ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

JargonJolt

Team #28

LUKE HARTMANN
(lukegh2@illinois.edu)
DANIEL CHAMOUN
(chamoun2@illinois.edu)
NAN KANG
(nankang2@illinois.edu)

TA: Angquan Yu

May 1, 2024

Abstract

In this project, we successfully designed and built a device that aids users in maintaining consistent progress in flashcard-based language learning. The device uses a digital pet to show progress, which helps keep users engaged and consistent in their studies. Additionally, the device is portable: it has a compact design, lightweight construction, and a long-lasting battery. This report outlines the design, verification, and results of our project.

Contents

1 Introduction

1.1 Problem

When learning a new language, amassing and retaining vocabulary is often one of the most challenging parts of the learning process and can be a choke point for advancing into conversational fluency. It is very easy for people to fall off track when learning a new language/new content, especially in the later stages which can prove detrimental to spaced repetition algorithms. Many peoplewho give up on learning a new language do so due to either a lack of opportunity to practice, boredom, or a perceived high level of difficulty. Our project aims to assist those people to continue their endeavors to learn language.

Flashcard applications that already exist do so primarily as mobile or desktop applications. Desktop applications such as Anki have high functionality, but are not portable and could cause the user to miss days if they do not have access to their PC. Mobile applications require that the user has a smartphone, which is not ideal for certain audiences such as children or the elderly. Battery life is also a concern for longer practice sessions and portability.

1.2 Solution

Our solution is the JargonJolt, a digital pet and portable flashcard device that makes consistently practicing your language skills convenient and fun! The JargonJolt takes advantage of the "Tamagotchi effect". Named after the popular toy by Bandai, the Tamagotchi effect is the phenomenon of humans becoming emotionally attached to machines, robots, or otherwise inanimate entities. We plan to harness this aspect of human psychology to encourage people to keep up with their daily language review and practice. Nurturing/playing with a digital pet who gets happier as you do better in your flashcard reviews will keep flashcard users more engaged during their reviews as well as more consistent.

The JargonJolt makes use of spaced repetition algorithms to show users flashcards in optimal order for memory and knowledge retention. The JargonJolt features dual low-power digital ink screen for displaying both flashcards and the digital pet and several buttons for selecting options for responding to flashcards. Applications of similar functionality may exist as smartphone apps, but the JargonJolt has unique advantages that give it cause to exist as a product. The simplicity and toy-like nature of the JargonJolt makes it ideal for children who are not ready for a smartphone or tablet. A rechargeable battery will also allow users to take their JargonJolt on the go without worrying about the battery life of their mobile devices or the cell reception in any given area.

1.3 Subsystem Overview

As shown in figure 1, this project is broken into 4 main sections: the control/internet subsystem, the memory subsystem, the power subsystem, and the user interface subsystem.

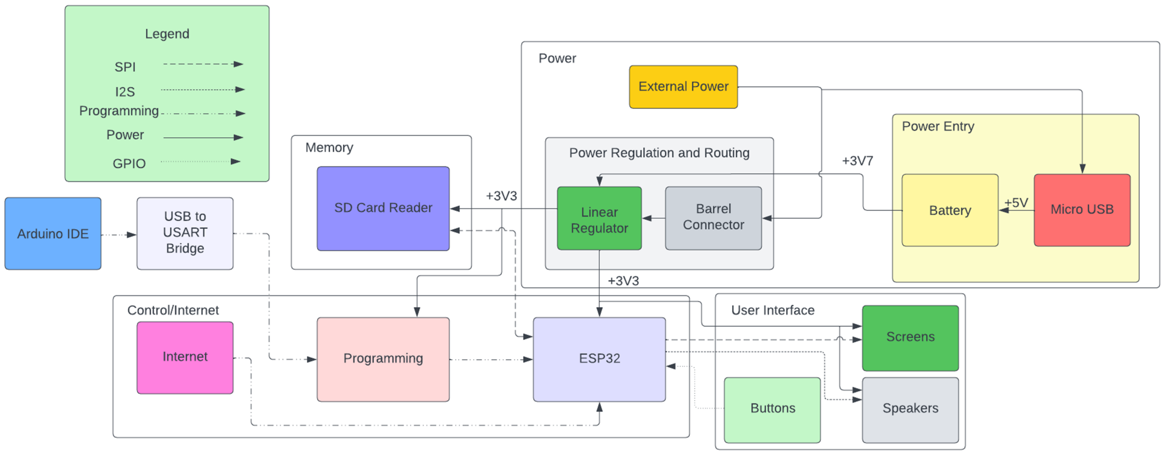

Figure 1: Block Diagram

Each subsystem perform a vital portion of the entire design, each of which is essential to even the most basic operation of the device. The control/internet subsystem serves as the center point for all operations. The core of the control/internet subsystem is an ESP32-S3-WROOM-1U, which interacts directly in some way with all other subsystems and runs code the user interacts with the entire time the device is powered on [1]. The power subsystem consists of a battery that supplies +3.7 V to linear voltage regulator, which in turns supplies the rest of the circuitry with +3.3 V. The memory subsystem consists of an SPI SD card reader and a micro SD card with 32 GB of storage capacity. This subsystem stores the raw text files which are parsed into flashcard questions and answers, as well as mp3 audio files and any data on user flashcard progress which needs to be retained upon power down. The user interface subsystem contains three simple buttons, two SPI digital ink screens, and an I2S amplifier and speaker pair. This subsystem contains sends information to and from the control/internet subsystem and allows for two way interactions between the user and the flashcards.

1.4 High Level requirements

At the start of this project 3 high level requirements were defined for the overall sucesss of the project.

- The device enables users to view flashcards, see answers, select their results, and monitor the status of a digital pet. Flipping and switching between flashcards must be completed within 1 second, and the digital pet should respond to any state changes within 1 second.
- The device must have the capacity to store and recall 'question and answer' data for up to 500 flashcards, in addition to retaining user interaction history with the flashcard set. Furthermore, it should be capable of downloading flashcard sets from the internet in under 5 minutes.

- The device should be portable, with dimensions not exceeding 160mm x 120mm, and designed for long-term use. It must feature a rechargeable battery with a lifespan of at least 2 hours on a single charge.

2 Design

2.1 Control/Internet Subsystem

2.1.1 Control/Internet Subsystem Description

The control/internet subsystem is the most complicated and most important subsystem of this project, as most of the other subsystems depend on the control system to function properly or function at all. The control/internet subsystem consists of the ESP32-S3-WROOM-1 module and all of its direct supporting circuitry, including programming and strapping circuitry. The tasks performed by this subsystem include storing and recalling flashcard data to/from an external SD card; tracking the user's flashcard progress and the state of the digital pet; sending/receiving control signals from the user interface subsystem including audio, buttons and screens; and retrieving date and time information from the internet. This subsystem also takes care of running code that dictates the user's flashcard experience. This includes deciding which flashcards to show the user and in what order, retrieving that data from the raw flashcard text, and stepping through cards as the user answers questions. The control subsystem needs to perform its job quickly and consistently to provide a smooth and comfortable experience for the user. Figure 2 shows the high level flowchart for the code run by the ESP32. Lastly, this subsystem was intended to be able to use the internet to download flashcards and corresponding audio clips, but this function was left unfinished as the timeline for the project expired.

2.1.2 Control/Internet Subsystem Design Details

The Control/Internet Subsystem exists entirely on our custom PCB, and the relevant schematics can be found in A.1 and A.2. The ESP32-S3-WROOM-1 provides the user with 2 functionally identical general purpose SPI channels, known as HSPI and VSPI. One of these channels will be used to communicate with the memory module, while the other will be used to communicate with the two digital-ink screens in the user interface module. The SPI channels can support running in half-duplex mode at up to 80 MHz, which is a much faster data rate than we will require for communication that will feel instant to a human user. In order to send audio to the user interface module, an I2S line will be used. The final design uses a 4 MHz serial clock, which is indeed fast enough for the time needed for an SPI interaction to be unnoticeable to a human. The buttons in the user interface module will have pull up resistors and will feed into GPIO pins of the ESP32-S3-WROOM-1. +3.3 V will be supplied to the subsystem by the linear voltage regulator in the power subsystem. This subsystem is expected to have the highest energy consumption of all subsystems. The ESP32 also features very high levels of pin reconfigurability, allowing us to essentially use any of the available pins for any purpose. The pins that were chosen for each function, visible in figure 9 in A.2 were picked for their physical location on the ESP32-S3-WROOM-1 as to avoid complicated traces and networks of vias on the PCB layout.

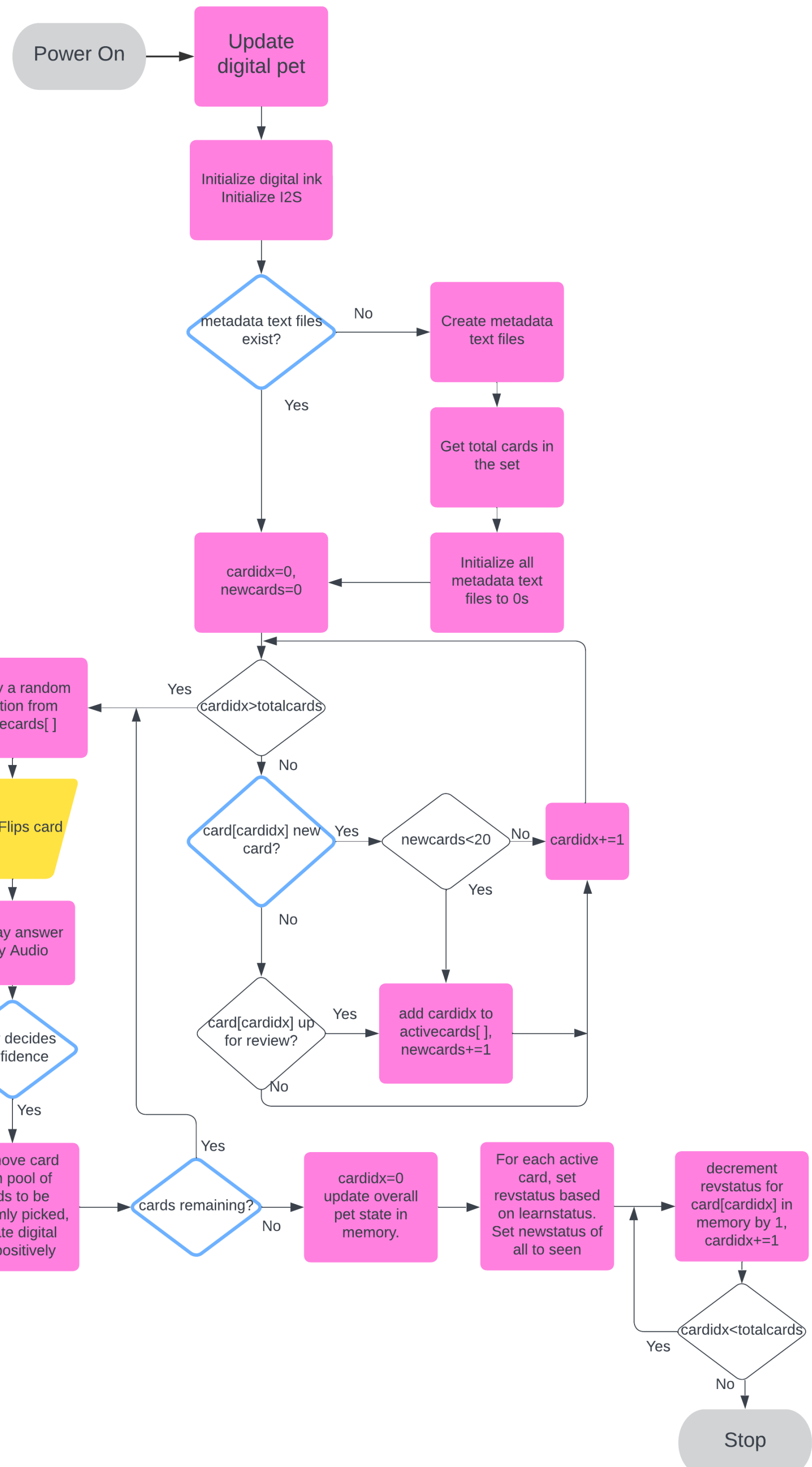

Figure 2: High-level MCU algorithm flowchart

2.2 Power Subsystem

2.2.1 Power Subsystem Description

The power subsystem is responsible for regulating and delivering power to the rest of the project. This subsystem will draw primarily from a battery, which needs to have the capacity to power the project for at least 2 hours. A battery was chosen due to the necessity of the project to be portable to fulfill basic requirements for success [2]. The power subsystem must also supply power at a steady 3.3 V, as all of the circuitry including the ESP32, I2S amplifier, and SD card reader all run on 3.3 V.

2.2.2 Power Subsystem Design Details

The power subsystem centers around a 3.7 V rechargeable lithium ion battery with 1000 mAh charge capacity. This corresponds to a total of 3600 coulombs, which at 3.7 V is a total energy storage of 13.32 kW from equation (1). An 1000 mAh battery was determined to be sufficient despite larger batteries being on the market due to the worst case power draw from table 1. In the worst possible case, the battery should be able to power everything for 1.27 hours, which is already close to the 2 hour requirement. This scenario is far from realistic, and in testing the battery powered the circuit for over 4 hours.The battery will be rechargeable through a 5 V micro-USB type B port, a common type of port used for phones and other small electronic devices. The battery output will be regulated down to 3.3 V by a linear voltage regulator. The part we have selected is the ADP3339, which is capable of outputting up to 1.5 A continuously. Table 1 also shows the worst case scenario for total current draw. Given the total worse case current draw calculated, 1.5 A is plenty to make sure that the linear regulator does not dip below the 3.3 V specification.

$$3600C \times 3.7V = 13320J \tag{1}$$

According to figure 3, with an input voltage of 3.7 V the ADP3339 can still output at above 3.294 V even when loaded to the maximum rating of 1.5 A. This is well within the

Current and Power Draw

Device	Worst Case current Draw (mA)	Worst Case Power Draw (mW)
Digital Ink Screen	18.7	61.7
I2S Amplifier	333	1099
ESP32	500	1650
SD card	30	99
Total	881.7	2909.01

Table 1: The worst case scenario for both current and power draw in mA and mW

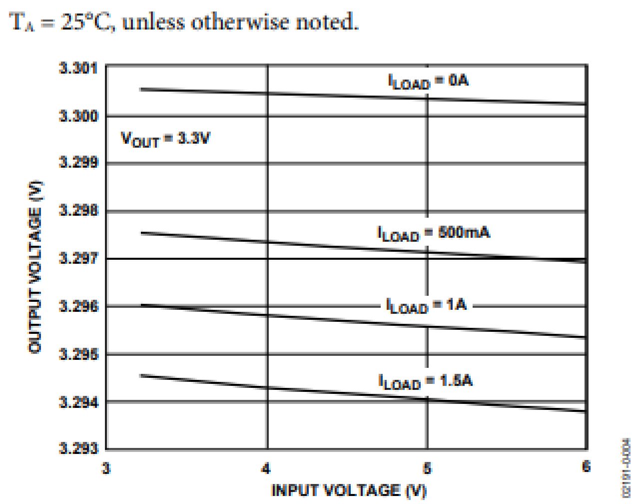

Figure 3: ADP3339 Voltage Input/Output curves with current load lines

requirements that we have listed in the requirements and verification section. The power subsystem also consists of a low power LED that signifies to the user when the device is powered on, which was not included in table 1 due to it being for debugging purposes and being unpopulated to save power for the final demo. Lastly, the power subsystem includes a switch that will power off the ESP32-S3-WROOM-1 and all of the powered user interface devices. This switch is mounted on the front panel of the device, allowing users to easily power the device on and off. For testing purposes, a secondary method of powering the electronics is implemented on the PCB. A 3 way solder jumper dictates whether the linear regulator is fed either by the battery or by a barrel jack connector. This second method of power was used for the majority of development. The full schematic for the power subsystem can be found in section A.3.

2.3 Memory Subsystem

2.3.1 Memory Subsystem Description

The memory subsystem is the least complex subsystem included in this project. However, it is still vital to the overall success of the project. The memory module consists simply of an SD card reader and a 32 GB micro SD card [3]. The SD card reader allows the SD card to read from and written to through an SPI interface, and it will interact with the control/internet subsystem to send information regarding either flashcard or user data when a request is made [4]. It will also store user progress data so that it is not lost upon power down. The original design instead included an on board 32 MB SRAM integrated circuit, also with SPI interface. The SD card was chosen as the largest design change made between the original design and final product. The change was made both to gain access to more memory per flashcard and to because SRAM is volatile memory that would have needed to remain powered at all times to avoid losing data. In order to store 500 flashcards on 32 MB of external memory, each flashcard would have been allocated 64 KB

Name	Date modified	Type	Size
mp3	5/1/2024 6:40 PM	File folder	
dateinfo	5/1/2024 6:40 PM	Text Document	0 KB
flashcards	5/1/2024 6:40 PM	Text Document	0 KB
learnstatus	5/1/2024 6:42 PM	Text Document	0 KB
newstatus	5/1/2024 6:42 PM	Text Document	0 KB
pethealth	5/1/2024 6:40 PM	Text Document	0 KB
revstatus	5/1/2024 6:40 PM	Text Document	0 KB

Figure 4: The root file structure of the SD card

of data, which is not enough to store both text data and an audio sample.

2.3.2 Memory Subsystem Design Details

A 32 GB SD card was chosen for several reasons. As mentioned before, SD cards are stable memory that will retain data through power loss. Furthermore, SD cards all innately support SPI reading and writing, with the SD card reader serving mostly as a conduit to route signals from pins on a breadboard or PCB to the SD card contacts. The SD card reader also includes 10 kΩ resistors that pull up all data pins to 3.3 V, which are required for the ESP32 to communicate with an SD card via SPI when using remapped pins. If 500 flashcards are stored, a 32 GB SD card would be able to partition around 64 MB per flashcard. This is vastly more than necessary to store a few lines of text for the actual flashcard text as well as a lengthy audio clip. The SD card also is responsible for storing data concerning the user's progress. Several files are generated by the ESP32 and saved on the SD card, shown in figure 4. The file newstatus.txt contains information about the new or seen status of an individual flashcard. This is represented as a string of 0s and 1s of the same length as the number of flashcards in the set. The file revstatus.txt is a string of 2 digit numbers which contains the days until a flashcard should be reviewed for each flashcard in the set. The file learnstatus.txt contains a string of numbers from 0 to 9 which represent the level to which a user has learned and become comfortable with a given flashcard. dateinfo.txt contains information about the most recent date the user has practiced, and pethealth.txt contains a single digit from 0 to 9 that represents the health of the digital pet. The folder mp3 and the file flashcards.txt are not generated, and must be added to the SD card manually. This was not the original design, and is the case only because the Control/Internet subsystem was left unfinished. Figure 5 shows an example flashcard set, which was used for the final demo. Question, answer, and mp3 file path are separated by a tab character and flashcards are separated by line. This structure easily allows the code running on the control/internet subsystem to retrieve the appropriate information about each flashcard. The full schematic for the memory subsystem can be found in section A.4, although because the subsystem is very self contained, the schematic only contains a single pin header that routes pins from the ESP32 to where the SD card reader will be mounted.

flashcards - Notepad

File Edit Format View Help

```
agradable       agreeable       /mp3/agreeable.mp3
amarillo        yellow          /mp3/yellow.mp3
anaranjado      orange          /mp3/orange.mp3
azul            blue            /mp3/blue.mp3
barato          inexpensive     /mp3/inexpensive.mp3
blanco          white           /mp3/white.mp3
debil           weak            /mp3/weak.mp3
delgado         slender         /mp3/slender.mp3
dificil         difficult       /mp3/difficult.mp3
el amigo        the friend      /mp3/friend.mp3
el banco        the bank        /mp3/bank.mp3
el barco        the boat        /mp3/boat.mp3
el cafe         the coffee      /mp3/coffee.mp3
el carro        the car         /mp3/car.mp3
el clima        the climate     /mp3/climate.mp3
el dia          the day         /mp3/day.mp3
el gato         the cat         /mp3/cat.mp3
el hermano      the brother     /mp3/brother.mp3
el hombre       the man         /mp3/man.mp3
el idioma       the language    /mp3/language.mp3
el jardin       the garden      /mp3/garden.mp3
```

Figure 5: Example of flashcards.txt

2.4 User Interface Subsystem

2.4.1 User Interface Subsystem Description

The user interface subsystem consists mainly of the pieces of the project that the user directly interacts with. This includes the two black and white digital ink screens, three face buttons, and speaker. The digital ink was originally planned to be a color screen, but larger power cost, price, and refresh time prompted a change to black and white. The user interface subsystem also includes an I2S amplifier which is an off the shelf part and also the only component to the user interface subsystem that the user does not directly interact with in some way. The I2S amplifier converts I2S signals directly from the control unit into an audio signal that is able to be fed directly into the speaker [5].

2.4.2 User Interface Subsystem Design Details

The digital ink screens are the Waveshare 3.52inch e-Paper HAT, 360 × 240, which will communicate with the control/internet subsystem via an SPI interface [6]. The screen has a specified 1.5 second refresh time, but in testing refreshes much faster than this, timing from when the screen begins to change to when the new image is fully shown. The full schematic for the user interface subsystem can be found in section A.5, but again mostly consists mostly of pin headers that route signals from the ESP32 to the digital ink screens and I2S amplifier. The exception to this is the face buttons, which contain more involved circuitry. The face buttons make use of 1 MΩ pull down resistors, such that when the button is not pressed the button pin connected to a GPIO port of the ESP32-S3-WROOM-1 is grounded and no power flows. When the button is pressed, the pin is shorted to 3.3 V. A very large pull down resistor is used so that when the button is pressed, a very small amount of current flows through the resistor and is wasted.

$$\tau = R \times C = (1000)(0.1 \times 10^{-6}) = 0.0001\text{seconds} \tag{2}$$

To prevent any other unwanted signals from being input each button is equipped with, a debouncing circuit consisting of a 1 kΩ resistor, 0.1 μF capacitor, and Schmitt trigger buffer. This results in a time constant of one tenth of a millisecond, which will create an effective debouncing circuit that will not create a notable delay for the user.

3 Verification

3.1 Control/Internet Subsystem Verification

The verification for the control/internet subsystem was largely performed through the verification of other subsystems in the overall design. The first verification from table 2 in section B.1 was unfortunately unable to be met. This requirement dictated that the control/internet subsystem would be able to download flashcard sets from a web server. Due to the project time period expiring, this requirement was not able to be verified.

The second requirement describes accurately reading button presses via GPIO pins. This requirement was verified multiple ways. Firstly, when a button was pressed, a multimeter was used to verify that the pin of interest was brought to a voltage of 3.3 V $\pm$ 0.1 V. Secondly, a serial print command was coded to output a message to the serial monitor whenever a button was pressed. Which button was pressed was also printed to verify accuracy. These tests were passed by the device.

The last requirement was to facilitate communication between the other subsystems. This was verified simply by observing a working interaction between subsystems. In the case of the user interface subsystem, this consisted of correctly displayed images on the digital screen and audio coming through the speaker. For the memory subsystem, this consisted of writing a file to the SD card and reading it back, printing the result on the serial monitor. The SD card was then plugged into a computer to verify that the file had been correctly written. A test with an oscilloscope to ensure data was being sent across the SPI line was also conducted. Figure 6 shows the result of one of these tests where SCK was monitored. These tests were all passed.

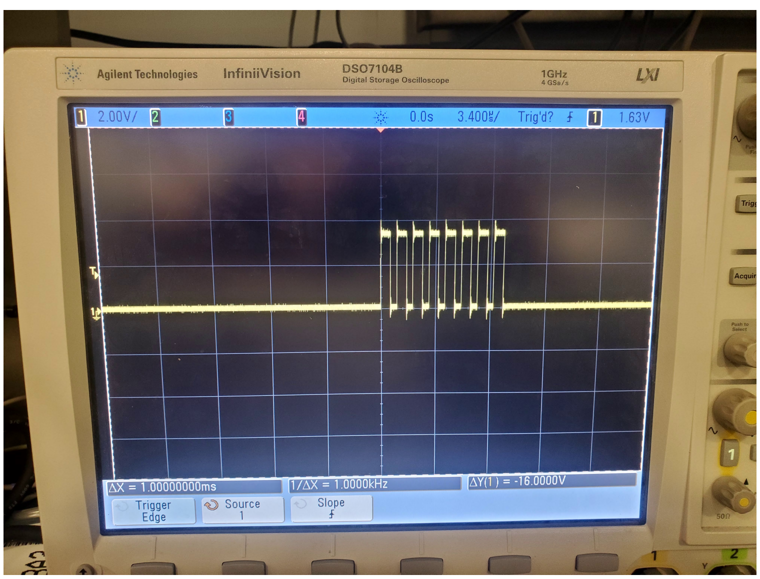

Figure 6: Oscilloscope trace of SCK for a SPI data transfer

3.2 Power Verification

The power subsystem required a consistent output of 3.3 V with at least 500 mA of continuous current supply when the device was switched on. The verification is described in table 3 section B.2 Verification occurred over multiple tests, the first of which was simply measuring 3.3 V via a multimeter which was successful when verifying.

After confirming that this first test was verified, this test was repeated after pushing a button to make sure this consistency doesn't decrease when swapping screens or sending data from the SD card. For this second test, the oscilloscope did not peak over 3.3 V and stayed consistently at 3.25-3.3 V which verified that the ESP32 would be powered sufficiently.

To test the battery life, we utilized the onboard LED to determine that the battery was fully charged. Once fully charged, the LED turned off, and we left the device on for more than three hours, which is significantly longer than the two hours we determined we would need for the device.

3.3 User Interface Subsystem Verification

To verify our user interface subsystem, the process is straightforward in identifying any issues. The verification table can be found as figure 4 in section B.3. The primary requirement is to ensure that users can interact with the flashcards smoothly. By using the device, we can confirm that it displays the correct flashcards each round, plays the word audio clips, and flips them within a short period.

Another aspect to verify is our pet response. Our digital pet must show a response on the screen immediately after the user's input and display the health status after the user completes daily practice and every time a user responds to a flashcard. This is verified by using a simple and short flashcard library. Working through this library of flashcards while responding to each question differently across many practice sessions allowed us to see if our pet would respond properly. Figure 7 shows the response of the digital pet to the user answering positively to a flashcard question. This requirement was verified.

After passing the previous test, we verified the functionality of each button by pressing them individually and observing the output on the console to ensure that each press elicits a proper response. We then rapidly pressed each button to check if each press is registered as a distinct action without any unintended multiple inputs.

3.4 Memory Subsystem Verification

To verify the memory subsystem in our device, we focus primarily on data handling and storage efficiency. The verification table can be found as figure 5 in section B.4.

The first requirement was that the device must download data for 500 flashcards from the internet within five minutes and store the data efficiently. Due to time constraints, we were unable to complete the internet function; therefore, we opted to load them directly

Figure 7: Digital ink screen correctly displaying the digital pet

via the SD card module. Given more time, it is entirely possible that the entire dataset could be downloaded to the device quickly and efficiently through the internet as we were able to achieve fast loading times from the SD card as well as a wireless connection through a hotspot. While the times to load data from the SD card was instantaneous (¡ 0.1 s), the time to load the data onto the screen took significantly longer. In terms of storage, storing full data for 20 flashcards required about 1 MB of storage space. This means that the 500 flashcard bench mark is easily achievable with the current storage situation.

Additionally, verifying the recall functionality of the stored data is crucial, and is the second and final verification for the memory subsystem. For this purpose, we created a custom flashcard set and simulated retrieval requests to the control module to ensure that the data could be accurately recalled. We generated and sent requests for specific flashcards within the set and monitored the system to verify that the correct data was retrieved each time. By tracing via an oscilloscope, we were able to correctly determine that signals were being carried across SPI lines as well as via the serial monitor within Arduino IDE through the TX and RX pins on the ESP32.

4 Costs

4.1 Parts

Table 6 in section C contains our costs for all parts we used during development and for the final product.

Miscellaneous surface mount components such as resistors, capacitors, and n-type MOSFET transistors are also part of the design, but as they were sourced from the ECE supply center, the exact cost cannot be calculated. Passive components can be bought for a few cents each, and small n-type MOSFETs can be purchased for around 50 cents on sites such as Mouser or DigiKey. A quantity of $5.00 for these components is a reasonable estimate that would easily cover these costs. This brings the total cost of parts up to $124.30. It should be noted that this is the cost of producing a single unit when purchasing small quantities from suppliers. If the JargonJolt were to undergo mass production, the cost per unit would go down significantly.

4.2 Labor

We assume that a graduate from ECE at Illinois makes $35 per hour. We estimate that each member spent around 10 hours per week for 10 weeks on this project. With each member working 100 total hours, this brings the labor cost to a total of $10,500.

4.3 Other Miscellaneous Costs

Other than parts with manufacturer part numbers that are directly purchasable from sites such as Digikey and Mouser, there are other costs associated with this project. The approximate cost of the PCB material of dimension 160mm x 100mm is about $12. For 3D printing the physical shell, assuming the total amount of print material used is approximately 300g, the estimated cost for this material will be about $35.

4.4 Grand

The total cost is estimated to be: $10,500 + $124.30 + $12 + $35 = $10,671.30

5 Conclusion

5.1 Accomplishments

Overall this project was a tentative success. Much of the functionality that we set out to achieve has been demonstrated, but the unfinished portions severely limit the project from being a truly completed project in its current state. However, the project is certainly recognizable as the idea that was formed at the beginning of the semester, and this project shows a strong proof of concept demonstration, and many of the shortcomings are due to time constraints. We feel confident that the original vision for this project could be achieved in full. At the conclusion of the semester, the project is capable of showing users flashcard questions and answers using a simplified spaced repetition algorithm, display the changing status of the digital pet as the user progresses, store user progress for future practice sessions, retrieve time and data information to determine user practice patterns, and playing audio clips along with flashcard answers. The project also fulfills its battery life and portability requirements, and is housed inside a neat 3D printed shell. Of the high level requirements listed in section 1.4, the first requirement, regarding user-flashcard interaction is fulfilled. The third requirement, which concerns portability is also met. The second requirement is only partially filled, as while the device certainly has the capacity to store well over 500 flashcards worth of data, it is unable to download them from the internet.

5.2 Uncertainties

The largest remaining uncertainty is the inability to download new flashcard data from the internet. This limits the user friendliness of the device, as currently the only way to load new data onto the device is to open the shell, remove the SD card, and add files through other means. A website where flashcards can be downloaded would greatly improve the device's ability to serve many different types of people, especially those uncomfortable with opening up electronics and doing even minimal work in a file explorer. An alternative system that would partially solve the problem would to make the SD card removable akin to a game cartridge. This would solve the issue of needing to open the electronics box, but not the issue of forcing users to manually add files to the SD card. If a web server were to be created, the user interface subsystem would also need to be updated to allow more complicated inputs for things such as typing the ID and passkey for a WiFi network and selecting which flashcard set to download.

The other significant uncertainty concerns supported fonts and characters. Currently, the JargonJolt uses Waveshare's library for displaying characters, and the included fonts only support letters of the roman alphabet and some special characters. Other fonts that include characters in other languages could be created to allow practice in a wider variety of languages.

5.3 Ethical Concerns

When developing JargonJolt at the University of Illinois, the IEEE Code of Ethics and Safety will be upheld including improving capabilities of emerging technologies (I.2), seeking criticisms of our technological developments (I.5), and crediting those who have contributed to any of our own developments (I.5) [7]. In addition to the IEEE Code of Ethics, the ACM Code of Ethics will also be upheld [8]. This project focuses on limiting the damages that electronics often have on the environment. By using a wide range of power-reducing techniques, JargonJolt will aim to limit the amount of energy needed in accordance with ACM's environmental sustainability principle (1.1). Should any issues arise, we will disclose any problems that our developments may cause in a transparent manner (1.3), all while respecting the privacy of our users' data. To mitigate any high risk factors in this project, all safety precautions will be carefully followed in accordance with safety manuals given by the manufacturers of determined high risk parts such as power supplies or any high voltage devices. Should any issues with parts arise to this regard, a safe procedure will be followed to ensure the defective or unsafe part is replaced, further assuring that the developed device is safe for any user to use.

One of the primary ethical concerns associated with our device pertains to potential copyright infringement. The development of our learning tool relies on various digital resources, including images of digital pets, word libraries, and audio clips, which were sourced from the internet [9] [10]. Utilizing these resources without proper authorization poses a significant risk of violating copyright laws.

To address this issue, it is imperative that we secure the necessary permissions for all copyrighted materials used in our device before it enters commercial production. Ensuring compliance with copyright laws not only protects our company from legal repercussions but also respects the intellectual property rights of the creators whose materials contribute to the functionality and appeal of our product.

5.4 Future Work and Impact

After evaluating our device, we have identified several improvements for future iterations. First, we can enhance portability by replacing the current screens with smaller ones, which would reduce the overall dimensions of the device as well as the costs.

Additionally, to support a broader range of language learning, we plan to expand our foreign font capabilities as mentioned in section 5.2. Initially, we intended to incorporate Japanese characters to cater to Japanese learners, but we faced challenges due to the unavailability of suitable libraries. If this product advances to mass production, it will be crucial to include comprehensive language support to accommodate diverse flashcard sets.

Furthermore, user preferences vary significantly, and a fixed font size may not meet all users' needs. To address this, we propose developing a more adaptable user interface that allows users to customize font size and voice volume, thereby enhancing the overall user experience.

If the JargonJolt were to be fully realized, it could fill a specific but very real niche in the community of language learners. The inspiration for this project, Anki, is already used by thousands of people. The JargonJolt would provide a tweaked experience tailored to children, those who do not have access to a personal computer, those who want to practice while commuting or otherwise not at home, and anyone who would rather have a dedicated practice device rather than downloading a computer or phone application.

References

[1] Espressif. "ESP32-S3-WROOM-1 Datasheet." (2024), [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf (visited on 04/18/2024).

[2] L. Wienclaw. "CHARGING TINYCIRCUITS LITHIUM ION POLYMER BATTERIES." (2024), [Online]. Available: tinycircuits.com/blogs/learn/charging-tinycircuits-lithium-batteries (visited on 01/24/2024).

[3] S. Santos. "ESP32: Guide for MicroSD Card Module using Arduino IDE." (2021), [Online]. Available: https://randomnerdtutorials.com/esp32-microsd-card-arduino/ (visited on 04/15/2024).

[4] S. Santos. "ESP32 SPI Communication: Set Pins, Multiple SPI Bus Interfaces, and Peripherals (Arduino IDE)." (2021), [Online]. Available: https://randomnerdtutorials.com/esp32-spi-communication-arduino/ (visited on 05/01/2024).

[5] Wolle. "ESP32-audioI2S." (2024), [Online]. Available: https://github.com/schreibfaul1/ESP32-audioI2S/wiki (visited on 04/09/2024).

[6] Waveshare. "3.52inch e-Paper HAT Manual." (2024), [Online]. Available: https://www.waveshare.com/wiki/3.52inch_e-Paper_HAT_Manual#ESP32.2F8266 (visited on 03/15/2024).

[7] IEEE. ""IEEE Code of Ethics"." (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 05/01/2024).

[8] ACM. "Code of Ethics - Association for Computing Machinery." (2018), [Online]. Available: www.acm.org/code-of-ethics (visited on 02/07/2024).

[9] BowTieCat. "BowTieCat." (2024), [Online]. Available: https://weibo.com/u/7600518442 (visited on 04/15/2024).

[10] Anki. "Easy Spanish Step-by-Step." (2015), [Online]. Available: https://ankiweb.net/shared/info/638144462 (visited on 04/15/2024).

Appendix A PCB Schematics

A.1 High Level Schematic

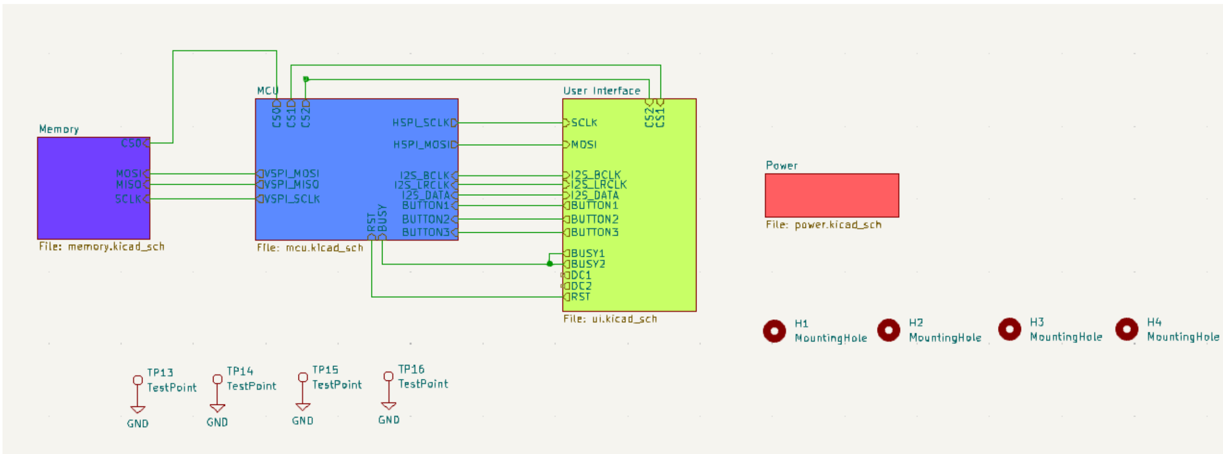

Figure 8: High Level Schematic

A.2 Microcontroller Schematic

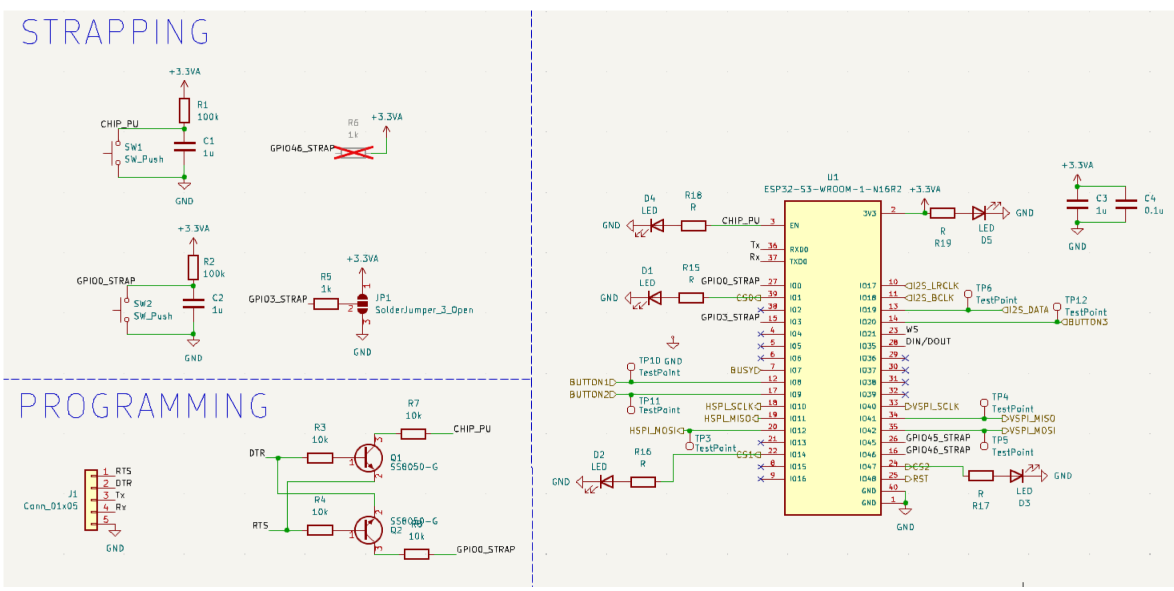

Figure 9: Microcontroller connections

A.3 Power Schematic

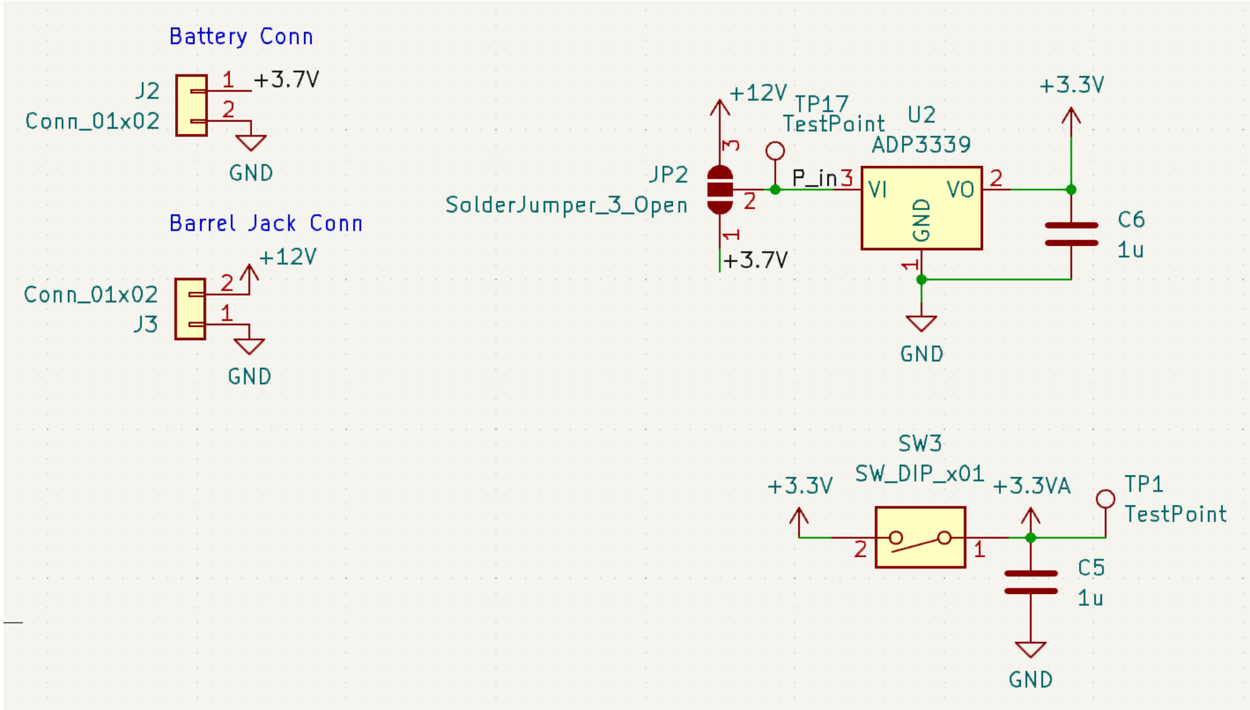

Figure 10: Power

A.4 Memory Schematic

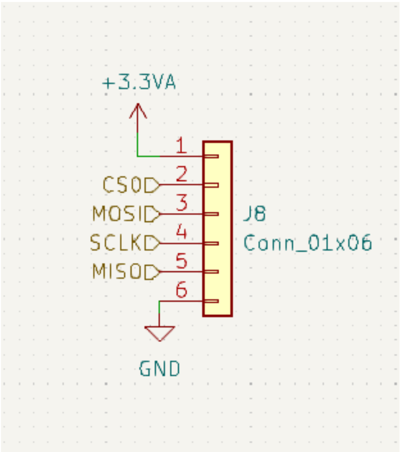

Figure 11: Memory

A.5 User Interface Schematic

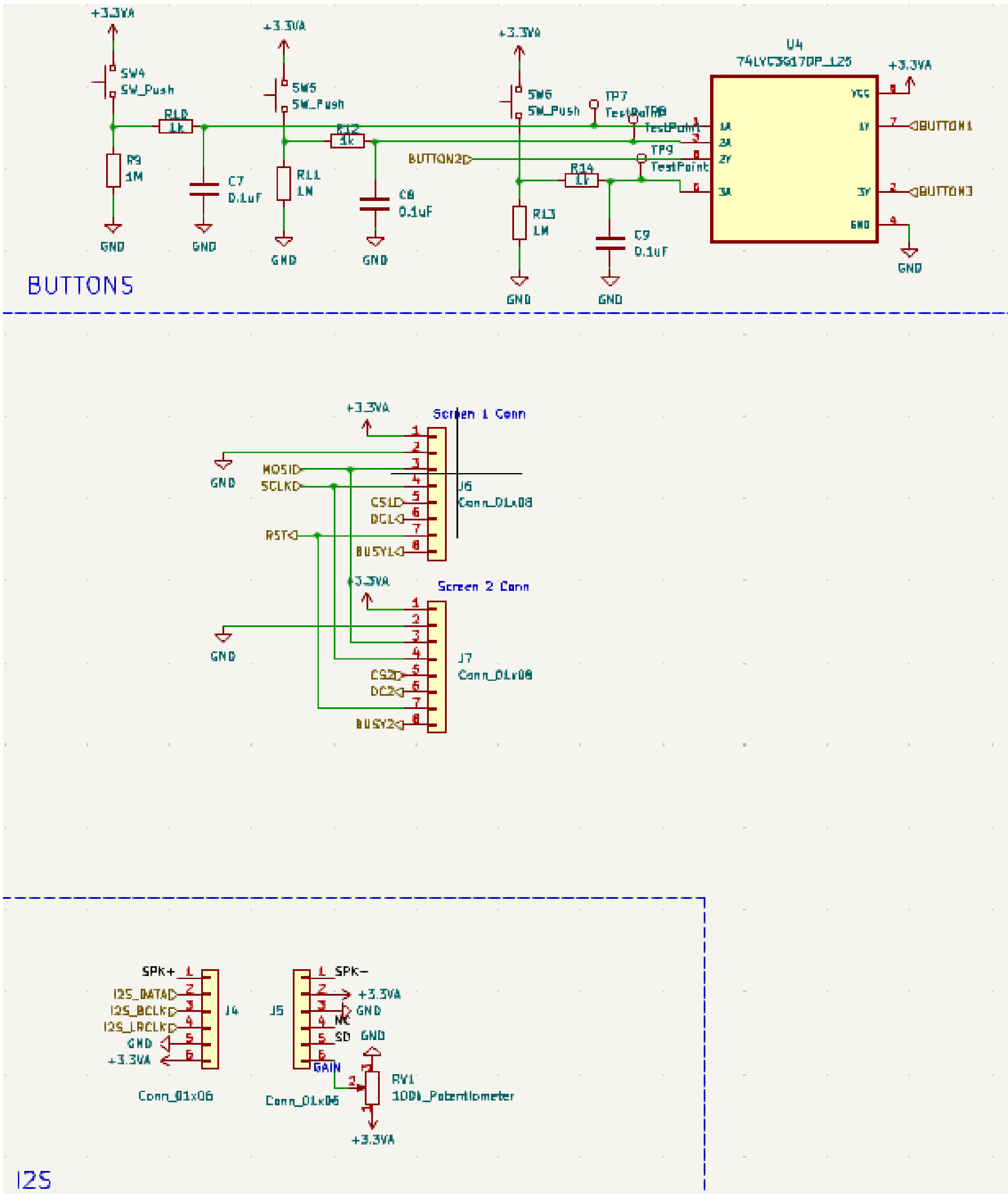

Figure 12: User Interface

A.6 Final PCB Design

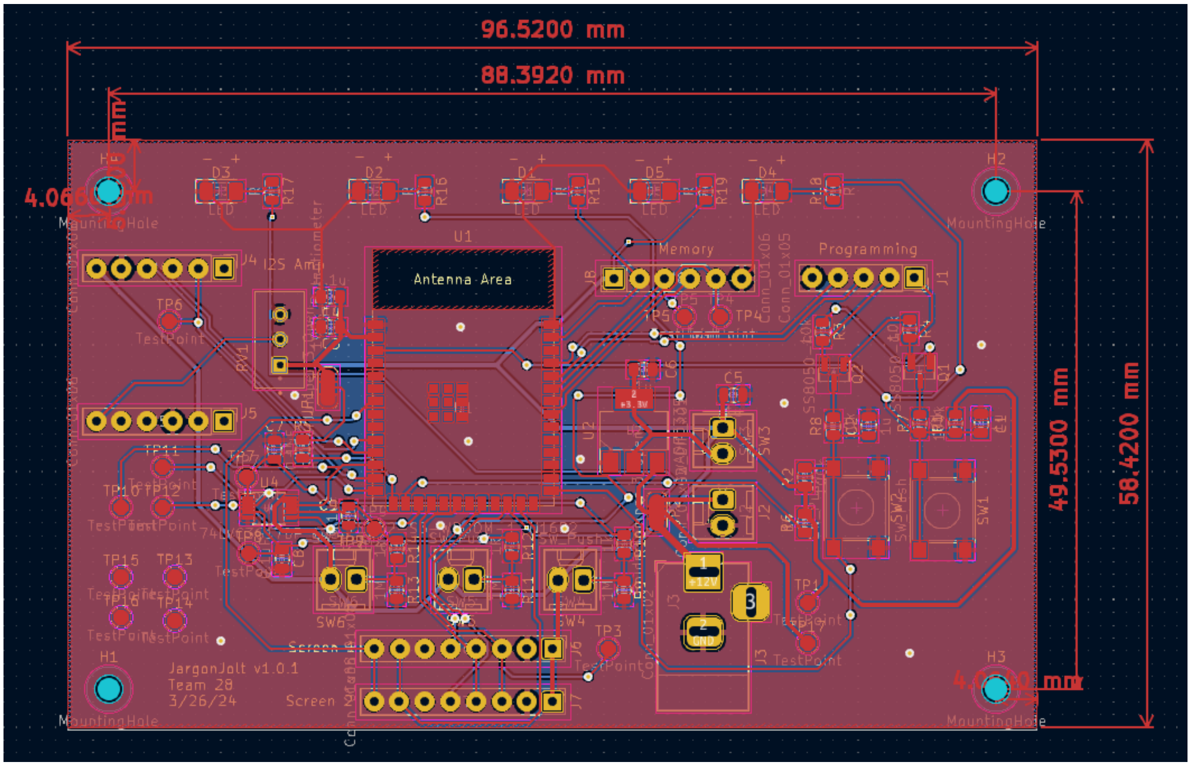

Figure 13: Final PCB Design

Appendix B Requirements & Verification Tables

B.1 Control/Internet Subsystem

Control/Internet Subsystem R&V Table

Requirements	Verifications	Verified
Communicate with a web server to retrieve flashcard set information.	• Ensure the MCU is in an empty state with no flashcards loaded with a constant voltage of 3.3V at the Vin pin. • Start the protocol to retrieve text and audio data from the web server. • Use a multimeter to ensure the VSPI_MOSI and VSPI_MISO data pins are transferring data when the corresponding chip select is on. • Check the HSPI_MOSI and HSPI_MISO data pins to ensure a voltage of 3.3 V +/- 0.1 V during data transfer. • Check the contents of the external memory by downloading the data back to the web server to ensure that data transfer protocols are functioning as intended.	no
Read user button inputs.	• Ensure all buttons send interrupts to the microcontroller. • Then, based on the button pressed, ensure that the corresponding status updates on the microcontroller. • BUTTON1, BUTTON2, and BUTTON3 pins on the MCU should all show a voltage of 3.3 V +/- 0.1 V upon being pressed in agreement with the MCU's max voltage tolerance.	yes
Facilitate communication between the memory and user interface modules including audio, display, and flashcard data.	• Ensure the MCU can first receive data from the web server. • Then, after updating a flashcard by clicking a button, check that the MISO/MOSI output properly from the VSLI and HSLI lines on the ESP32 which can clearly be seen via the digital screen. • Then, ensure that the audio is functioning by following I2S protocol and probing the data pins.	yes

Table 2: Verification of Control/Internet Subsystem Requirements

B.2 Power Subsystem

Power Subsystem R&V Table

Requirements	Verifications	Verified
When the device is idling, the Power Subsystem must be able to supply at least 500 mA continuously at 3.3 V +/- 0.1 V.	• Ensure the device displays the pet and the flashcard • Then, use a multimeter to measure the resistance across the power supply by connecting the probes to the input and the output of the Linear Voltage Regulator and selecting the 'Resistance' option. • Then, obtain the DC voltage out from the power supply using the multimeter using the same probes as before and selecting the 'DC Voltage' option to ensure a voltage of 3.3 V +/- 0.1 V. • Then, use Ohm's Law to verify that the current out of the linear voltage regulator is at least 500 mA.	yes
When the device undergoes data transfers or other short-term actions like screen updates, be able to supply at least 1 A for short periods of time to the rest of the system at 3.3 V +/- 0.1 V.	• Ensure the device displays the pet and the flashcard • Then, after causing a change from the idle mode, use a multimeter to measure the resistance across the power supply output by connecting the probes to the input and the output of the Linear Voltage Regulator and selecting the 'Resistance' option. • Then, obtain the DC voltage out from the power supply using the multimeter using the same probes as before and selecting the 'DC Voltage' option to ensure a voltage of 3.3 V +/- 0.1 V. • Then, use Ohm's Law to verify that the current of the linear voltage regulator is at least 1 A.	yes

Must allow recharging through the use of a USB charging cable with a battery life of at least 2 hours.	• Ensure that the led on the charging board turns on when charging. • Then, when fully charged, ensure that the status LED turns off. • Then, while the device is in idle mode, ensure that the screen stays on for at least 2 hours.	yes

Table 3: Verification of Power Subsystem Requirements

B.3 User Interface Subsystem

User Interface Subsystem R&V Table

Requirements	Verifications	Verified
Must allow the user to interact with the flashcards.	• Ensure that words are properly demonstrated on the screen. • Then, that the user can give positive/negative feedback on the current words by buttons. • Then, that the user can choose/exit different word units. • Then, ensure that the audio from sent from the MCU outputs properly.	yes
Must be able to show the digital pet's status on the screen.	• Ensure that the digital pet can generate different responses according to the user input on each flashcard. • Then, that the digital pet can demonstrate different statuses according to the flashcard learning progress.	yes
Must have accurate debounced buttons.	• BUTTON1, BUTTON2, and BUTTON3 pins should be debounced without any improper inputs.	yes

Table 4: Verification of User Interface Subsystem Requirements

B.4 Memory Subsystem

Memory Subsystem R&V Table

Requirements	Verifications	Verified
Be able to store data for 500 flashcards in 5 minutes or less.	• Create a set of 500 flashcards complete with questions, answers, and audio and ensure download can complete. • Time the download to make sure it completes within the allotted time.	partial
Send flashcard data back to the control module upon request.	• Create a custom flashcard set and ensure that data can be recalled. • Create requests for specific flashcards in the set and ensure the correct data is recalled.	yes

Table 5: Verification of Memory Subsystem Requirements

Appendix C Costs Table

Table of Part Costs

Part Name	Part Number	Manufacturer	Quantity	Cost($)
ESP32-S3-WROOM-1	ESP32-S3-WROOM-1-N16	Espressif	1	3.48
Micro USB-B Connector	10118194-0001LF	Amphenol ICC (FCI)	1	2.95
3.52 inch digital-ink screens	3.52 inch e-Paper (G)	Waveshare	2	39.98
Barrel Jack Connector	PJ-102AH	CUI Devices	1	0.70
Speaker	SP-1605	Soberton Inc.	1	1.95
3.7V 1000mAh Lithium Battery	ASR00012	TinyCircuits	1	9.95
Battery Charger	ASL2112	TinyCircuits	1	6.95
Linear Voltage Regulator	ADP3339AKCZ-3.3-R7	Analog Devices Inc.	1	5.25
MicroSD Card Reader Breakout	Micro SD TF Card Module	HNX DIY	1	0.63
I2S Amplifier	MAX98357	DFRobot	1	5.04
Schmitt Trigger	74LVC3G17DP,125	Nexperia USA Inc.	1	0.57
01x02 Connector	0022232021	Molex	6	1.26
01x05 Connector	PPTC051LFBN-RC	Sullins Connector Solutions	1	0.48
01x06 Connector	PPTC061LFBN-RC	Sullins Connector Solutions	2	1.04
01x08 Connector	PPTC081LFBN-RC	Sullins Connector Solutions	2	1.32

Connector Header Through Hole 40 position	HDR100IMP40M-G-V-TH	Chip Quik Inc.	1	0.69
SanDisk 32GB Extreme microSDXC UHS-I Memory Card with Adapter	SDSQXA2-064G-GN6MA	SanDisk	1	10.85
Push Button	7mm Mini Momentary Push Button	Gebildet	3	7.99
Tactile Switch	TS04-66-65-BK-160-SMT	CUI Devices	3	0.54
LED RED DIFFUSED CHIP SMD	AP3216ID	Kingbright	5	1.70
Miscellaneous*	N/A	N/A	N/A	5.00
Total	N/A	N/A	N/A	124.30

Table 6: Part Costs