the structure of country_coords database, including how each country name is stored with its corresponding x_steps and y_steps values.

```
C/C++
typedef struct {
    const char *name;
    int x_steps;   // horizontal offset from reset (0,0)
    int y_steps;   // vertical offset from reset (0,0)
} country_coord;

static const country_coord country_coords[] = {
    { "country1", x1, y1 },
       // ...
       // remaining 196 countries alphabetically
       // ...
     { "start", 0, 0 }   // one entry for reset function "start" at the very end
    }
```

Figure 11: country_coords database, demonstrating how country and x_steps and y_steps is stored together to be easily retrieved

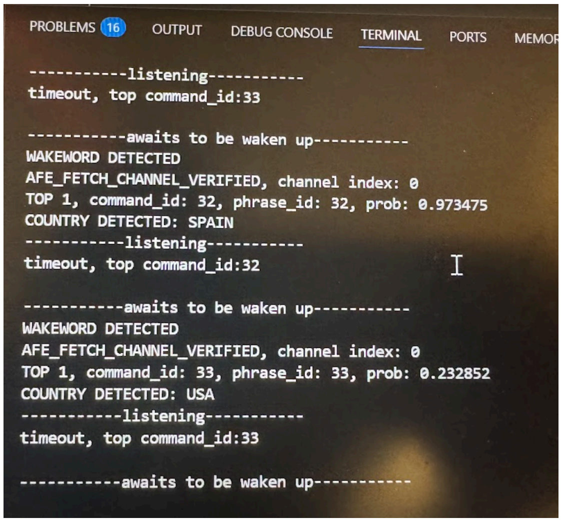
When the speech recognition software identifies a country, the globe software uses the country index to retrieve the stored step coordinates and passes the target values to the motor control software. The motor control logic calculates the required change in steps for each axis and moves the globe to the destination in three phases. First, the vertical axis returns to the equator, then the globe rotates horizontally to the target longitude, and finally, the vertical axis moves to the target latitude. To see the pseudocode logic for how this computation and movement to a target position is done, see Appendix B.3. This movement sequence reduces accumulated positioning error from rotating on a tilted axis and keeps the path consistent when moving between countries.

The actual motor motion is controlled through the A4988 driver using DIR and STEP signals from the ESP32. The firmware waits 20 µs after setting DIR so the direction is stable before stepping, then creates each step by pulsing STEP HIGH and STEP LOW. The step delay, 35,000 µs for the X direction and 70,000 µs for the Y direction, was significantly larger than the A4988 minimum timing requirement of 1 µs HIGH and 1 µs LOW [4], since the motors require time to move the weight of the globe smoothly without skipping steps. To see the code for controlling the motor with these pulses, see Appendix B.4. After the movement is complete, both motor drivers remain enabled so that the globe holds its final position.

3. Design Verification

3.1 User Input

To ensure compliance with our requirement that the globe system must operate in ambient noise conditions, as specified in Appendix Table C.1 Requirement 3, all spoken tests were conducted in an open setting with a decibel meter showing at least 50 dB. Additionally, we had firmware logs showing successful detection of a country name when standing at a distance of 20 cm away from the microphone. Figure 12 shows the successful verification of Requirement 2 in Appendix Table C.1.



```
-----------listening----------
timeout, top command_id:33

-----------awaits to be waken up----------
WAKEWORD DETECTED
AFE_FETCH_CHANNEL_VERIFIED, channel index: 0
TOP 1, command_id: 32, phrase_id: 32, prob: 0.973475
COUNTRY DETECTED: SPAIN
-----------listening----------
timeout, top command_id:32

-----------awaits to be waken up----------
WAKEWORD DETECTED
AFE_FETCH_CHANNEL_VERIFIED, channel index: 0
TOP 1, command_id: 33, phrase_id: 33, prob: 0.232852
COUNTRY DETECTED: USA
-----------listening----------
timeout, top command_id:33

-----------awaits to be waken up----------
```

Figure 12: User Input Firmware Logs from a distance of 20 cm from the microphone

3.2 Compute, Control, and Logic

We were able to determine that the functionality of the full project is dependent on the functionality of the compute, control, and logic subsystem. As a result, we were able to verify that the Compute, Control, and Logic subsystem met all of the requirements.

3.3 Motion Control

We were able to determine that the functionality of the full project is dependent on the functionality of the motion control subsystem. Due to this, we were able to verify that the motion control subsystem met all of the requirements outlined in the requirements and verifications table, as well as satisfying the high level requirement pertaining to the motors. A demonstration of these requirements is shown in our verifications video in our final presentation.

3.4 Pointer

We were able to verify the pointer subsystem works as intended through observation. The laser turns on and off within the times required. A demonstration of these requirements is shown in our verifications video in our final presentation. Additionally, as specified by Requirement 4 in Appendix Table C.4 the laser is powered by the 5 V rail which is within the range of 2.6 to 6 VDC. Figure 13 shows the verification for the requirement of the laser pointing directly at a country or within our tolerance bounds of 2.5 mm.

Figure 13: Laser pointing directly at Vatican City

3.5 Power Subsystem

We were able to verify the power subsystem using a multimeter. As seen in Table C.5 in Appendix C, we were able to verify that the power subsystem passed all requirements set for it. The figures below demonstrate the successful multimeter readings for each requirement specified in Table C.5.

Figure 14: 12 V input

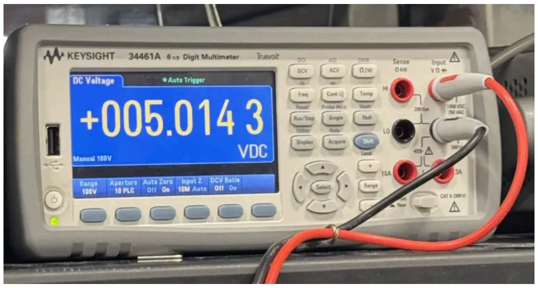


Figure 15: 5 V buck output

Figure 16: 3.3 V buck output

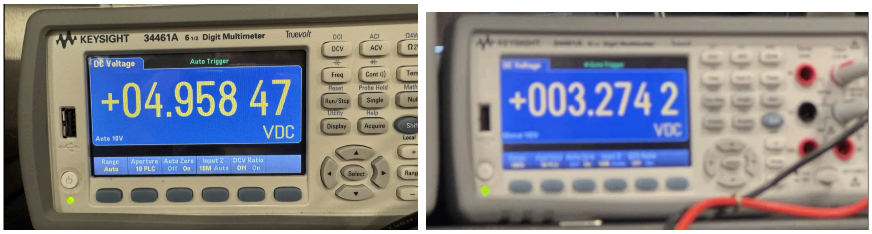


Figure 17 & 18: Both 5 V and 3.3 V rails operating simultaneously

Figure 19: 0 V when power is turned off

4. Costs

4.1 Parts

Table 6. Parts and Costs List

Description	**Manufacturer**	**Qty**	**Unit Price (USD)**	**Total Cost**	**Link**
Globe	Wizdar	1	$30.97	$30.97	Link
Laser Module	Quarton	1	$15.99	$15.99	Link
NEMA 17 Medium Stepper Motors	SparkFun Electronics	2	$14.95	$29.90	Link
12 V Power Supply	AspenTek	1	$9.99	$9.99	Link
INMP441 Microphone Modules	Teyleten	1	$12.99	$12.99	Link
LM2596S-5.0/NOPB IC Buck	Texas Instruments	1	$6.97	$6.97	Link
LM2596S-3.3 IC Buck	Texas Instruments	1	$8.76	$8.76	Link
Power Inductors	Bourns	2	$0.939	$1.88	Link
ESP32-S3-WROOM-1 DEVKIT-C1-N8R8	Espressif Systems	1	$15.00	$15.00	Link
A4988 Stepper Driver Breakout Module	E-outstanding	2	$4.44	$8.88	Link
Status LEDS, capacitors, resistors, connector wires	—	–	—	$5.00	—
Total:				**$146.33**	

4.2 Labor

The average starting salary for a Computer Engineering graduate from the University of Illinois Urbana-Champaign is $103,222 [7]. This would mean that the hourly rate would be approximately $49.63. Each team member spends approximately 15 hours a week for around 10 weeks, entailing that the total salary for one team member would be $18,611.25. Since there are three of us, the total group labor costs would be $55,833.75. Additionally, the Machine Shop is assisting with the development and assembly of our globe base and dual-axis cradle system. It will take around a week for the construction of the globe system to complete. Assuming a total of ten hours for the total construction and assembly of the globe and an hourly salary of $70, the cost for the machine shop labor would be $1750. Therefore, the total cost of labor alone would approximately be $57,583.75.

5. Schedule

Table 7. *Project Schedule*

Week	Task	Person
February 23rd – March 1st	Finalize ESP32 and microphone breakout board	Mahathi
	Finalize motor control breakout board. Provide machine shop reference diagram and parts	Rijul
	Finalize parts to order and finish buck regulator breakout board	Varsha
March 2nd – March 8th	Design Review with Instructors and TAs	Everyone
	Communicate with Machine Shop regarding progress of globe construction	Everyone
	Revise PCB based on feedback and submit second round PCB order	Everyone
March 9th – March 15th	Breadboard Demo with Instructors and TAs	Everyone
	Program ESP32 MCU Breakout board to test microphone, LEDs, and laser control	Mahathi
	Test buck regulators to ensure safe voltage conversion	Rijul and Varsha
March 16th – March 22nd	**SPRING BREAK**	
March 23rd – March 29th	Set up ESP32-SR and test its capabilities	Mahathi
	Pick up globe from Machine Shop and integrate it with PCB	Rijul and Varsha
	Modify PCB based on testing and submit fourth round orders	Everyone

March 30th – April 5th	Testing and Debugging with globe integration	Everyone
	Individual Progress Reports	Everyone
April 6th – April 12th	Testing and Debugging with globe integration	Everyone
	Progress Demos	Everyone
	Team Contract Assessment	Everyone
April 13th – April 19th	Testing and Debugging with globe integration	Varsha & Mahathi
	Finalize assembling all components for the final globe system	Rijul
April 20th – April 26th	Mock Presentations	Everyone
	Mock Demos	Everyone
April 27th – May 3rd	Final Presentations	Everyone
	Final Demos	Everyone
May 4th – May 17th	Final Papers & Lab Notebooks Due	Everyone

6. Conclusion

6.1 Accomplishments

Our project successfully met all the high-level requirements that we specified at the beginning of the semester. Additionally, we were able to successfully verify all the requirements that we specified for each subsystem, as seen in Appendix C. Our project works as intended. In our first high level requirement, we state that our intended accuracy for command recognition is 75%, but we were able to get 100% of our commands to be successfully recognized when spoken into the microphone. We were also able to overcome many challenges throughout the semester to deliver a working product. However, our biggest accomplishment this semester was receiving an honorable mention for our project. This was huge for us considering our group had little to no experience with PCBs or PCB design prior to taking this class.

6.2 Uncertainties

As mentioned in section 2.1, we noticed various design flaws late in the design process. The PCBs we had ordered during the third round PCB orders were not usable due to design issues. Due to this, we had to order new PCBs on our own. As mentioned in Section 2.1, due to routing issues, we were not able to use these new PCBs either. We ended up reverting back to an older PCB design that used both the ESP32-S3-Devkit-C1 and the Pololu A4988 Breakout Modules rather than their SMD counterparts. We also had to manually bridge several components in the power subsystem to remove the “protections” added for the power input due to them causing unwanted voltage drops. Despite all of these issues, we were able to create a fully working product.

6.3 Future Work and Alternatives

We have several ideas for future work, consisting of both improvements to our current design and additions to our project. In regards to improving our current design, we want to implement the SMD ESP32-S3-WROOM1-N8R8 instead of the ESP32-S3-Devkit-C1 and the A4988 SMD Motor Chips instead of the breakout motor modules on our PCB. The next improvement to make would be mounting our PCB in the PCB box neatly. This would ensure that none of the components are in danger due to loose wires in close proximity of all of our major components. If the PCB were mounted, it would not move around, making transportation safer. Also, the overall design would look more polished and professional. The last improvement we would like to make would be to enable microstepping in our motors for increased precision.

In terms of future modifications, we want to use a laser with a smaller diameter. Currently the laser is larger than some of the smallest countries on Earth (like Luxembourg and Vatican City), making it hard to clearly point them out on the globe. Additionally, we would like to add a homing sensor that will automatically position our globe at the Reset position, eliminating the need for manual calibration of the globe. The last addition we would like to implement is an electronic display screen that will display information about the country selected, such as the population, flag, fun facts, etc., which will facilitate further learning.

6.4 Ethical, Safety, and Social Impact

As we are living in an age of data privacy concerns, we recognize that our globe can bring up ethical concerns, especially with regards to our globe accepting audio input from the user. To mitigate these concerns, our speech command recognition model and the overall system work fully offline and do not store the audio samples in any database. The audio samples are used to identify the spoken command and match it to a command from our local list. After this process ends, the collected audio sample is destroyed and forgotten. Therefore, there is no risk of the user's voice being stolen, leaked, and/or used for nefarious purposes.

Since our globe uses a Class 2 laser, we have to abide by several international regulations to ensure the user's safety in regards to the laser. Class 2 lasers do not require any eye protection when in use, but do require warning labels to be placed near the laser, as stated in IEC 60825-1 [5]. To comply with these regulations, we have placed two warning labels around the base of the project, near the laser. In addition, the laser is mounted in place and cannot be moved to prevent children from playing with the laser and potentially harming their eyes.

In terms of social impact, we expect our reference globe to be used in an educational setting. As mentioned earlier, we intend for this globe to be a screen-free alternative to educational apps or websites that teach geography. Additionally, the voice activated geographic reference globe can be used in areas without a stable internet connection since the device is fully offline. This makes geographic education more accessible to less fortunate areas of the world, since every child deserves the same access to education regardless of socio-economic status.

References

[1] 0015, “esp-skainet,” GitHub repository, branch “ESP32-S3-Devkit-C,” forked from Espressif’s esp-skainet. [Online]. Available: https://github.com/0015/esp-skainet/tree/ESP32-S3-Devkit-C

[2] Espressif Systems, ESP32-S3 ESP-SR User Guide, Release master, Apr. 28, 2026. [Online]. Available: https://docs.espressif.com/projects/esp-sr/en/latest/esp32s3/esp-sr-en-master-esp32s3.pdf

[3] W. W. Wang, “multinet_g2p.py,” in esp-sr, Espressif Systems, GitHub, commit 749c963. [Online]. Available: https://github.com/espressif/esp-sr/blob/749c963/tool/multinet_g2p.py

[4] Allegro MicroSystems, LLC, A4988 DMOS Microstepping Driver with Translator and Overcurrent Protection, Rev. 5, May 7, 2014. [Online]. Available: https://www.pololu.com/file/0J450/a4988_dmos_microstepping_driver_with_translator.pdf

[5] Woodrow Scientific. “Class Two Laser.” Woodrow Scientific Blog. https://woodrowscientific.com/blog?p=class-two-laser

Appendix A

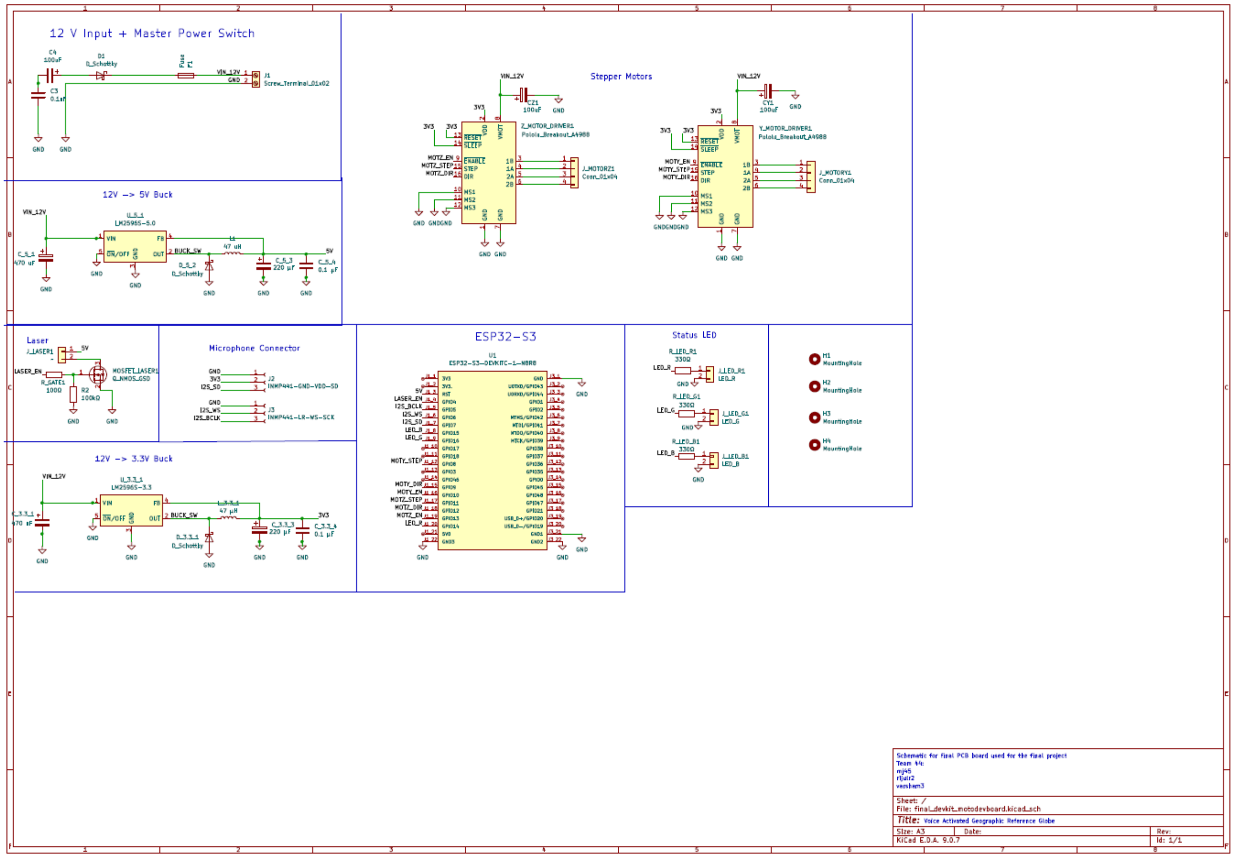


Figure A.1: Full Schematic for the Final PCB board used for the final demonstration

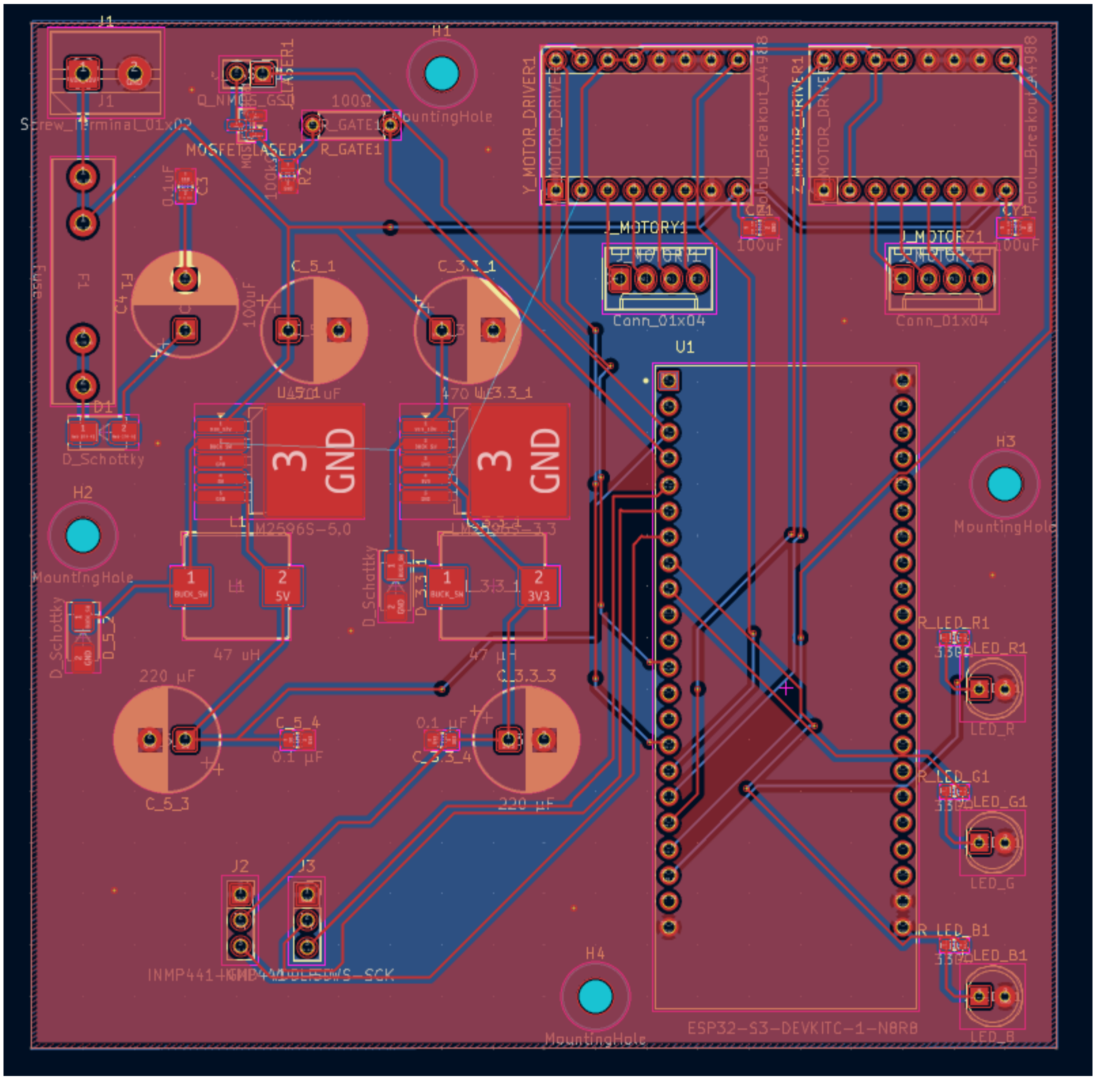


Figure A.2: Routing of the full PCB used for the final demonstration

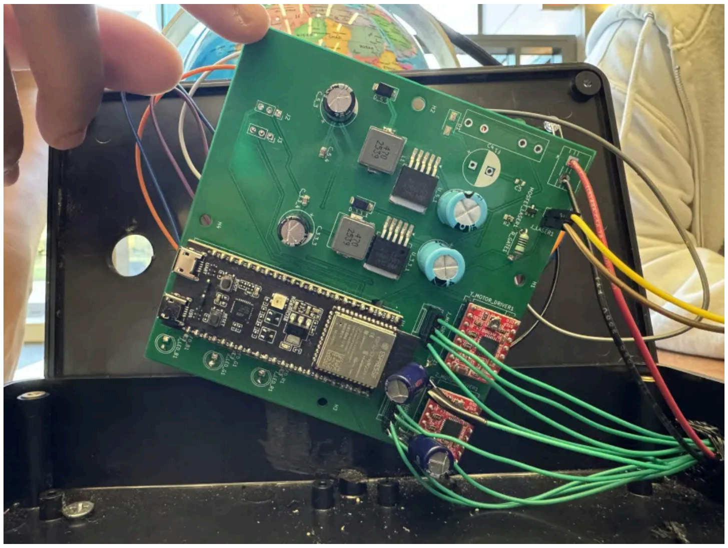


Figure A.3: Front Side of Final PCB

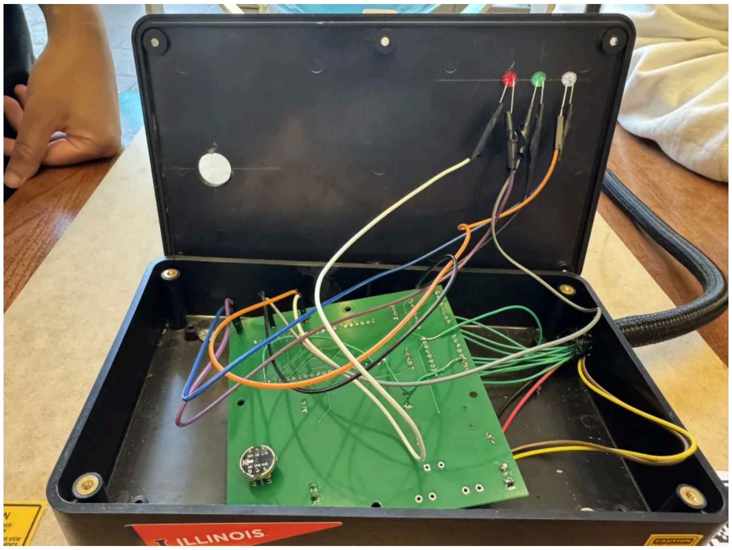
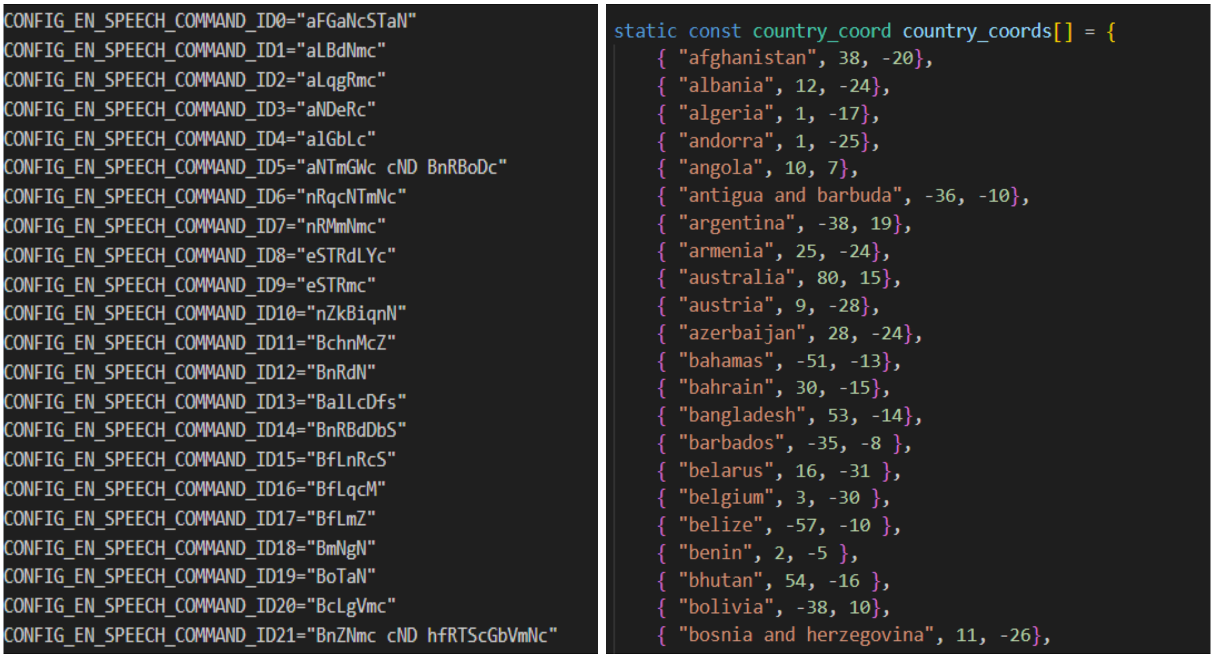


Figure A.4: Back Side of Final PCB

Appendix B



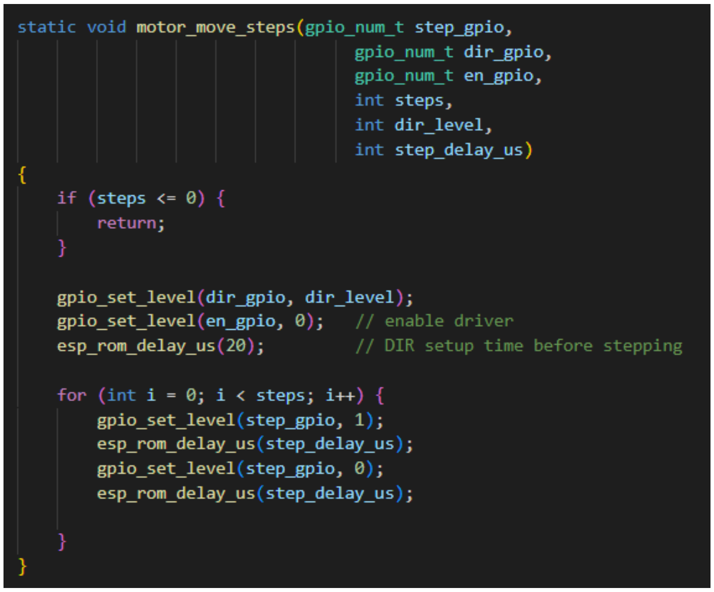
```
CONFIG_EN_SPEECH_COMMAND_ID0="aFGaNcSTaN"
CONFIG_EN_SPEECH_COMMAND_ID1="aLBdNmc"
CONFIG_EN_SPEECH_COMMAND_ID2="aLqgRmc"
CONFIG_EN_SPEECH_COMMAND_ID3="aNDeRc"
CONFIG_EN_SPEECH_COMMAND_ID4="alGbLc"
CONFIG_EN_SPEECH_COMMAND_ID5="aNTmGWc cND BnRBoDc"
CONFIG_EN_SPEECH_COMMAND_ID6="nRqcNTmNc"
CONFIG_EN_SPEECH_COMMAND_ID7="nRMmNmc"
CONFIG_EN_SPEECH_COMMAND_ID8="eSTRdLYc"
CONFIG_EN_SPEECH_COMMAND_ID9="eSTRmc"
CONFIG_EN_SPEECH_COMMAND_ID10="nZkBiqnN"
CONFIG_EN_SPEECH_COMMAND_ID11="BchnMcZ"
CONFIG_EN_SPEECH_COMMAND_ID12="BnRdN"
CONFIG_EN_SPEECH_COMMAND_ID13="BalLcDfs"
CONFIG_EN_SPEECH_COMMAND_ID14="BnRBdDbS"
CONFIG_EN_SPEECH_COMMAND_ID15="BfLnRcS"
CONFIG_EN_SPEECH_COMMAND_ID16="BfLqcM"
CONFIG_EN_SPEECH_COMMAND_ID17="BfLmZ"
CONFIG_EN_SPEECH_COMMAND_ID18="BmNgN"
CONFIG_EN_SPEECH_COMMAND_ID19="BoTaN"
CONFIG_EN_SPEECH_COMMAND_ID20="BcLgVmc"
CONFIG_EN_SPEECH_COMMAND_ID21="BnZNmc cND hfRTScGbVmNc"
```

```
static const country_coord country_coords[] = {
    { "afghanistan", 38, -20},
    { "albania", 12, -24},
    { "algeria", 1, -17},
    { "andorra", 1, -25},
    { "angola", 10, 7},
    { "antigua and barbuda", -36, -10},
    { "argentina", -38, 19},
    { "armenia", 25, -24},
    { "australia", 80, 15},
    { "austria", 9, -28},
    { "azerbaijan", 28, -24},
    { "bahamas", -51, -13},
    { "bahrain", 30, -15},
    { "bangladesh", 53, -14},
    { "barbados", -35, -8 },
    { "belarus", 16, -31 },
    { "belgium", 3, -30 },
    { "belize", -57, -10 },
    { "benin", 2, -5 },
    { "bhutan", 54, -16 },
    { "bolivia", -38, 10},
    { "bosnia and herzegovina", 11, -26},
```

Figure B.1 and B.2: Phonemes in SDK Configuration file versus the actual country names for the first 22 countries in the country_coords database

```
FUNCTION move_axis_to(target_steps):

      change_in_steps = target_steps - current_steps

      direction = sign of change_in_steps

      steps_to_move = absolute value of change_in_steps

      move motor in direction for steps_to_move steps //reference
                                                      //Appendix B.4

      current_steps = target_steps


FUNCTION move_globe_to(target_x_steps, target_y_steps):

    move_y_axis_to(0)                // return globe to equator

    move_x_axis_to(target_x_steps)   // rotate to target longitude

    move_y_axis_to(target_y_steps)   // tilt to target latitude
```

Figure B.3: Pseudocode for logic on how steps and movement are computed to move from the current_steps position to target steps position



```c
static void motor_move_steps(gpio_num_t step_gpio,
                             gpio_num_t dir_gpio,
                             gpio_num_t en_gpio,
                             int steps,
                             int dir_level,
                             int step_delay_us)
{
    if (steps <= 0) {
        return;
    }

    gpio_set_level(dir_gpio, dir_level);
    gpio_set_level(en_gpio, 0);   // enable driver
    esp_rom_delay_us(20);         // DIR setup time before stepping

    for (int i = 0; i < steps; i++) {
        gpio_set_level(step_gpio, 1);
        esp_rom_delay_us(step_delay_us);
        gpio_set_level(step_gpio, 0);
        esp_rom_delay_us(step_delay_us);

    }
}
```

Figure B.4: Logic for how pulses with delays are sent to the motor driver to control the motor

Appendix C

Table C.1 User Input Subsystem - Requirements and Verification

Requirements	Verification	Verification Status
● The user input subsystem will stop capturing audio within 1s of successful country detection	● **Equipment:** ESP32-S3 MCU, firmware with timestamps, microphone ● **Test:** Speak a valid country name after the Wake Word from the country database into the microphone. Verify through firmware output that audio capture immediately stops after successful country detection. ● **Demonstration:** Immediately after successful country detection, indicated by the green status LED, speak another valid country name and verify that the globe does not start moving to the next spoken location immediately.	**Verified**
● The subsystem will accept spoken input from a user located 10 to 20 cm from the microphone under normal indoor audio conditions	● **Equipment:** Ruler, microphone, ESP32 MCU, status LED ● **Test:** Stand at distances between 10 cm and 20 cm from the microphone and use a ruler to confirm the distance. Speak several country names under normal indoor room noise conditions. Verify through the status LED output that spoken input is recognized. ● **Demonstration:** Status LED output showing successful spoken input capture and detection	**Verified**
● The ESP accurately detects the country spoken even in the presence of ambient noise (50 decibels), such as a normal conversation between individuals	● **Equipment:** ESP32-S3 MCU, firmware, ● **Test:** Speak a valid country name from the country database into the microphone. While this is happening, we play an audio recording measuring around 50 dB at the microphone, and check firmware output that the correct country name was detected. ● **Demonstration:** Recording of globe moving to the correct destination with decibel meter showing 50 dB of ambient noise at the microphone.	**Verified**

Table C.2 Compute, Control & Logic Subsystem - Requirements and Verification

Requirements	Verification	Verification Status
The Compute, Control, and Logic subsystem will set the status LEDs accordingly, and the status LEDs must update to the correct state within 400 ms of any state change	**Equipment**: ESP32-S3 MCU, INMP441 Microphone, firmware, camera **Test**: Trigger each of the three states associated with the three status LEDs.	**Verified**
The Compute, Control, and Logic subsystem must process audio data from the User Input Subsystem using ESP-SR and output a single recognized country label or error within 2s of audio capture ending	**Equipment**: ESP32-S3 MCU, INMP441 Microphone, PS1024ALRED Push Button, firmware **Test**: Perform multiple trials of speaking valid country names while holding the push-to-talk button. Release the button and use timestamped firmware logs to measure the time from capture to final recognition. **Demonstration**: Serial log showing recognition output and measured latency for each trial.	**Verified**
The subsystem must determine whether the recognized country exists in the country database. If it does not exist, the subsystem will generate an error state and set the red status LED within 400ms of the invalid recognition.	**Equipment**: ESP32-S3 MCU, firmware, status LED hardware **Test**: Speak a series of phrases that are not in the country database. Verify that the firmware flags "not in the database" and that the red LED is set within 400ms of the result. **Demonstration**: Video and serial log showing invalid input detection and LED change timing.	**Verified**
If the recognized country exists in the database, the subsystem must retrieve the correct target (z, y) coordinates from the country database	**Equipment**: ESP32-S3 MCU, firmware **Test**: Select a fixed list of countries with known stored coordinates. Speak each one and compare the printed (z, y) values to the expected values from the database. **Demonstration**: Table of expected vs. retrieved coordinates for each tested	**Verified**

	country.	
- The subsystem must compute the difference between the current globe position and the target position, convert it into motor steps for the z-axis and y-axis, and send step commands to the Motion Control Subsystem within < 500 ms after coordinates are retrieved	- **Equipment**: ESP32-S3 MCU, firmware with timestamped step-command prints, Motion Control interface (GPIO outputs or command packets) - **Test**: Initialize the system to the neutral position, then input multiple countries. Confirm that the computed step counts match hand calculations for the same (current → target) coordinate pairs, and measure timing from coordinate retrieval to command output. - **Demonstration**: Serial log showing current position, target position, computed z-steps/y-steps, timestamps.	**Verified**
- The subsystem must keep track of the current globe position after each completed movement	- **Equipment**: ESP32-S3 MCU, firmware, laser - **Test**: Run a sequence of at least 10 country inputs. After each movement completion signal, verify that the stored "current position" equals the prior movement's target position and that subsequent step calculations use the updated value.	**Verified**

Table C.3 Motion Control Subsystem - Requirements and Verification

Requirements	Verification	Verification Status
- The motion system must rotate the globe on both axes to the location specified by the Compute, Control & Logic Subsystem.	- **Equipment:** Camera, Ruler - **Test:** Enter a series of countries into the globe. Check the location the globe rotates to and compare this to the target location specified in the code. - **Demonstration:** A recording of these tests and a table of the target location and the actual location rotated to.	**Verified**
- The z-axis must have a full 360 degrees of motion available.	- **Equipment:** Camera - **Test:** Manually rotate the globe along the z-axis in both directions to ensure the globe has a total 360 degrees of motion available. - **Demonstration:** A recording of these tests.	**Verified**

The y-axis must have 180 degrees of motion available.	**Equipment:** Camera **Test:** Manually rotate the globe along the y-axis in both directions to ensure the globe has a total 180 degrees of motion available. **Demonstration:** A recording of these tests.	**Verified**
The globe must rotate to the desired position within 10 seconds.	**Equipment:** Camera, Stopwatch **Test:** Enter a series of countries into the globe. Measure the time from the green LED activating to when the rotation of the globe is complete. This should be less than 10 seconds. **Demonstration:** A recording of these tests and a time per trial table.	**Verified**
Once rotation is complete, the system must send a signal to the Pointer & Feedback Subsystem to turn on the laser within 1 second.	**Equipment:** Stopwatch, Camera **Test:** Set up the recording. Record multiple trials of giving a country to the globe. Once rotation to the country is complete, use the stopwatch to measure the time to laser activation. **Demonstration:** A recording of successful laser activations and a time per trial log.	**Verified**
Each motor has an operating voltage around 12 Volts, and an operating current around 2 Amps.	**Equipment:** Digital Multimeter **Test:** Activate the motor, and use the multimeter to measure the voltage and current of the motor to ensure both values are around the accepted values. **Demonstration:** A table of current and voltage values from the various trials.	**Verified**

Table C.4 Pointer System - Requirements and Verification

Requirements	Verification	Verification Status
The laser turns on within 1 second of completion of rotation (with the exception of the "Reset" function as mentioned below).	**Equipment:** Stopwatch, Camera **Test:** Set up the recording. Record multiple trials of giving a country to the globe. Once rotation to the country is complete, use the stopwatch to measure the time to laser activation. **Demonstration:** A recording of successful	**Verified**

	laser activations and a time per trial log.	
● The laser turns off within 1 second of a new country being spoken and recognized (with the exception of use after the “Reset” function as mentioned below).	● **Equipment:** Stopwatch, Camera ● **Test:** Set up the recording. Record multiple trials of speaking a new country with the laser on. Once the country is recognized and the green LED indicator turns on, use the stopwatch to measure the time to laser deactivation. ● **Demonstration:** A recording of successful laser deactivations and a time per trial log.	**Verified**
● The laser pointer either points directly within the specified country, or for countries with an area less than 0.25 cm^2 on the globe, the laser points within a distance range of 2.5 mm from the borders.	● **Equipment:** Ruler, Camera ● **Test:** Speak a series of countries into the globe for the set of tests. Use a variety of countries, including big countries like Canada and China, and small countries like Panama and Croatia. For big countries (above 0.25 cm^2, the laser must be within the borders of the country. For smaller countries, the laser must be within 2.5 millimeters from the country’s borders. ● **Demonstration:** A recording of these tests and a distance per trial log.	**Verified**
● The laser pointer will be mounted on the base and will point directly at the equator when the globe is in a neutral position, and the desired country will rotate onto that spot.	● **Equipment:** Camera ● **Test:** Use the “Reset” function to reset the globe to the neutral position. Manually activate the laser (as the neutral position will not activate the laser. Check if the laser is pointing at the equator on the globe. ● **Demonstration:** A recording of these tests.	**Verified**
● The laser pointer operates at a voltage between 2.6 to 6 VDC, and at a current less than 35 mA/50 mA (which is a power of less than 1 mW/2.5 mW).	● **Equipment:** Digital Multimeter ● **Test:** Activate the laser, either manually or by speaking a country into the globe. Measure the voltage and current outputs of the laser. Make sure they are within the accepted range. Calculate the power from these two values. ● **Demonstration:** A table of current, voltage, and power values from the various trials.	**Verified**

- The “Reset” function sets the globe into a neutral position (with the laser pointing at a predetermined spot along the equator) and **does not** activate the laser.	- **Equipment:** Camera - **Test:** Speak the reset word into the globe, and wait for the globe to reset into the neutral position. Ensure the laser does not turn on. - **Demonstration:** A recording of the globe successfully resetting with no laser activations.	**Verified**

Table C.5 Power Subsystem - Requirements and Verification

Requirements	Verification	Verification Status
- The power subsystem must accept a 12 V DC input from the external wall adapter and distribute power to all subsystems	- **Materials**: 12 V wall adapter, multimeter - **Test**: Measure the voltage at the point where the power system connects to the PCB and confirm it is 12 V ± 5%. - **Demonstration**: Multimeter reading showing a stable 12 V input while the system is plugged in.	**Verified**
- The power subsystem must generate a 5 V power rail using a 12 V → 5 V buck converter	- **Materials**: Multimeter - **Test**: Measure the output of the 5 V rail with 0 load and full expected load and confirm voltage remains within 5 V ± 5% - **Demonstration**: Multimeter reading showing a stable 5 V input while the system is plugged in.	**Verified**
- The power subsystem must generate a 3.3 V power rail using a 12 V → 3.3 V buck converter	- **Materials**: Multimeter - **Test**: Measure the output of the 3.3 V rail with 0 load and full expected load and confirm voltage remains within 3.3 V ± 5% - **Demonstration**: Multimeter reading showing a stable 5 V input while the system is plugged in.	**Verified**
- The 5 V and 3.3 V rails must operate simultaneously without causing voltage drop	- **Materials**: Multimeter - **Test**: Measure both rails for a full cycle of taking input, processing input, setting status LEDs, running motor and turning on	**Verified**

greater than 5% on either rail	laser pointer. Confirm neither rail drops below specified tolerance. ● **Demonstration**: Multimeter reading showing a stable 5 V and 3.3 V inputs while the system is plugged in for the 5 V and 3.3 V rails, respectively.	
● The subsystem must include a master on/off switch that completely disconnects power from the rest of the circuitry	● **Materials**: Multimeter ● **Test**: Flip the on/off switch to off. Confirm system is not being powered. ● **Demonstration**: Multimeter reading showing a stable 0V input while the system is plugged in.	**Verified**
● The power subsystem must operate with an efficiency of at least 80% at typical load conditions	● **Materials**: Multimeter ● **Test**: Measure input power (12 V × input current) and output power (sum of 5 V and 3.3 V rail power). Calculate efficiency and confirm it is greater than or equal to 80%. ● **Demonstration**: Calculation sheet showing measured efficiency.	**Verified**