ECE 445
Fall 2025
Design Document

Project #29: Modular Wafer Track for Semiconductor Fabrication

Team Members:
Jack Schnepel (jackks2)
Hayden Kunas (hkunas2)
Nathan Pitsenberger (nmp5)

TA: Shenyang Liu
Professor: Rakesh Kumar

1. Introduction

1.1. Problem Statement

In today's world, where semiconductors drive nearly every aspect of technological innovation, little room is left for small-scale fabrication and experimentation. Commercial wafer processing equipment ranges from tens of thousands to hundreds of millions of dollars, putting it far out of reach for hobbyists, educational laboratories, and early-stage researchers. Existing systems are not only cost-prohibitive but also lack the flexibility and modularity needed for experimentation on a smaller scale. As a result, innovation outside of large industrial fabs is limited, leaving students, independent researchers, and small labs without access to tools that enable exploration of semiconductor device fabrication.

1.2. Solution

Our team's solution to this problem is to design, build, and demonstrate a modular, cost-effective wafer track system that lowers the barrier to entry for small-scale semiconductor processing. The idea is to create a track that will:

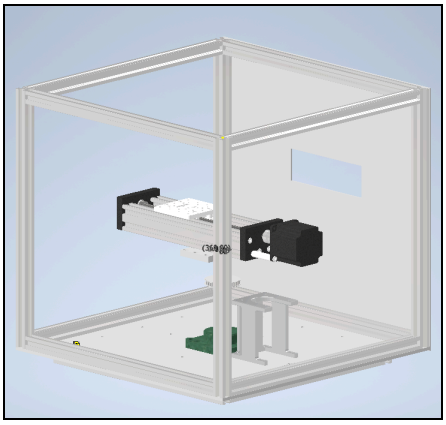
Transport wafers between the interchangeable processing modules, execute repeatable fabrication recipes that ensure process consistency, and communicate standardized instructions to each module through a defined packet interface, enabling true modularity and user-created modules.

The system architecture will be layered, with a Raspberry Pi serving as the front-end controller, providing recipe management, a user interface, and real-time monitoring. An ESP32 Microcontroller will delegate low-level instructions to each module and control the stepper motors for wafer transport. Individual modules (demonstrated through a wafer alignment station that reorients a wafer's major flat at the start of each recipe) will showcase the modular framework and mechanical precision of the track.

By defining a standardized track-module interface and releasing the system as open source, our design will empower hobbyists, students, and small research labs to reproduce, extend, and customize the platform. This solution not only addresses cost barriers but also promotes accessibility, flexibility, and innovation in semiconductor fabrication education and prototyping.

1.3. Visual Aid

[1]Tokyo Electron Limited (TEL)
Track for Wafer Processing

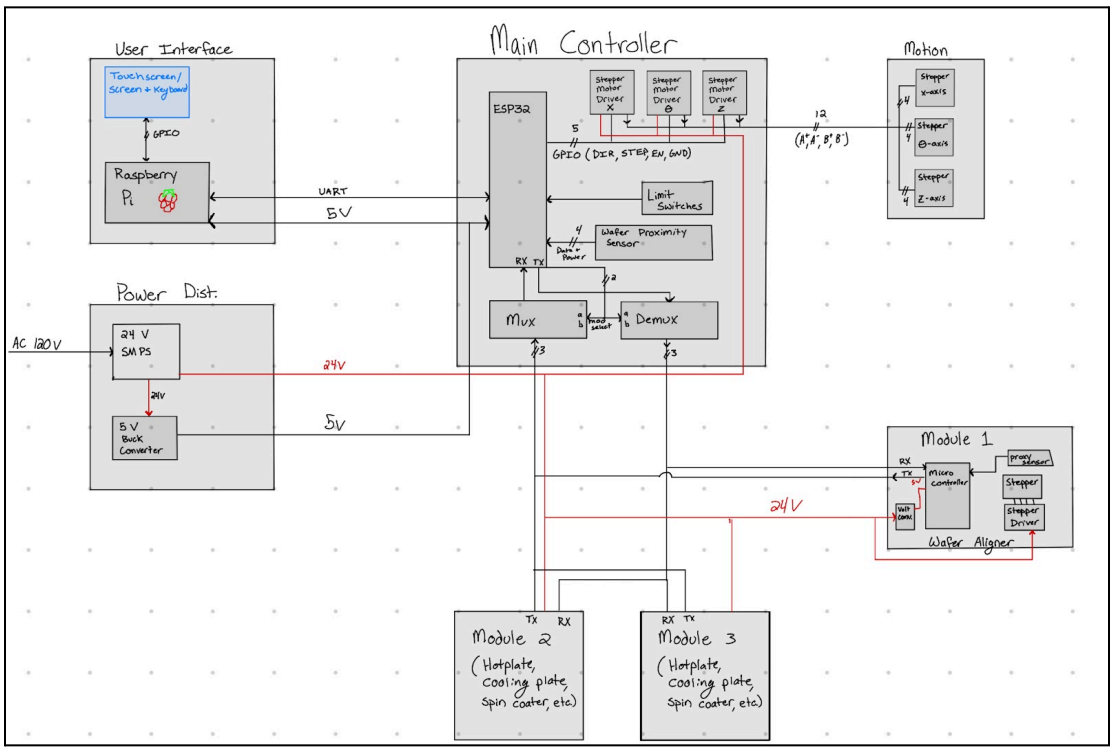


1.4. High-Level Requirements

A high-level requirement is allowing the user to create and run a program on the system, which allows the wafer to be inserted by the user in any orientation, pulled into the main chamber, transferred to the aligner tool, which aligns the wafer by the major flat, moved back into the main chamber, and finally pushed out of the chamber. This program should be able to run more than once without any noticeable misalignment of the major flat.

2. Design

2.1. Block Diagram



2.2. Physical Design

The physical design will have a frame of 2020 aluminum with acrylic paneling to cover the sides. Inside of the housing will be the motors and the PCB required for all of the communication between the submodules. The axes assembly will consist of a center shaft for rotational control, with the linear axes sitting on top of this shaft. The y-axis will be responsible for translating the wafer into each submodule and the z-axis is for lifting the wafer arm for mounting/dismounting. Additionally, the user interface system will be mounted to the frame for recipe creation and system monitoring.

2.3. Subsystem Overviews

Power Distribution Subsystem overview:

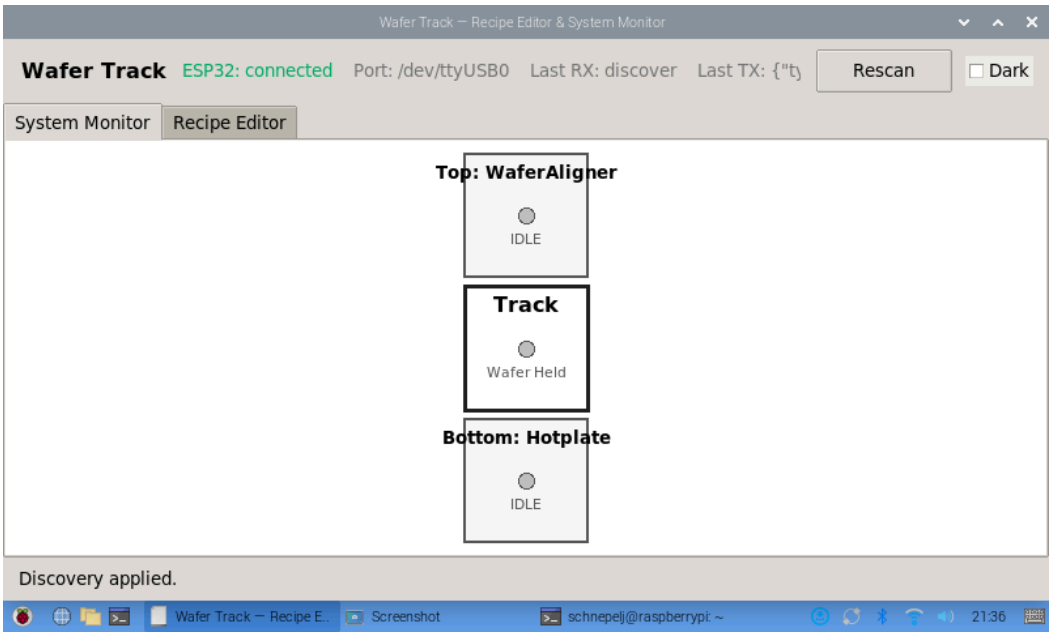
The power subsystem will be able to supply power to all of the components by stepping it down from a standard wall outlet. These components include logic ICs, a Raspberry Pi 4 model B, and linear actuators.

Main Controller Subsystem overview:

This subsystem will be responsible for receiving the wafer from the user and taking it into the system. It will feature a linear actuator on top of a stepper motor, capable of rotating 360 degrees. This would allow the wafer to move similarly to polar coordinates, except there is no phi coordinate. The wafer holder will be in the shape of a horseshoe, which can securely move the wafer around on the plane.

User Interface Subsystem overview:

The user interface subsystem will allow the user to create recipes to be executed by the modules. It will interface with the entire system to send the wafer through the steps that have been specified. Ideally, this subsystem will allow the user to save designs to be executed wherever desired by the push of a button. The user interface will consist mainly of a Raspberry Pi 4 model B, a screen, and peripherals for navigation.



Module 1 (Wafer Aligner) Subsystem overview:

This is a demonstration of a possible subsystem that can attach to the main housing. This subsystem will be able to pick up the wafer using rubber feet that approach from the bottom of the wafer and lift it while the main housing actuator retracts. It will then lower, which will bring the wafer down to the main platform. That platform can rotate such that optical sensors can detect the major flat and align it parallel to the place where the wafer entered the subsystem.

2.4. Subsystem Requirements

Power Distribution Subsystem requirements:

The power subsystem must be able to turn 120VAC from the wall to usable voltages for each component. These voltages are 24VDC rated for 5A for the linear actuators, 5VDC rated for 1A for all logic components, and a separate 5VDC system rated for 3A to power the Raspberry Pi 4 model B. These will all be buck converters. All components will work continuously within 5% of the mentioned currents and voltages for it to be considered successful.

Main Controller Subsystem requirements:

The Main Controller will be responsible for multiple high-level items. Firstly, it must control and delegate movement instructions to the stepper motors to move the wafer arm to specific modules as described by a recipe. Secondly, the Main Controller must also create files (.txt, .json, etc.) to send to the Raspberry Pi upon startup to notify it of the available modules and capabilities and store user-created recipes. The Main Controller will also be responsible for maintaining communication between each of the processing sub-modules. Finally, the Main

Controller must receive signals from the proximity sensor(s) and limit switches for feedback and control. The main controller must use less than 16MB of memory for its boot code, program code, and recipe management filesystem.

User Interface Subsystem requirements:

The User Interface System will be strictly responsible for user input and system monitoring. The screen should display the system status and allow the user to create new recipes depending on the current module(s) attached to the system. The User Interface System should be able to boot on startup and be ready to execute within 20 seconds of powering up the system. Additionally, the Raspberry Pi should be able to send data to the ESP32 in less than 100ms.

Module 1 (Wafer Aligner) Subsystem requirements:

The wafer aligner will be responsible for picking up the wafer from the main track without dropping it and lowering it down onto the main chuck in the module. It will then be able to rotate the wafer and align the major flat parallel to the door that opens to the subsystem. It should be able to correctly align the wafer within 15 seconds and have an error of less than 3.6°.

2.5. Subsystem Verification

Power Distribution Subsystem requirements:

The power distribution subsystem will be verified by using a multimeter. We will measure to make sure that the system is outputting a consistent 5V and 3.3V to the correct systems with a maximum tolerance of 5%. Our procedure will first connect all the proper wiring and then use our multimeter using set inputs and outputs of power to make sure the voltages are stable. We will keep a note of any issues of verification in our notebooks and our final presentation.

Main Controller Subsystem requirements:

The main controller verification will be handled by the Raspberry Pi. As stated before, the Raspberry Pi will communicate with the main controller and will display whether or not it is connected properly and whether the subsystems are being read. The equipment needed is the Raspberry Pi itself and nothing more (which is to display the errors or to determine if communication is being processed correctly). The test procedure will include running simulated subsystem calls through the main controller to the Raspberry Pi and displaying the intended outputs. These results will be documented in our notebooks, and any major findings will be in our final presentation.

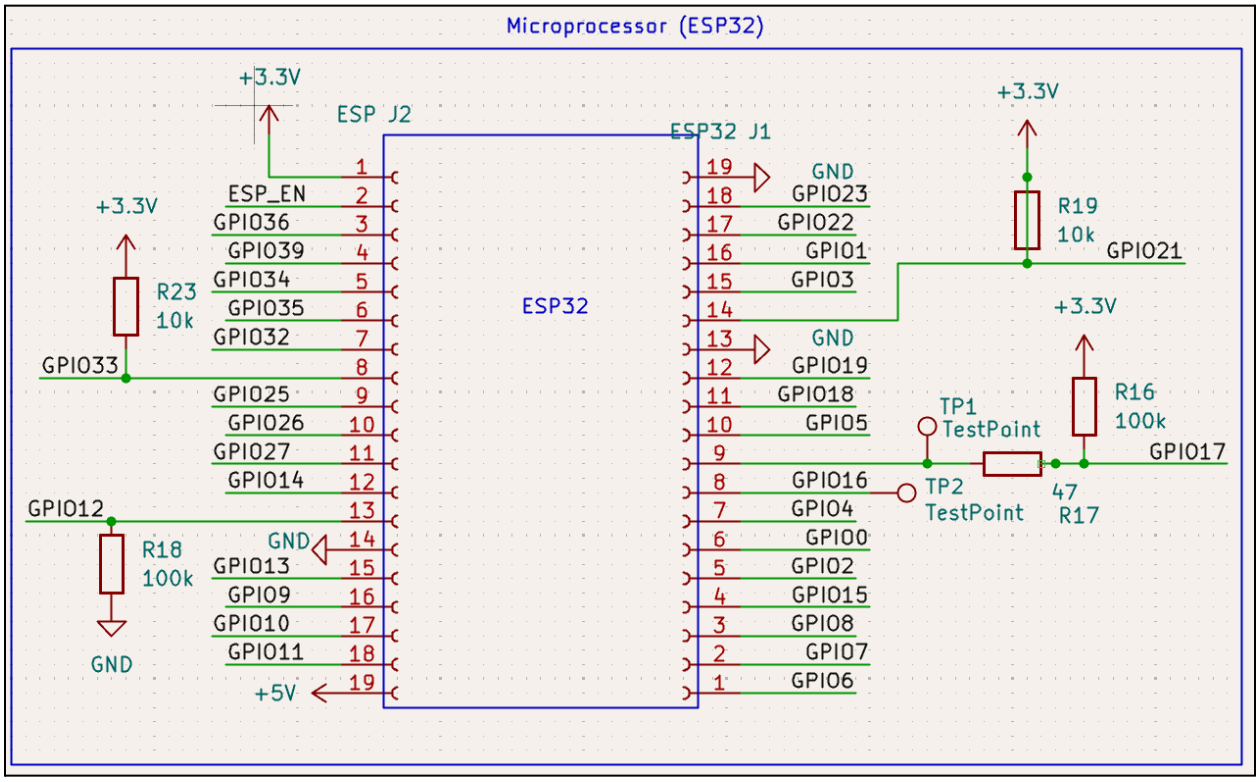
User Interface Subsystem requirements:

For the user interface, there is not much verification needed to be done. In order to verify that the system is working properly, the user needs to watch the startup interface and make sure there are no warnings or errors that occur. (These warnings include whether or not any subsystems are detected, or if there is an unknown subsystem that was connected.) There is no equipment needed other than the interface itself to display errors. The only procedure is to turn on the system, and the results should show no error message on the screen. If there is any error, we will record these in our notebook and final presentation.

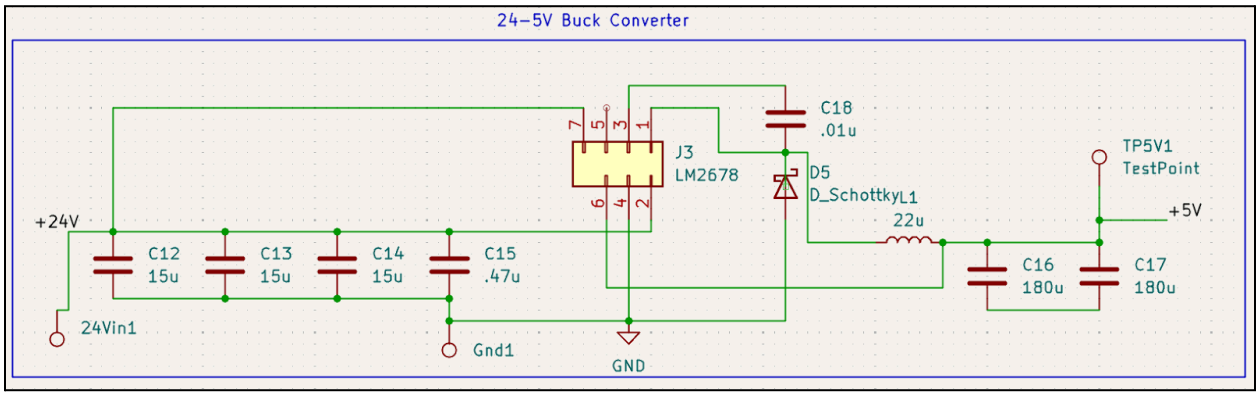
Module 1 (Wafer Aligner) Subsystem requirements:

The wafer aligner will be equipped with a visual verification system. The equipment we will use for this verification will already be included on our wafer sample holder, which is on our linear acquiator. The holder is inscribed with tick marks incrementing by 1° will allow the user to verify how far the system rotated the sample and if it did so correctly. For our team, our wafer will be inscribed with a mark on top of the wafer, which would allow us to visually see how much the wafer is being rotated and if it is doing so correctly. The results will be stored in our notebooks, along with being shown in our final report.

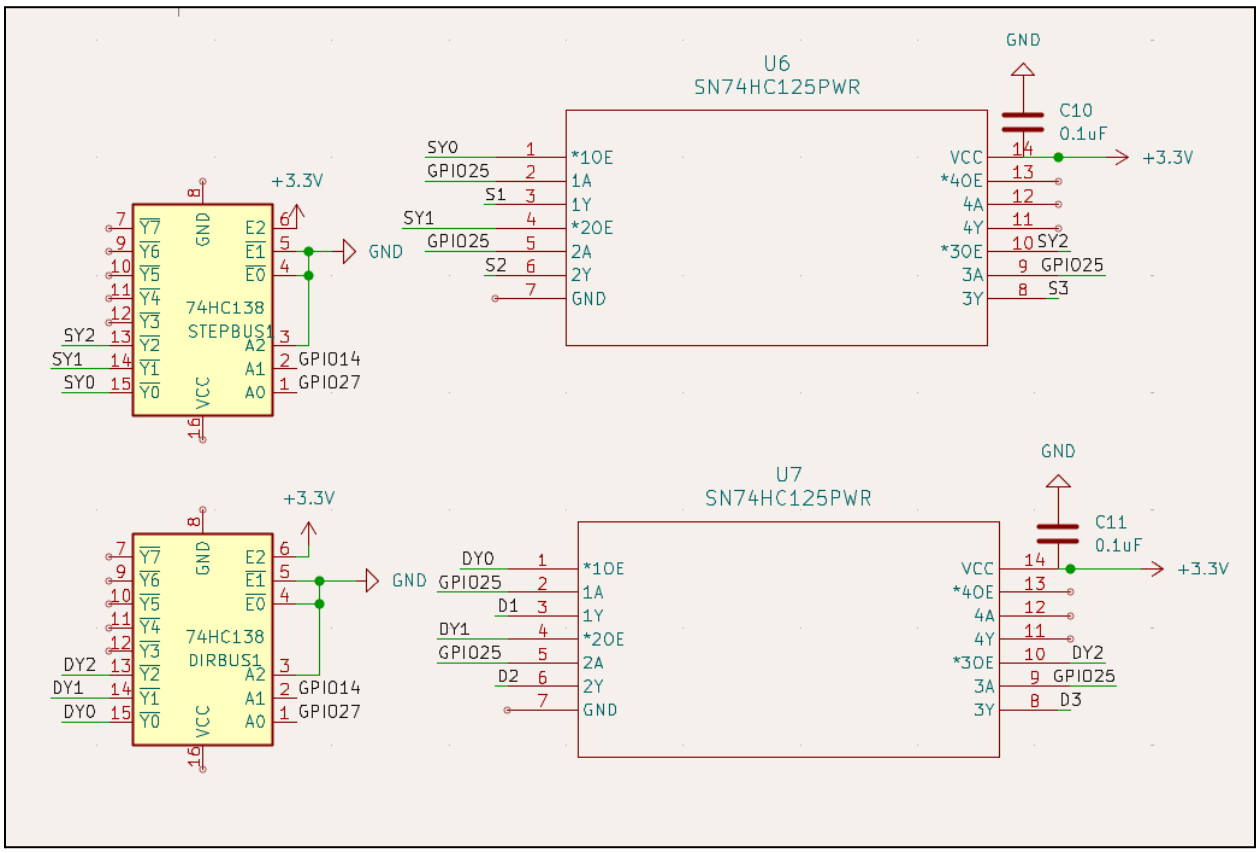
2.6 System Schematics



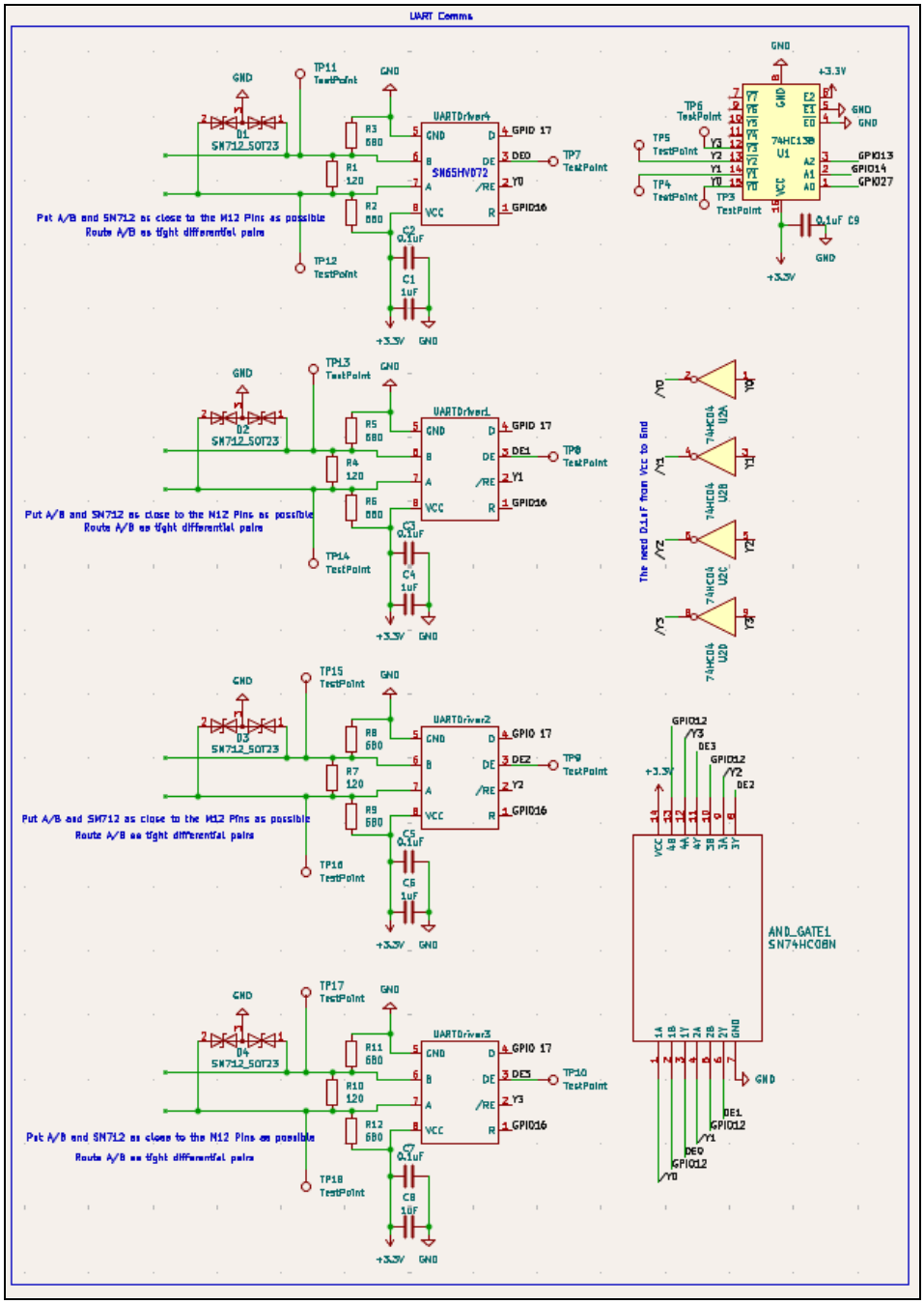
This microprocessor schematic contains the necessary GPI assignments from the dev board to the rest of the components in the schematic. It also has the necessary pull-up resistors that are needed for proper operation. This development board will be turned into just a single ESP32 processor when we verify that the circuit works as it is intended to.



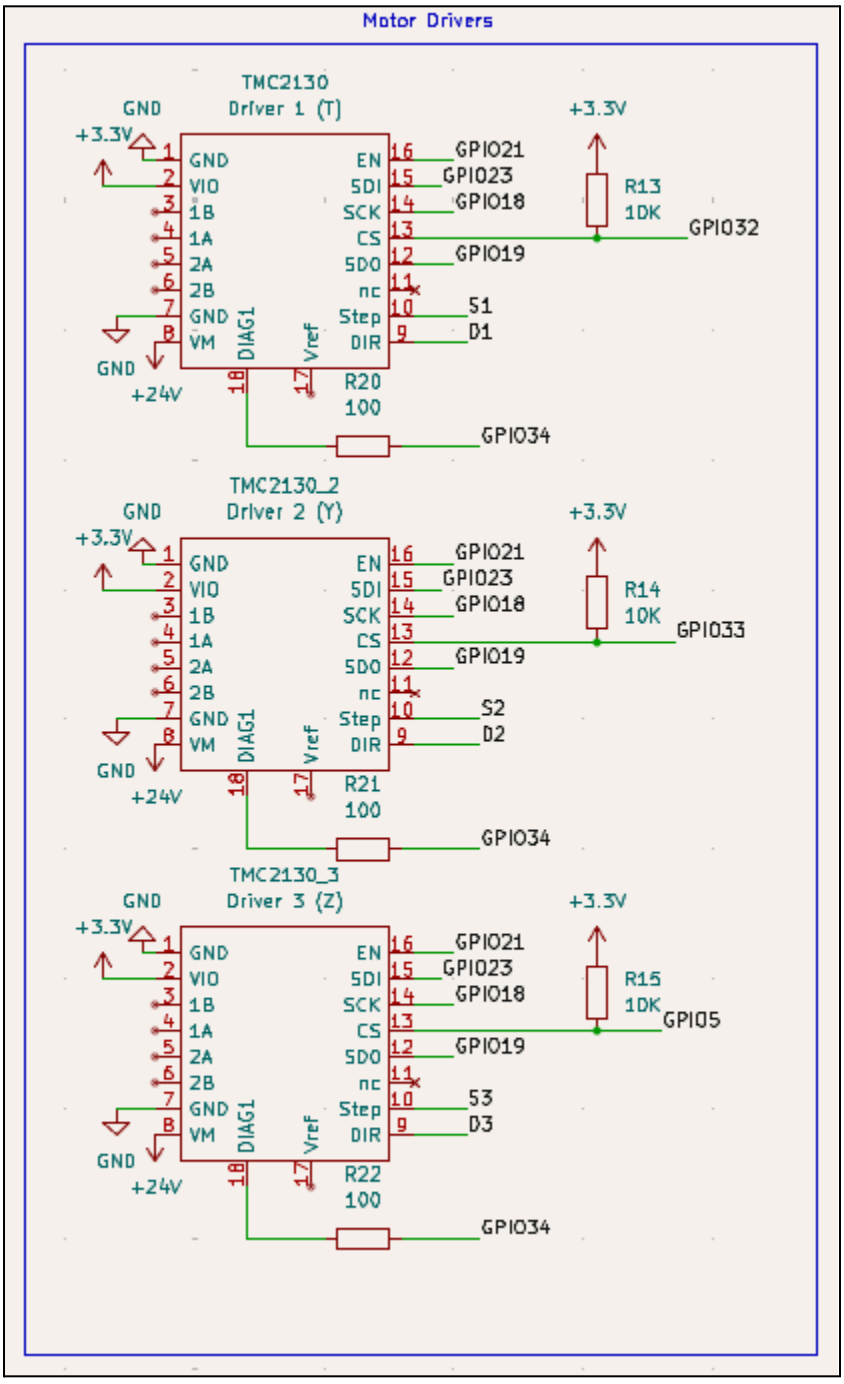
This is the schematic for the 24V-5V buck converter. The circuit is based around the LM2678 chip, which is the main converter. This design allows for high amperage to be delivered to the components that need it most, such as the Raspberry Pi 4 Model B, which needs 2A, and the stepper motors, which also need 2A. The design allows for a maximum of 5A to be delivered, which is well above that which is required of our components.



This schematic is designed to fan out the STEP and DIR pins (via the tri-state buffer) for the stepper motor drivers to limit GPIO use from the ESP32. The 2:4 decoders pull only one channel's /OE LOW at a time, which enables the specific driver's STEP/DIR.



For the UART schematic, there are a few key items here in play. This circuit lets the ESP32 share one TX (D**)** and one RX ® with up to four RS-485 ports, while guaranteeing that only the selected port is allowed to transmit (DE) and only that port's receiver is enabled (/RE) during a transaction. This also provides no bus contention between multiple UART peripherals, making it just a simple firmware select. The RS-485 differential provides noise-robustness.



This schematic is for the TMC2130 stepper motor drivers. There are three; one for extension, one for rotation and one for the z-axis. These are powered by the main power supply and will take inputs from the ESP32 as control logic.

2.7 UART Transceiver and Communication Protocol

The UART transceivers on the main board implement RS-485, half-duplex home-run links - one cable per submodule - operating at 115200 baud. The ESP32 is the bus master; submodules are slaves and only transmit when explicitly polled. For each transaction, the ESP32 selects the link, asserts DE on the SN65HVD72 only while bytes are transmitting, then de-asserts DE and keeps /RE enabled to receive the reply.

Messages are framed with COBS (Consistent Overhead Byte Stuffing) and delimited by a 0x00 byte. COBS ensures the delimiter never appears inside a frame, giving unambiguous packet boundaries and quick resynchronization after errors.

COBS Packet Layout			
Offset (Bytes)	Size (Bytes)	Field	Notes
0	1	SOF = 0xA5	
1	1	Version = 0x01	
2	1	FLAGS	
3	1	MSG_TYPE	0 = Master (ESP32)
4	1	SRC_ID	Logical module id
5	1	DEST_ID	
6	2	SEQ	
8	2	LEN	Payload length in bytes
10	N	PAYLOAD	Command-specific fields
10+N	2	CRC16	
Core message type examples: ● 0x01 DISCOVER/0x81 DISCOVER_R - identity & capabilities ● 0x02 GET_STATUS/0x82 STATUS_R - (IDLE/BUSY/ERROR) ● 0x03 SET_PARAM/0x83 SET_PARAM_R - set operating parameters ● …			

Only one outstanding request per link. On timeout, the master retries up to two times to recover the message; persistence failure marks the module offline. During long-running steps, the master polls with GET_STATUS.

2.8 Tolerance Analysis

A potential risk in determining project quality is the stepper alignment. The stepper motors must be positioned correctly to ensure transferring does not collide with the frame or other structures. To avoid this issue, we will implement two different measures. Firstly, we may include microstepping on the stepper motors for very precise movements (could 1/16 microstep, making one revolution 3200 steps, opposed to 200 steps). Additionally, we will implement limit switches on the stepper motors to home the axes at startup.

Another potential risk can come in the form of a power surge. This could result in broken electrical components and motors. This can be prevented by a simple surge protection IC chip. An example of this would be the TVS2200, which can handle current surges up to 40A.

2.9 Cost Analysis

Description	Manufacturer	Quantity	Extended Price	Link
Raspberry Pi 4	RASPBERRY PI	1	$64.78	Link
Raspberry Pi 4 Touch Screen	Hosyond	1	$42.50	Link
250mm High-Torque C-Beam Linear Actuator	Bulkman3D	1	$122.50	Link
Custom-Cut Aluminum Baseplate	Xometry	1	$73.83	N/A
2020 Aluminum Extrusions	8020			
BigTreeTech 2130v3.0 Stepper Motor Driver	BIGTREETECH	3	$14.99	Link
M5 T-Nut 200pc Set	JCSPBYL	1	$10.89	
Hardened Chrome 150mm Shaft	Vigorous	1	$7.51	Link
Aluminum Timing Pulley 1:4	Zeelo	1	$8.50	Link
UCF201 Pillow Block Flange Bearing	XIKE	1	$10.00	Link
12mm Flange Coupling Connector	Diaer	1	$2.25	Link
APDS-9930 ALS IR RGB and Proximity Sensor	PAMEENCOS	2	$4.00	Link
UART Driver SN65HVD72	Texas Instruments	4	$13.52	Link
LM2678SX-5.0/NOPB	Texas Instruments	1	$8.24	Link
TLV77533PDBVR	Texas Instruments	1	$0.14	Link
PA4342.223ANLT	Pulse Electronics	1	$1.06	Link
SK64L-TP	Micro Commercial Components	1	$0.95	Link
SM712	Texas Instruments	4	$1.12	Link
SN74HC04DR	Texas Instruments	1	$0.29	Link
SN74HC08DR	Texas Instruments	1	$0.30	Link
SN74AHC139DR	Texas Instruments	2	$0.64	Link
SN74HC125PWR	Texas Instruments	2	$0.74	Link
120 Ω Resistor	Vishay Dale	4	$0.40	Link
24V 10A Power Supply	HUEMIHUI	1	$25.99	Link
47 Ω Resistor	Vishay Dale	1	$0.10	Link

680 Ω Resistor	Vishay Dale	8	$0.80	Link
100 Ω Resistor	Vishay Dale	3	$0.30	Link
10k Ω Resistor	Vishay Dale	4	$0.40	Link
100k Ω Resistor	Vishay Dale	2	$0.20	Link
.01uF Capacitor	Vishay Vitramon	1	$0.16	Link
.1uF Capacitor	Vishay Vitramon	7	$1.26	Link
.47uF Capacitor	KEMET	1	$0.18	Link
1uF Capacitor	Vishay Vitramon	4	$0.80	Link
15uF Capacitor	KEMET	3	$0.62	Link
180uF Capacitor	Panasonic Electronic Components	2	$0.82	Link

Cost of materials: $298.14
Cost of Labor: 3 [people] * 10 [weeks] * 8 [hours/week] * 42.5 [dollars/hour] = $10,200
Total cost: $10,498.14

2.10 Schedule

Week Of	Tasks	Member
10/13	Finish the design document and the teamwork evaluation	All
	Assemble the base plate and finish the design of the Z-axis assembly	Jack
	Work on smaller ESP32 integration for the second round of PCBWay ordering	Nathan
	Create a screen saver for Raspberry Pi, along with a recipe-saving system	Hayden
10/20	Finalize the wafer alignment subsystem process and design	Jack
	Finalize wafer alignment PCB	Nathan
	Work on Arduino and Raspberry PI simulation	Hayden

10/27	Prepare for breadboard demo 2	All
	Assemble all components for the system	Jack
	Solder the main component and subsystem PCBs	Nathan
	Ordering wiring for connecting the subsystem and the main component integration	Hayden
11/3	Work on individual progress reports	All
	Calibrate the alignment subsystem	Jack
	Prepare for the third round of PCBWay ordering	Nathan
	Connect subsystems to the main components and the Raspberry PI interface	Hayden
11/10	Fix any wiring/power mistakes for the mock demo	Jack
	Prepare for the fourth round of PCBWay ordering	Nathan
	Finalize any bug errors within the code for the mock demo	Hayden
11/17	Practice for mock demo	All
	Finalize any last design changes before the final demo	Jack
	Work on any general errors found during the mock demo	Nathan
	Fix any code errors/integration errors found in the mock demo	Hayden
11/24	Work on the final presentation report, final papers, and practice for the final demo by fixing any last-minute errors.	All
12/1	Prepare for the final demo and practice mock presentation. The project should be finished, and this week should just be for preparation.	All
12/8	Prepare for the final presentation and finish the final paper	All

3. Ethics and Safety

Pledge:

Our group promises to follow all the ethics and safety protocols mentioned in this document. We will adhere strictly to the IEEE code of ethics by pledging to follow the following statements: Public safety and welfare will be at the forefront of our development, we will prioritize the safety, health, and welfare of the public, and we will make our product/design suitable and safe for the public. Our group will hold integrity and honesty throughout our entire process, making sure we stay clear of any conflict of

interest or unlawful conduct. We will create a safe and welcoming environment for all members of the team, including our mentors and advisors, and not discriminate or engage in any form of harassment. Along with this, we will uphold the professional development of all members of the team and hold each other to a high standard to reach our success. Lastly, our group will strive for continuous improvement of technical abilities, professional skills, products that will help humanity, and leadership skills.

Along with our pledge, our group will strictly adhere to any local laws and regulations, as well as any federal laws. If our group needs any certifications to operate machinery or any permits to continue our project, we will get them. As a part of our pledge to adhere to the IEEE code of ethics, we will make sure that anyone in our vicinity is safe and never put anyone in harm's way. These pledges and guidelines will make our group the best if we can be and have the best final project we can. Since working on our project, no existential safety concerns have been brought to light, nor were any brought up during our proposal meeting.

Project Safety Procedures:

When designing our project, we will keep user safety at the core of our efforts. One such way this will be enforced is by ensuring there are no moving parts or pinch points with human contact in our design. The UI interface, when powering on, will warn the user of any potential dangers of using the system, especially when unloading and loading the samples. Any moving parts in our system will never move on their own until specified by the user. This ensures user safety and mitigates any potential risks of harm to users when using the system. Our project does not involve any high-risk factors when operating or assembling, which includes not using high voltage, high speeds, or harsh chemicals. The user, if creating their own subsystem, will be responsible for what the subsystem does and any risks associated with its creation. Nothing in our base project will be able to produce harm to anyone who uses it or does the required maintenance.

4. References

[1] Tokyo Electron Ltd., “Coater/Developer LITHIUS™ Series,” [Online]. Available: https://www.tel.com/product/lithius.html. Accessed: Sep. 18, 2025.

A. Levido, “Consistent Overhead Byte Stuffing,” Circuit Cellar, Feb. 9, 2023. [Online]. Available: https://circuitcellar.com/resources/quickbits/consistent-overhead-byte-stuffing/ Accessed: Oct. 12, 2025.