Smoothie Recipe Maker

ECE 445 Final Report – Spring 2025

Team 49

Max Gendeh, Avyay Koorapaty, Anay Koorapaty

TA: Jason Zhang

Professor: Arne Fliflet

May 7th, 2025

Abstract

This report details our Automatic Smoothie Machine project for ECE 445. Specifically, we outline the design, implementation, and verification of the Control and User Interface (UI) subsystems, as well as the integration and processing of data from the Sensor subsystem. This involved programming the ESP32-S3-WROOM-2 microcontroller, developing the logic for ingredient dispensing based on load cell feedback, creating the user interaction logic via LCD and buttons, and ensuring accurate measurement conversion and display. This report includes the problem identified, our solution, design decisions, implementation details, and verification procedures performed. The overall project aims to automate the ingredient measurement process for smoothies.

Table of Contents

Introduction

1.1 Smoothie Market and Benefits

Smoothies are a healthy and nutritious drink utilized by gym goers and health-conscious individuals. Smoothies often have a variety of fruits blended together, which is beneficial for those who struggle to consume an adequate amount of fruits in its whole form. Blending fruits also helps to retain fiber content, and enables one's body to absorb nutrients more efficiently. The process of blending breaks down the cell walls of fruits, making it easier for your body to access the vitamins, minerals, and antioxidants present in the fruits [1].

Moreover, the smoothie market is rapidly growing. According to precedenceresearch.com, "The global smoothies market size was estimated at USD 13.71 billion in 2024 and is predicted to increase from USD 14.99 billion in 2025 to approximately USD 33.10 billion by 2034, expanding at a CAGR of 9.21% from 2025 to 2034" [2].

1.2 Problem

Making smoothies often requires measuring different ingredients with different measurement utilities. Liquid ingredients must be measured in mL, while solid ingredients are usually in units of cups or tablespoons. This can be a long process especially when the ingredients are numerous and are of various sizes, textures, and shapes. Additionally, the measurement steps are repeated for every subsequent smoothie made. To make the process more efficient, our design automates measuring and dispensing of ingredients. This simplifies creating smoothies with different recipes and reduces repetitive steps when creating more than one smoothie. The products currently on the market that automate the process of making smoothies are either too simplistic or not cost effective. The Ninja drink makers, although affordable, turn liquid ingredients into frozen drinks but do not provide the capability to deal with solid ingredients. Other smoothie machines such as Albert's smoothie station and the Milk station company smoothie station are essentially vending machines that are not commercially available. These machines likely cost upwards of $3,000 to buy. To address these issues, our system is simple, versatile, and cost effective.

1.3 Solution

Our system dispenses smoothie ingredients by following preset smoothie recipes or recipes the user creates. Our solution is a three compartment structure with two solid compartments and one liquid compartment. The two solid compartments have stepper motors to control the dispensing system while the liquid compartment has a solenoid valve controlling its flow rate. The ingredients are dispensed one at a time into a cup sitting on a load cell, which enables mass measurements. The motors and solenoid valve are part of the Actuation Subsystem, and the load cell is part of the Sensors Subsystem. To be able to create different kinds of smoothies, we have a UI with buttons and an LCD for inputting recipes and selecting preset recipes, creating the UI Subsystem. Complementing this, we have software to control the motors, read the load cell, display recipes properly on the LCD, and convert mass amounts to tangible measurement quantities like cups, mL, and tablespoons. This is the Control Subsystem.

These subsystems, Actuation, Sensors, UI, Control, and the Power subsystem to provide power to them all, work together to automatically dispense the appropriate amounts of ingredients into the cup for different recipes, making the ingredient measuring and placing process more efficient. The user can then empty the cup's contents into a blender. Our dispensing subsystem currently can only deal with very small ingredients such as grains and oats and wet ingredients like fruits cannot be utilized. To get around this fact, we could allow users to deposit ingredients into the blender cup directly and the LCD display would, in real-time, show the amount of each ingredient deposited.

1.4 Visual Aid

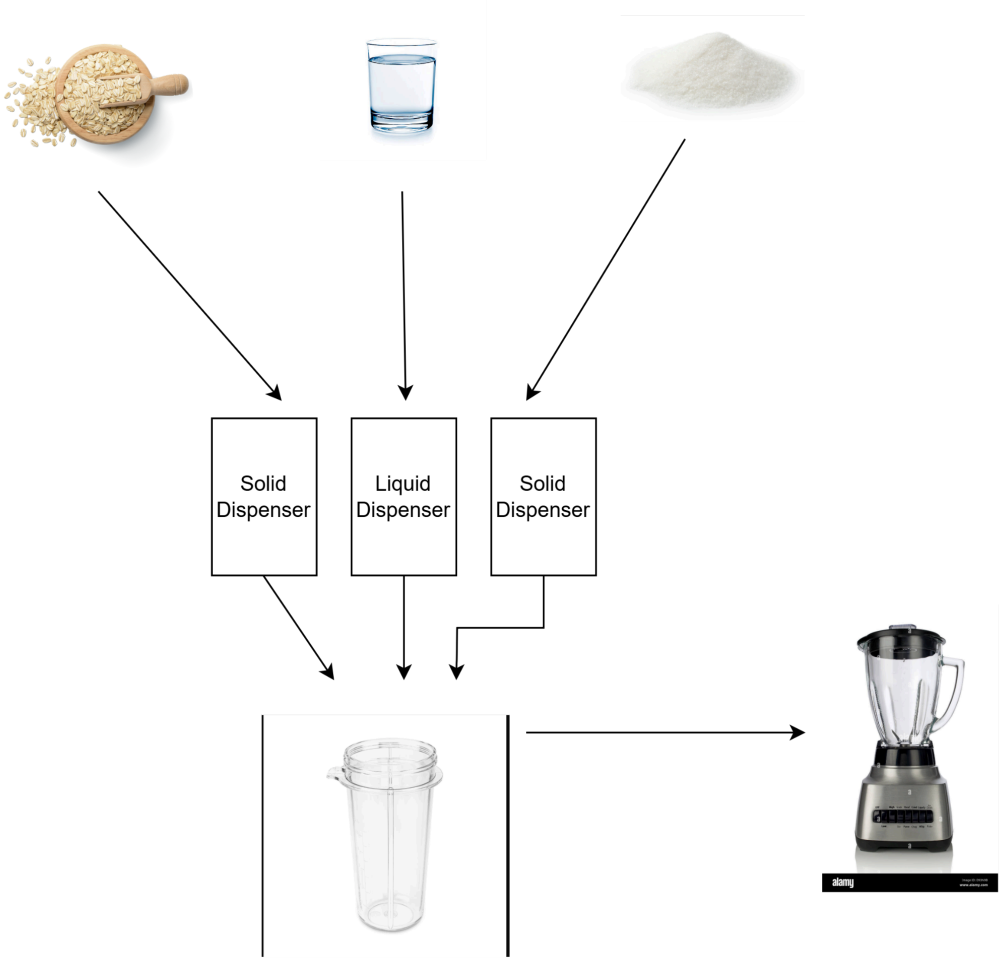

Figure 1.1: Visual Aid

1.5 Block Diagram

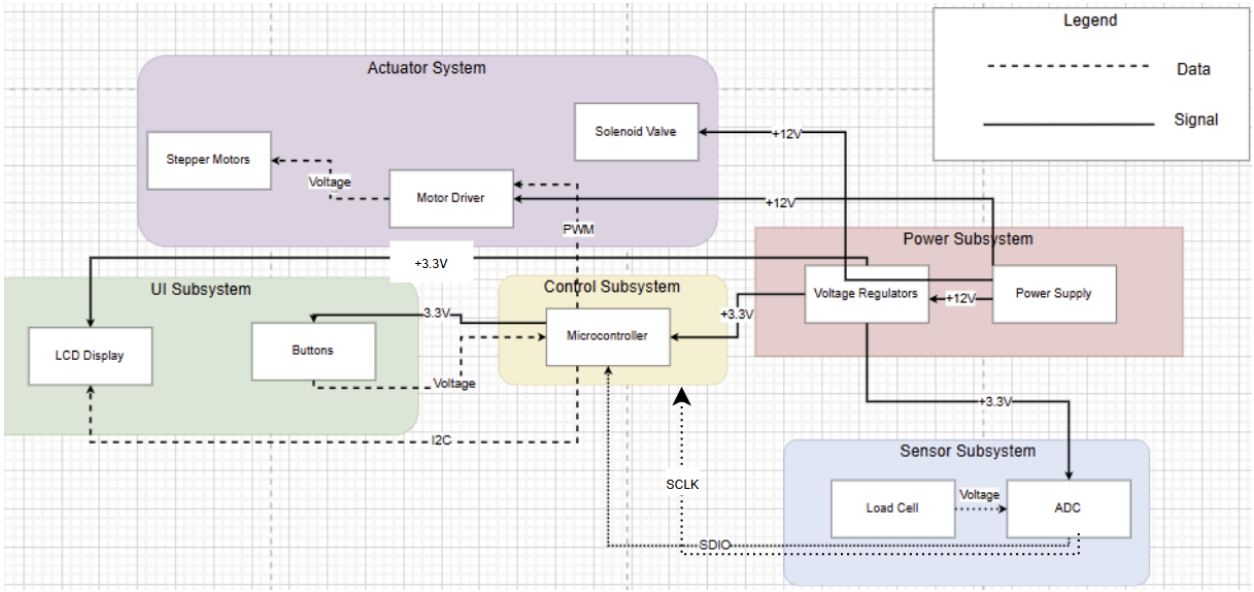

Figure 1.2: Block Diagram

The critical subsystems are the Actuator, Sensor, Control, UI, and Power subsystems. The Actuator subsystem dispenses the solid and liquid ingredients from their compartments into the cup sitting on the load cell. The Sensor subsystem measures the mass of each ingredient added using the load cell. The UI subsystem uses buttons and an LCD Display to allow the user to make their own recipes. The Control subsystem uses the information from the UI and Sensor subsystems to execute the dispensing of ingredients for the correct recipe and in the correct amounts, by controlling the Actuator subsystem. The Power subsystem provides power to all components.

1.6 High-Level Requirements

- The system takes a maximum of 1.5 minutes from the start of dispensing to all of the ingredients being inside the blender.
- Measurement accuracy will be to ±20% of intended ingredient amount.
- The system will dispense the correct ingredients given a preselected or user inputted recipe.

Design

2.1 Design Procedure

2.1.1 UI subsystem

For this subsystem we needed an interface to show and customize recipes, so we chose a 20x4 LCD display. To control changes on the LCD display, we chose to utilize 4 buttons. These buttons include up and down buttons to increment and decrement ingredient amount, and a next recipe and next ingredient button to switch between recipes and switch between ingredients in a recipe. We could have used a touch screen display and encapsulated the UI purely through software, but the hardware with the LCD and buttons was simple to wire along with programming our ESP-32 microcontroller through Arduino. Originally, we intended on purchasing an LCD without an I2C backpack attached to it. With this version our schematic would have 16 wires. With the I2C backpack the LCD only has 4 pins: VCC, SDA, SCL, GND, and our schematic would only have 4 wires. The buttons are active-low inputs, which the microcontroller can read to execute changes in the LCD display.

2.1.2 Sensor Subsystem

For this subsystem we needed a way to standardize our weight measurements. We had numerous weight sensors to choose from including strain gauge load cells, hydraulic load cells, pneumatic load cells, and capacitive weight sensors. In our application, we wanted a weight sensor that would output a differential voltage signal which we could easily convert through an Analog to Digital converter to be interpreted by the microcontroller. Hydraulic load cells output mechanical fluid pressure while pneumatic load cells output air pressure, both of which need a pressure transducer to convert to voltage. Capacitive sensors output a change in capacitance which again needs extra electronics to convert to voltage. These sensors add extra hardware to our design. This is why we chose a strain gauge load cell. Additionally, strain gauge load cells have high precision which we need to measure small ingredients. In our design, the load cell output is connected to the NAU7802SGI, a 24 bit Analog to Digital Converter. This is used to convert the differential voltage signal from the load cell to a number that can be interpreted by the microcontroller. A linear relationship can be developed between the weights of items and readings from the NAU7802SGI by using known weights of items and their readings. This relationship can be used to find the weight of any item. We utilized a 1kg load cell and found our known weights in grams. By standardizing the measurement of weight to grams, the corresponding conversion to ounces, cups, or tablespoons can be made based on a particular ingredient's conversion factor.

2.1.3 Actuation Subsystem

For this subsystem we needed actuators to dispense solid and liquid ingredients. For solid ingredients we wanted an actuator that would have high precision and turn to any angle we desired. DC Motors are only continuous rotation motors, and we reasoned that servo's would have too little torque to turn the oval disk that made up our dispensing system. So, we chose stepper motors for the dispensing of solid ingredients. For the liquid ingredient we needed a simple and inexpensive way to control flow rate. A 12V Normally-closed solenoid valve accomplished this task. It is wired in series with an N-channel

mosfet that turns on or off, making the solenoid valve turn on or off. When the solenoid valve is powered, liquid is allowed to go through. To control the stepper motor, we utilized the DRV8825 motor driver. Since we wanted to control two stepper motors and we didn't want to replicate the motor driver circuit which took up a lot of space and electronic components, we utilized a relay, an electromagnetic switch. This 4 pole double throw switch, through a one bit select line, dictated which stepper motor's wires fed into the motor driver. This method was satisfactory because we didn't need to control both stepper motors simultaneously.

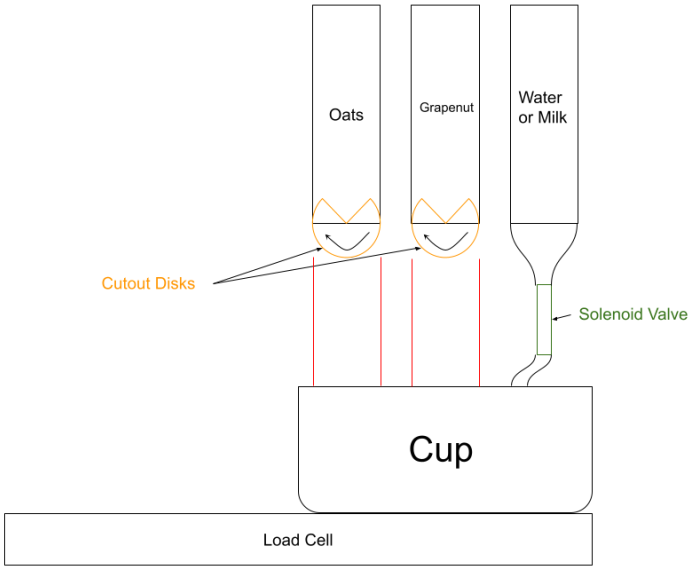

Figure 2.1: Original Mechanical Design

Figure 2.1 shows our original mechanical design with the cutout disks that our stepper motors would turn. These vertically rotating disks would load the ingredient when the cutout part was facing up, and unload the ingredient when the cutout part was facing down. This design was more difficult with large compartments, so we moved to the horizontally moving oval disks design. See Appendix C for a figure of the latest design.

2.1.4 Control Subsystem

For this subsystem we chose an ESP-32 WROOM 2 microcontroller. We chose this microcontroller because it has numerous programmable GPIO pins, supports I2C and UART communication, supports Arduino and has numerous software libraries, and has a low cost and high availability. In our design, the microcontroller interfaces with the UI, Actuation, and Sensor subsystems. It communicates through I2C communication protocol with the LCD display and NAU7802SGI Analog to Digital Converter. It transfers data to the LCD display and receives data from the NAU7802SGI. The microcontroller sends a PWM signal to the motor driver which controls the stepper motors. The microcontroller sends a digital high or low signal to the gate of the N-channel mosfet to turn the solenoid valve on or off, controlling when water flows.

2.1.5 Power Subsystem

For this subsystem we supplied 12 volts from an external power supply which was converted to 3.3V through a XL1509 buck converter. We chose a buck converter over a linear regulator because it consumes a lot less power when regulating a steep voltage drop. The 12 volts powers the solenoid valve and stepper motors, through the motor driver, and the 3.3 volts powers the lcd display, microcontroller, and analog to digital converter.

2.2 Design Details

2.2.1 UI Subsystem

Figure 2.3: LCD Recipe Format

Figure 2.4: Recipe I/O Buttons

The LCD display is hooked up to 3.3 volts through the VCC pin. The SDA and SCL lines are connected to GPIO pins of the microcontroller. One terminal of each of the buttons is connected to a GPIO pin of the microcontroller while the other terminal is connected to ground. On the LCD display shown in figure 2.3, the first row represents the name of the recipe, the second row represents the current ingredient within the recipe, and the third row represents the ingredient amount of that ingredient. The ingredient amount can be adjusted for each ingredient in the recipe using the up and down buttons, the top left and bottom left buttons respectively shown in figure 2.4. The ingredient amount changes in increments of 0.5g. The conversion to common units for each ingredient e,g ounces/cups/tablespoons is

shown in figure 2.4. There are preset recipes which cannot be modified and user recipes which can be modified. The recipe shown in figure 2.4 is a preset recipe. Once ingredient amounts are set, the recipe can be executed by holding the Next Recipe button, the top right button shown in figure 2.4, for approximately 3 seconds. See appendix C for UI subsystem schematics.

2.2.2 Sensor Subsystem

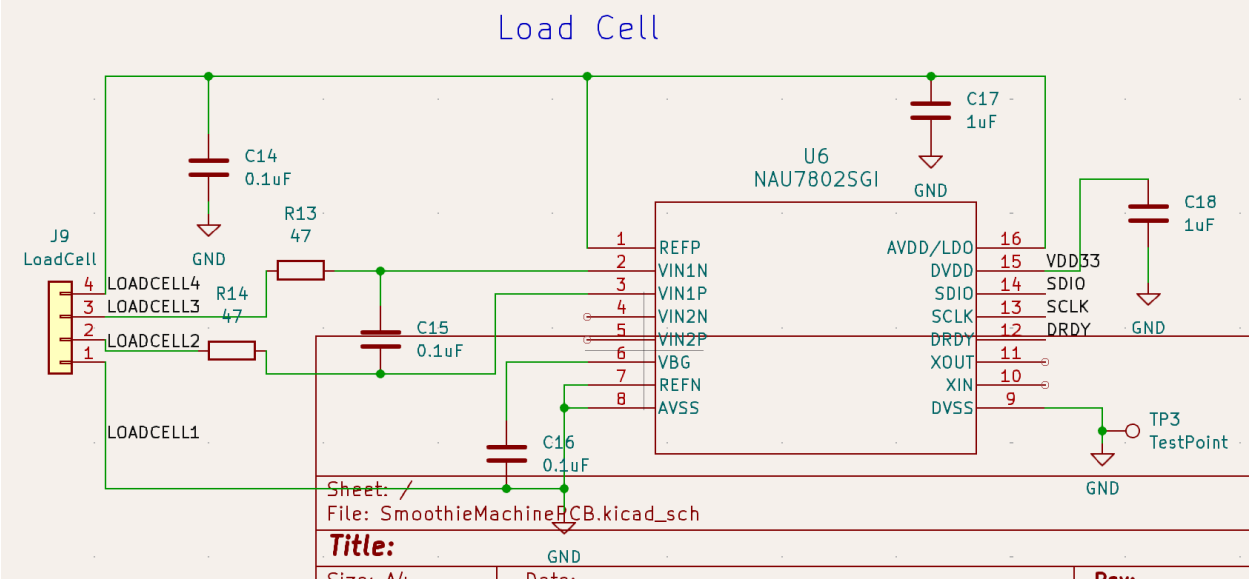

Figure 2.7: Load Cell Schematic

The load cell has 4 pins. Pins 4 and 1, as pictured in figure 2.7, correspond to the reference voltage supplied to the load cell, which goes into REFP and REFN pins of the NAU7802SGI respectively . Pins 2 and 3 correspond to the differential voltage signal output from the load cell, which goes into pins VIN1P and VIN1N of the NAU7802SGI respectively. The NAU7802SGI communicates via I2C with the microcontroller through its SDIO and SCLK pins. The DVDD pin of the NAU7802SGI, representing the digital voltage supply, must be at the same voltage used to power the microcontroller. The analog voltage supply has 6 programmable levels ranging from 2.7 volts to 4.5 volts. The NAU7802SGI outputs a 24 bit number between (-2^24 - 1) to (2^24 + 1) based on the differential voltage signal supplied to it by the load cell. There is a proportional relationship between the range of the differential voltage and the range of NAU7802SGI's numerical output.

2.2.3 Actuation Subsystem

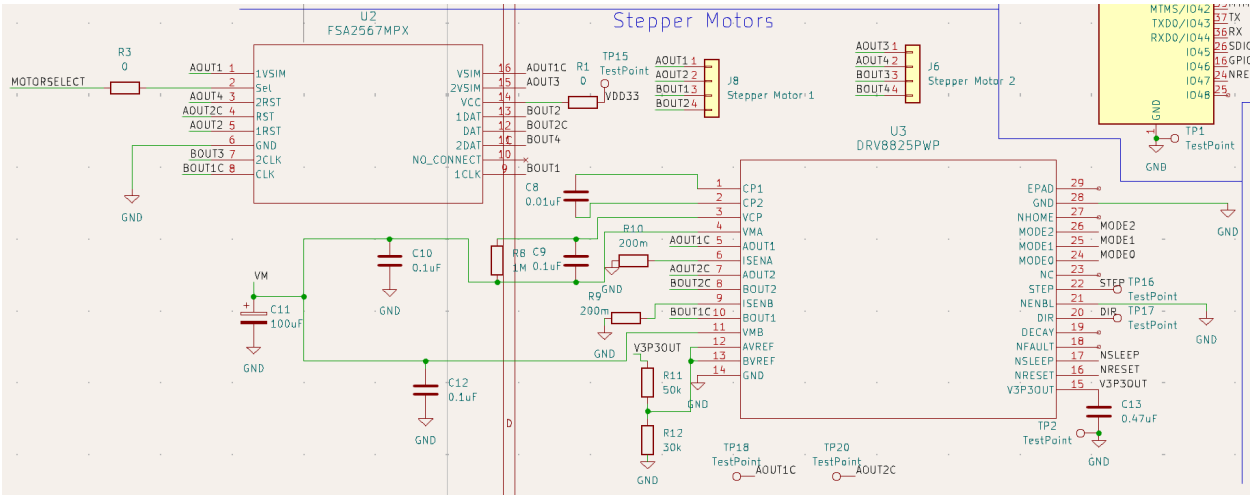

Figure 2.8: Stepper Motors Schematic

The DRV8825 motor is supplied 12 volts through the VMA and VMB pins with 0.1uF decoupling capacitors as shown in figure 2.8. There are 0.200 mOhm current sense resistors going into ISENA and ISENB as shown in figure 2.8. The 4 wires of the bipolar stepper motor go into AOUT1C, AOUT2C, BOUT1C, and BOUT2C where AOUT1C and AOUT2C represent one coil pair and BOUT1C and BOUT2C represent another coil pair within the stepper motor. The 8 important GPIO pins of the microcontroller on the right hand side of the driver are MODE 0, MODE 1, MODE 2, NSLEEP, NRESET, STEP, and DIR. MODE 0, MODE 1, and MODE 2 are signals used to control the microstepping mode of the motor. There are three signals packaged together, each of which can be high or low, contributing to 8 microstepping modes. Some modes include quarter step mode, half step mode, and full step mode. Since our motor has a step angle of 1.8 degrees, a single step pulse in full step mode would turn the motor 1.8 degrees but in half step mode would only turn the motor 0.9 degrees. NSLEEP and NRESET must be high for the motor driver to be on. The STEP signal is a PWM signal whose frequency determines the speed of the stepper motor. The DIR signal is a high or low signal used to control which direction the motor spins. Finally, the most important pin of the DRV8825 motor driver is the AVREF and BVREF pins shown in figure 2.8. We have a voltage divider circuit whose output is tied to those two pins as shown in figure 2.8. The max current that the stepper motor can draw through its windings is related to the voltage at AVREF and the current sense resistors through the following equation:

$$I_{chop} = \frac{Vref}{5 * R_{sense}} = \frac{Vref}{5 * 0.2} = Vref = max\_current \tag{2.1}$$

Our Nema-17 stepper motor is rated for 350 mA per phase. With the resistors pictured in figure 2.8 the Ichop = Vref is the following:

$$Vref = \frac{3.3}{R1 + R2} * R2 \tag{2.2}$$

$$R1 = 50k\,,\ R2 = 30k$$

$$Vref = \frac{3.3}{50000 + 30000} * 30000 = 1.2375V$$

This is too much current for our stepper motor and when we tried it on our PCB, the motor was vibrating and not turning smoothly. We changed the resistors to 10k and 1k where the Ichop = Vref was the following:

$$R1 = 10k\,, R2 = 1k$$

$$Vref = \frac{3.3}{10000 + 1000} * 1000 = 0.3V$$

This was within the current rating of our stepper motor and the motor turned smoothly.

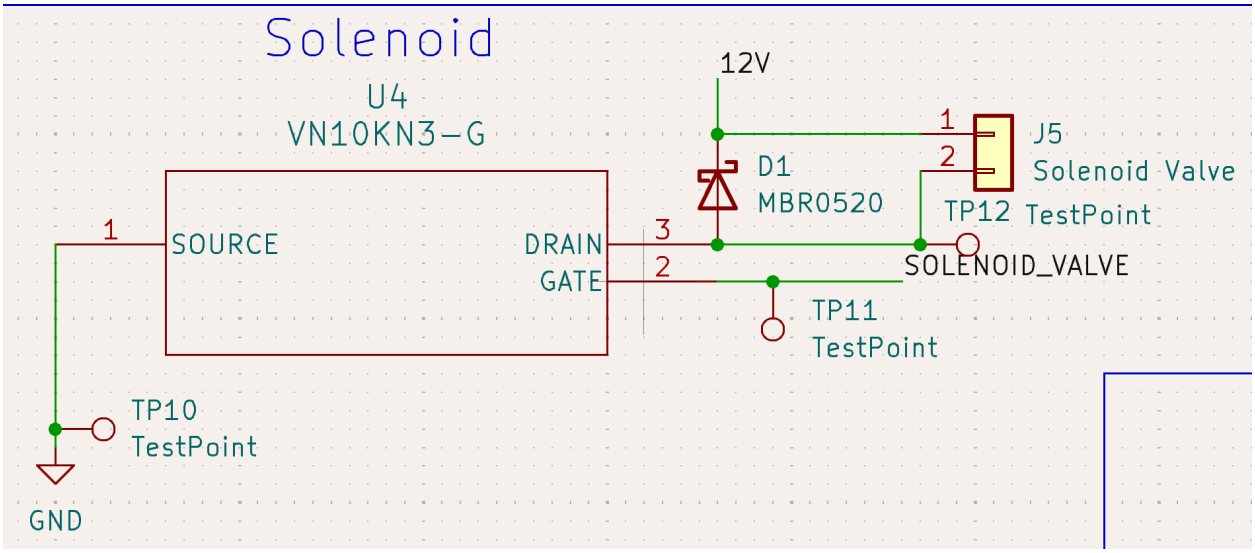

Figure 2.9: Solenoid Schematic

The solenoid valve is in series with an N-channel mosfet and contains a flyback diode across its terminals. The gate of the mosfet is connected to a GPIO pin of the microcontroller. The flyback diode is in place to prevent spurious voltage spikes by the inductive load, the solenoid valve, from damaging the mosfet. When the gate of the mosfet is high, it turns on and current flows through the solenoid valve, causing it to open. When the gate is low, the mosfet is off and no current flows through the solenoid valve and it remains closed. In our design, we used the VN10KN3-G which is not a logic level mosfet. At 3.3 volts, it is only partially turned on and the variable resistance of the mosfet causes it to heat up when current flows through it. To fix this circuit, we would replace the VN10KN3-G with a logic level mosfet that fully turns on at 3.3 volts.

2.2.4 Control Subsystem

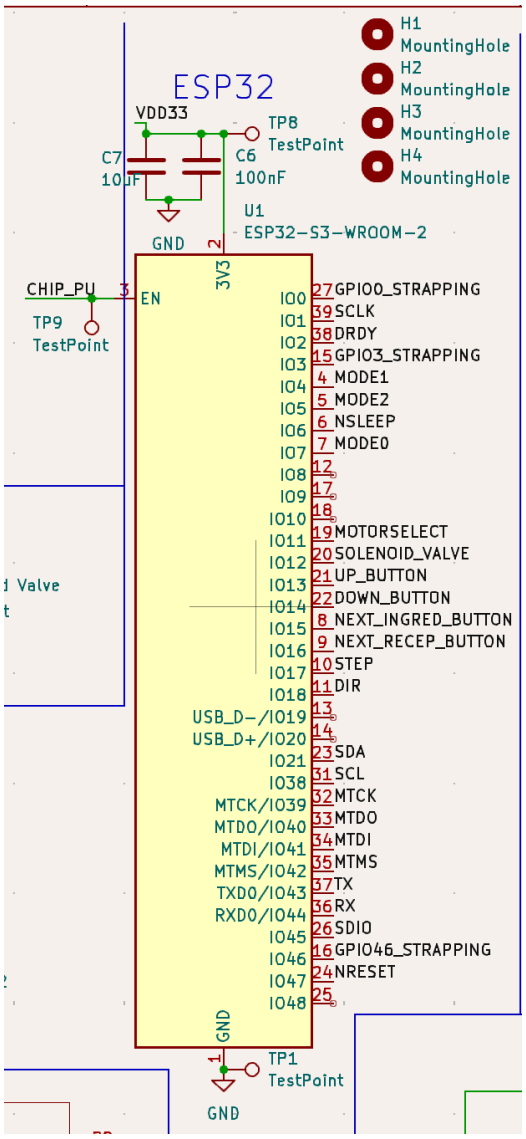

Figure 2.10: ESP32 Schematic

The schematic of the ESP32-WROOM-2 shown in figure 2.10 has labels for all the GPIO pins our design uses and indicative of what subsystem they belong to. These labels are described in the other subsystem sections. The microcontroller uses these signals in the Arduino software for the system.

2.2.5 Power Subsystem

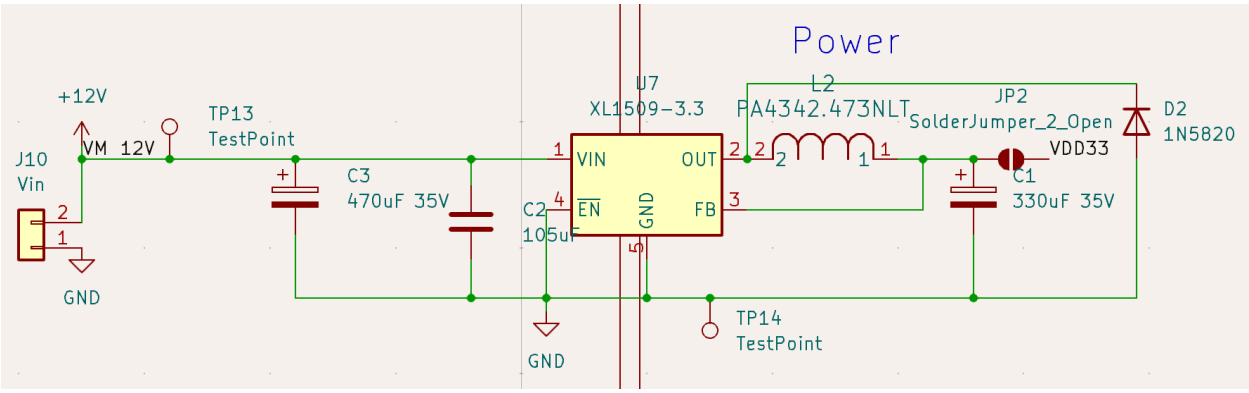

Figure 2.11: Power Schematic

The power subsystem converts 12 V from a wall supply into 3.3 V used by almost all of our components. The buck converter opens and closes the circuit with a duty cycle, which it varies to find the setting that converts to 3.3 best. It uses the feedback line, connected to FB, to do this. The inductor and capacitor on the right oppose a change in current and a change in voltage respectively, to stabilize current and voltage. The diode on the right prevents a reverse voltage spike from the inductor going back into the buck converter by providing an alternate path.

Verification

3.1 UI Subsystem

From viewing the LCD like in Figure 2.3, we verified that pressing the Up Button incremented the ingredient amount, pressing the Down Button decremented the ingredient amount, pressing the Next Recipe Button switched to the next recipe, and pressing the Next Ingredient Button switched to the next ingredient. We also made sure that decrementing below zero was not possible.

3.2 Sensor Subsystem

We calibrated the load cell by ascertaining the association between the numerical value returned from the NAU7802SGI ADC and the actual mass of items, by placing these items on the load cell and on a weighing scale. Our data is shown in Table 3.1 below. By plotting the data, as in Figure 3.2, we found the linear relationship between these variables, and the equation needed to convert the ADC's number into the corresponding mass value:

$$SM = \frac{LC - 211}{1030} \tag{3.1}$$

where SM is scale mass and LC is the ADC's number.

Table 3.1: Load Cell Calibration Table

Item	Scale Mass (g)	Load Cell 20 Values Average Number
Tape	53	54791
Screwdriver	115	118631
Pliers	186	191566.09
2A Stepper Motor	367	378410.19
2A Stepper Motor + Tape	421	433379
Cup	14	-113.9
Equation		
LC = 1030*SM + 211		
SM = (LC - 211) / 1030		

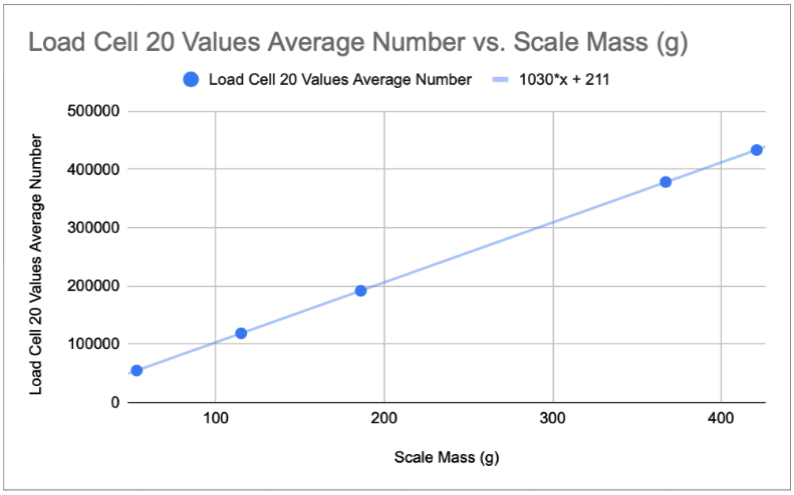

Figure 3.2: Load Cell Calibration Graph

3.3 Actuation Subsystem

The rpm requirement we initially had for the motors became not necessary as we moved to the newer design. With the original design of vertically rotating cutout disks, every revolution of the motor dispensed a little of the ingredient, but the new design started dispensing the ingredient with a small rotation of the motor, and stopped dispensing with a small reverse rotation of the motor. This removed the need for a quick rpm speed. The solenoid valve flow rate requirement we were unable to meet for installation reasons. The hose connected to the solenoid valve was installed in a very winding path, greatly decreasing its flow rate to a drip.

3.4 Control Subsystem

In order to dispense ingredient amounts accurate to plus or minus 10 g, we calibrated the software dispensing functions for oats and sugar, using this formula

$$stoppingpoint = (endmass - startmass) - \epsilon \quad (3.2)$$

Epsilon is subtracted so the stepper motor starts closing earlier than the user requested amount, to prevent overshooting the target amount. Some of our data is shown below. As shown in Table 3.3, we tested with a range of user input amounts and different epsilons, and found that

$$\epsilon = 19.6 - 0.095 * IngredientAmount \quad (3.3)$$

works best for oats, with a difference within plus or minus 10 g almost all of the time.

For sugar, we found that using half stepping of the stepper motor, with epsilon equals 7, works best with a difference mostly within 1 g and always within plus or minus 10 g.

Table 3.3: Oats Dispensing Calibration

Amount	Measured	Difference	Percentage Off	Epsilon
				Epsilon=13.5
14	22	8	57.14285714	
20	26	6	30	
28	28	0	0	
35	41	6	17.14285714	
44	44	0	0	
50	53	3	6	
14	18	4	28.57142857	
				Epsilon=19.6 - 0.095 * ingredient_amount
14	4	-10	-71.42857143	
20	18	-2	-10	
28	29	1	3.571428571	
35	31	-4	-11.42857143	
44	43	-1	-2.272727273	
50	52	2	4	
14	3	-11	-78.57142857	

Table 3.4: Sugar Dispensing Calibration

Amount	Measured	Difference	Percentage Off	Epsilon
				11 Half Steps Epsilon=4
5	9	4	80	
9	12	3	33.33333333	
14	17	3	21.42857143	
20	23	3	15	
28	31	3	10.71428571	
35	38	3	8.571428571	
44	48	4	9.090909091	
50	52	2	4	
14	16	2	14.28571429	
				11 Half Steps Epsilon=7
5	0	-5	-100	
9	8	-1	-11.11111111	
14	15	1	7.142857143	
20	20	0	0	
28	28	0	0	
35	36	1	2.857142857	
44	44	0	0	
50	50	0	0	
14	14	0	0	

3.5 Power Subsystem

Our wall supply outputs 12 V plus or minus 0.1 V, and the buck converter circuit successfully converts this to 3.3 V plus or minus 0.1 V. We originally thought the LCD required 5 V, but it actually needs 3.3 V, so the 5 V conversion requirement is not necessary.

Costs Analysis

4.1 Labor

Assuming $35/hr for labor, an average of 15 hours spent per week, and 15 weeks of work, our total is:

$$\$35 \times 2.5 \times 15 \times 15 \times 3 = \boxed{\$59,062.5}$$

4.2 Parts

Amazon:

No	Catalog/Part #	Description	Units	Qty	Unit Price	Ext Price
1	B016MP1HX0	DC 12V 1/4" Quick connect Solenoid valve	each	1	$7.49	$7.49
2	B098KMB3DJ	adafruit 1kg strain gauge load cell	each	1	$3.95	$3.95
3	B01GPUMP9C	SunFounder IIC I2C TWI Serial 2004 20x4 LCD Module Shield Compatible with Arduino R3 MEGA (IIC 2004)	each	1	$12.59	$12.59
4	B0B3884KGS	DRV8825 Stepper Motor Driver Module for 3D Printer Stepper Driver 4-Layer PCB Board	each	1	$10.99	$10.99
5	B00IJXZQ7C	HiLetgo FT232RL Mini USB to TTL Serial Converter Adapter Module 3.3V/5.5V	each	1	$6.48	$6.48

Total: $41.50

Mouser Electronics:

No	Catalog/Part #	Description	Units	Qty	Unit Price	Ext Price
1	356-ESP3S3WRM2N32R8V	ESP32-S3-WROOM-2-N32R8V	each	3	$0.21	$0.63
2	581-0805YD226MAT2A	Multilayer Ceramic Capacitors MLCC - SMD/SMT 25V 1uF Y5V 0805 20%	each	1	$3.95	$3.95
3	581-0805YD226MAT2A	Multilayer Ceramic Capacitors MLCC - SMD/SMT KGM21AR51C226MU NEW GLOBAL PN 16V 22uF X A 581-KGM21AR51C226MU	each	1	$1.04	$1.04

4	581-0805ZD474KAT2A	Multilayer Ceramic Capacitors MLCC - SMD/SMT 10V .47uF X5R 0805 10% Tol A 581-0805ZD474KAT4A	each	1	$0.54	$0.54
5	667-ERJ-UP6D3002V	Thick Film Resistors - SMD 0805 Anti-sulfur 0.5W, 0.5%, 30Kohm AEC-Q200	each	1	$0.28	$0.28
6	755-ESR10EZPF47R0	Thick Film Resistors - SMD 0805 47ohm 1% Anti Surge AEC-Q200	each	2	$0.20	$0.40
7	81-GRM21BR60J107ME5K	Multilayer Ceramic Capacitors MLCC - SMD/SMT 100 uF 6.3 VDC 20% 0805 X5R	each	2	$1.12	$2.24
8	667-ERJ-U6SJ20V	Thick Film Resistors - SMD 0805 0.25W 5% .20ohms AEC-Q200	each	2	$0.18	$0.36

Total: $9.44

Digikey:

No	Catalog/Part #	Description	Units	Qty	Unit Price	Ext Price
1	SW415-ND	SWITCH TACTILE SPST-NO 0.05A 24V	each	4	$0.52	$2.08
2	900-0022013027-ND	ONN RCPT HSG 2POS 2.54MM	each	10	$0.08	$0.78
3	1528-4538-ND	ADAFRUIT NAU7802 24-BIT ADC - ST	each	1	$5.36	$5.36
4	MBR0520LT3GOSTR-ND	DIODE SCHOTTKY 20V 500MA SOD123	each	3	$0.21	$0.63
5	1568-1511-ND	JUMPER WIRE M/F 6" 20PCS	each	2	$1.90	$3.80
6	FSA2567MPXTR-ND	IC SWITCH 4PDT 16MLP	each	2	$1.40	$2.80
7	1N5820GOS-ND	DIODE SCHOTTKY 20V 3A AXIAL	each	2	$0.38	$0.76
8	VN10KN3-G-ND	MOSFET N-CH 60V 310MA TO92-3	each	2	$0.55	$1.10
9	NAU7802SGI-ND	IC ADC 24BIT SIGMA-DELTA 16SOP	each	2	$1.46	$2.92
10	5272-XL1509-3.3-EVTR-ND	IC REG BUCK 3.3V 3A 8SOP	each	2	$0.71	$1.42
11	296-28915-5-ND	IC MTR DRVR BIPLR 8.2-45V 28SSOP	each	1	$5.36	$5.36
12	311-49.9KCRTR-ND	RES 49.9K OHM 1% 1/8W 0805	each	2	$0.10	$0.20
13	UUJ1V471MNJ6MTR-ND	CAP ALUM 470UF 20% 35V SMD	each	2	$1.30	$2.60

14	478-AEK1012331M035RTR-ND	CAP ALUM 330UF 20% 35V SMD	each	2	$1.00	$2.00
15	WM2748-ND	CONN HEADER VERT 6POS 2.54MM	each	2	$1.03	$2.06
16	732-2094-ND	CONN HEADER VERT 10POS 2.54MM	each	2	$0.39	$0.78
17	WM2011-ND	CONN RCPT HSG 2POS 2.54MM	each	12	$0.07	$0.80
18	WM2002-ND - B/O	CONN RCPT HSG 4POS 2.54MM	each	8	$0.18	$1.44
19	WM2015-ND	CONN RCPT HSG 6POS 2.54MM	each	2	$0.19	$0.38
20	1528-1961-ND	JUMPER WIRE F TO F 6" 28AWG	each	1	$1.94	$1.94
21	1965-ESP-PROG-ND	ESP-PROG, DEVELOPMENT AND DEBUGG	each	1	$12.50	$12.50
22	WM24402TR-ND	CONN SOCKET 20-24AWG CRIMP TIN	each	100	$0.05	$4.50
23	WM4111-ND	CONN HEADER VERT 2POS 2.54MM	each	12	$0.19	$2.24
24	WM4113-ND	CONN HEADER VERT 4POS 2.54MM	each	8	$0.36	$2.88

Total: $61.33

PCBway:

We planned on ordering 3 rounds of PCBs - totalling around $58

Total Estimated Cost

Factoring in labor and design, we estimate the total development cost to be $59,232.77.

Conclusion

We were able to successfully complete the project such that it meets all the high level requirements previously outlined. The machine showed full functionality of the recipe execution during the demonstration. We showcased the simplicity of the recipe UI, the functionality of the buttons, and the accuracy at which the machine performed.

We learned several lessons throughout this journey. The most important one by far was to start testing early. Had we tested the subsystems of each wave and not skipped the first wave, we could have had a working PCB that satisfied all functionality. Another important lesson we learned was to ask lots of questions. Had we done this we could have saved wasted iterations on useless PCB boards. For example, had we asked just a day earlier on how to actually program the microcontroller, we could have started testing on our second wave instead of having to wait for our third wave to arrive.

In the process of designing our project, we had to pass up several alternative implementations. We planned to have a lot more available ingredients, including fruits, beverages, and sweets. However, due to the constrictions of the machine shop, we were unable to carry them out. An example concept we had was a motor that pushed fruit through a slicer and into the cup. We also would have taken advantage of the ability to use multiple PCB boards to reduce clutter and improve packaging. Finally, we would have improved user friendliness through the use of sensors that informed users when ingredient levels were getting low and a thermocouple that warned users if a beverage may be spoiled.

Ethics and Safety

Our group is committed to complying with IEEE and ACM's Code of Ethics. We also take steps to ensure the safety and security of our users.

For full details, please refer to Appendix B.

References

[1] D. SEPULVEDA, “Blending Fruit vs Eating Whole Fruit,” *Shake Please*, Oct. 05, 2023. https://shakeplease.com/blogs/news/blending-fruit-vs-eating-whole-fruit

[2]“Smoothies Market Size to Hit USD 30.63 Billion by 2033,” *www.precedenceresearch.com*. https://www.precedenceresearch.com/smoothies-market

[3] IEEE-CS. “Code of ethics,” IEEE Computer Society, https://www.computer.org/education/code-of-ethics (accessed Feb. 12, 2025).

[4] “Oatmeal PNG transparent images,” PNG All -, https://www.pngall.com/oatmeal-png/ (accessed Mar. 6, 2025).

[5] A. satya, “Download strawberry slices on a transparent background, PNG for free,” Vecteezy, https://www.vecteezy.com/png/11198730-strawberry-slices-on-a-transparent-background-png (accessed Mar. 6, 2025).

[6] B. Foundation, “Home of the blender project - free and open 3D creation software,” blender.org, https://www.blender.org/ (accessed Mar. 6, 2025).

[7] K. Beck, “How to determine the RPM on Stepper Motors,” Sciencing, https://www.sciencing.com/determine-rpm-stepper-motors-10033323/ (accessed Mar. 6, 2025).

[8] Vitamix. “Making Large-Batch Smoothies at Home.” vitamix.com. Accessed: Mar. 5, 2025. [Online]. Available: https://www.vitamix.com/us/en_us/articles/making-large-batch-smoothies-at-home#:~:text=A%20typical%20smoothie%20size%20is,you're%20good%20to%20go!

[9] Laumas. “How to choose a load cell? The factors to assess.” laumas.com. Accessed: Mar. 5, 2025. [Online]. Available: https://www.laumas.com/en/blog/guides/how-to-choose-a-load-cell-the-factors-to-assess/#:~:text=As%20a%20rule%20it%20is,much%20as%20100%25%20or%20more.

[10] Nuvoton. *NAU7802 24-Bit Dual-Channel ADC For Bridge Sensors*, *Revision 1.7*. (2012). Accessed: Mar. 5, 2025. [Online]. Available: https://www.nuvoton.com/export/resource-files/NAU7802%20Data%20Sheet%20V1.7.pdf

[11] cs137, adafruit_support_mike, dlleigh. “Load Cell Sensitivity.” forums.adafruit.com. Accessed: Mar. 5, 2025. [Online]. Available: https://forums.adafruit.com/viewtopic.php?t=191695

Appendix A

Actuation R&V

Requirements	Verifications
• Motors must be able to rotate the disk between 0 and 180 degrees with tolerance of 5 degrees.	• Stepper motor's step angle of 1.8 degrees means it cannot be off by more than 3.6 degrees, so tolerance is satisfied.
• Motors can run at speeds from 60 rpm to 180 rpm, in order to control dispensing rate.	• Run motor for ten revolutions, and measure the time it takes, then convert to rpm
• The solenoid valve must be able to control flow rate of water and milk to a minimum of 1 cup per 20 seconds.	• Test the following with water and milk. • Fill the container that feeds into the solenoid valve with liquid, with the solenoid valve closed. • Collecting the solenoid valve output in a standard unit sized cup, open the solenoid valve and start a timer at the same time. • End timer when cup is filled, checking that the timer value is under 20 seconds.

Sensors R&V

Requirements	Verifications
• Differential mass measurements, i.e. the measurement of the change in weight caused by adding some of an ingredient, must be accurate to +- 10 grams.	• Start with the cup on the load cell, then add a 100 gram object. • Check that the mass measurement, i.e. the result from the ADC converted to a mass value, is 100 grams more than just the cup, +- 10 grams. • Repeat for 50, 150, 200, 250, and 300 grams to confirm the requirement for a range of ingredient mass additions.
• The load cell must be able to make mass measurements up to 600 grams.	• Place objects with masses known to be 100, 200, 300, 400, 500, and 600 grams on the load cell. • Check that the mass measurement, i.e. the result from the ADC converted to a mass value, increments by 100 each time, confirming that the load cell can

	adequately make mass measurements through the range of masses.
● Calibrate load cell so that absolute weight measurements are accurate to +- 10 grams	● Use the voltage values from the ADC when zero grams are on the load cell and when close to the maximum number of grams are on the load cell, and figure out offset and gain needed for conversion from the voltage value to the absolute gram measurement, with this formula: $Mass = (voltage - offset) * gain$ ● Ensure that gram measurements of 100, 200, 300, 400, 500, and 600 gram objects are correct to within +- 10 grams, so our measurement range is correctly measured by the load cell.

Control R&V

Requirements	Verifications
● Software must start and stop motors and solenoid to dispense ingredient amounts accurate to +- 10g	● Take a preset recipe as an example and check the gram amounts of each ingredient ● Take note of the current weight of the load cell, and after an ingredient finishes dispensing, stop the software and look at the difference in weight. Check if this is +- 10g of expected weight in preset recipe
● Software is able to change speed of stepper motor from 60 rpm to 180 rpm	● In the software, set the speed of the motor to 60 rpm and count the time it takes to make 10 revolutions. Make sure this is close to 10 seconds ● In the software, set the speed of the motor to 180 rpm and count the time it takes to make 30 revolutions. Make sure this is close to 30 seconds
● Software should debounce a button press so that aren't multiple button presses registered within half a second	● Press button 3 to switch to preset recipes ● Press button 1 or 2 and make sure LCD display only changes once
● Software should display recipe name on	● Press button 3 and check whether recipe

first row of LCD, the current ingredient compartment on the second row, the amount in grams on the third row along with its conversion to ounces/cups/tablespoons	name is positioned on first row of LCD ● Press button 4 and check if ingredient compartment is positioned on second row of LCD ● After pressing button 4, press buttons 1 and 2 and check if ingredient amount shows up in grams, ounces, cups, and tablespoons on the third row of the LCD

UI R&V

Requirements	Verifications
● Pressing button 3 without first pressing button 4 will move to the mode for selection between preset and user-made recipes, which will be done using buttons 1 and 2. The recipe names will be displayed on the LCD.	● Press button 3 and check that first preset recipe shows up ● Press buttons 1 and 2 and see if different recipe names show up
● When in ingredient mass adjustment mode, the LCD should display ingredient amounts in real-time when buttons 1 and 2 are pressed in increments of 0.5 units	● Check that one press of button 2 increments ingredient amount by 0.5 units ● Check that one press of button 1 decrements ingredient amount by 0.5 units ● Holding down button 1 or 2 should continuously increment/decrement the ingredient amount and this should be clearly reflected on the LCD display
● Pressing button 4 allows the user to create their own recipe. Buttons 1 and 2 allow the user to change ingredient amounts and button 3 is to to lock the amount into the recipe. The amount of each ingredient will be specified with the LCD showing which compartment's ingredient amount is currently being adjusted.	● Check that pressing button 4 displays first ingredient compartment on LCD display ● Check that pressing buttons 1 and 2 change ingredient amounts and pressing button 3 switches to next ingredient compartment
● Ingredient amounts should be shown on LCD display in grams along with its conversion to standard units in smoothie recipes: ounces, cups, tablespoons within some error tolerance(e.g 5%)	● Check that the numbers shown on the LCD display for ounces, cups, and tablespoons are equivalent using dimensional analysis with known conversion factors

Power R&V

Requirements	Verifications
• Power supply should output a constant voltage of 12V +- 0.1 volts.	Use a multimeter and place positive lead at output of power supply and negative lead at ground. Look at voltage reading.
• The first regulator should output 5V +- 0.1 volts. • The second regulator should output 3.3V +- 0.1 volts	Use a multimeter and place positive lead at output of the first regulator and negative lead at ground. Look at voltage reading. Use a multimeter and place positive lead at output of second regulator and negative lead at ground. Look at voltage reading.

Appendix B

Ethics and Safety

We will address the ethical and safety issues with reference to the IEEE and ACM Codes of Ethics and relevant regulatory standards [3].

Ethical Guidelines

Ethical guidelines require that we prioritize public safety and the well-being of users [3]. Because our project interacts directly with users, we want to provide full transparency of the capabilities of the machine. We aim to provide clear documentation, labeling intended use, limitations, and potential risks. An example is during an unlikely event when one of the ingredient chambers gets clogged, we'll let the user know on the LCD to try giving the machine a shake.

Safety Standards

We take user safety extremely seriously. Blenders are known to be dangerous and our design ensures that our product will not interfere with the blending process. Food safety is another concern. We aim to provide that through temperature sensors that let users know for example when the temperature of milk has exceeded a safe temperature for storage. Materials in contact with ingredients will be food-grade and easy to clean to prevent contamination. Finally, with the elimination of any batteries, we minimize the risk of fires and electrical hazards.

Regulations

We will adhere to campus safety and lab policies throughout the development of this project. Our team has set aside time for weekly reviews and feedback on our design throughout the project's lifecycle. With this, we will be able to collect valuable insight and feedback to ensure our project stays within relevant regulations and standards.

Respect and Compliance

Our team is committed to developing an environment of respect and ethical awareness. All members will treat all users fairly and hold each other accountable for any outcomes our project may have. Through this, we hope there will be efficient teamwork and good progress throughout the project lifecycle.

Appendix C

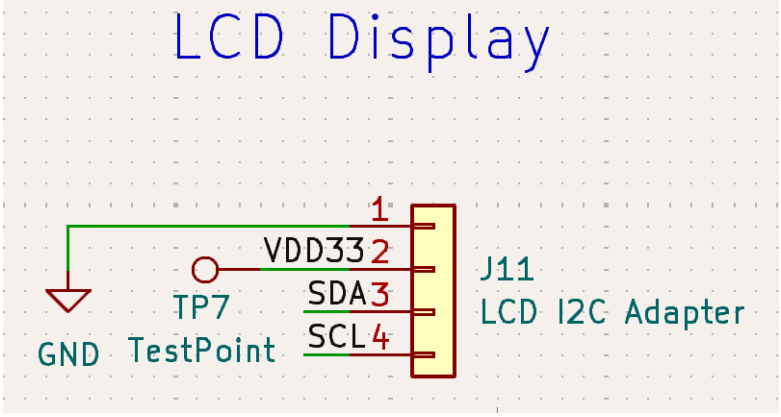

Figure 2.5: LCD Schematic

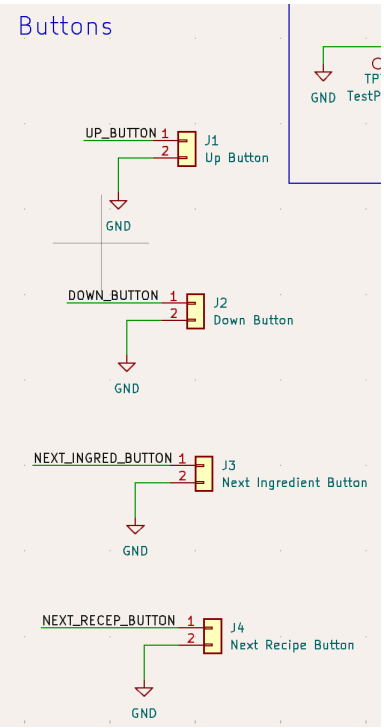

Figure 2.6: Buttons Schematic

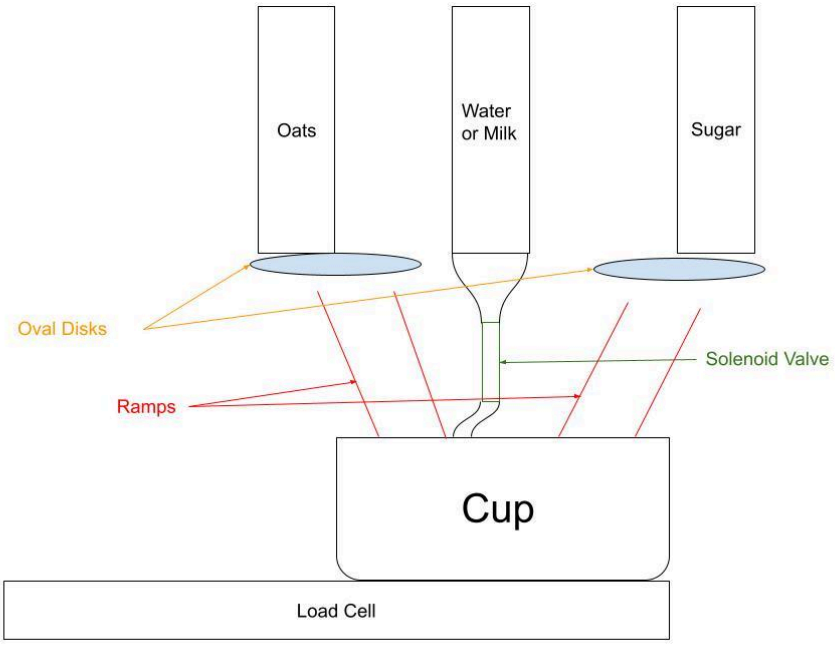

Figure 2.2: Latest Mechanical Design