Team 16 CDR
Trash·*E*

University of Massachusetts Amherst
BE REVOLUTIONARY™

Meet The Team

Stephen Townsend
Computer Engineer

Jasmine Hickey
Electrical Engineer

Smit Patel
Computer Engineer

John Diep
Computer Engineer

Team 16 Advisor
Professor Do-Hoon Kwon

University of Massachusetts Amherst

PROBLEM STATEMENT

Bringing garbage to the curb can be a difficult task for some, whether they are physically disabled or elderly. Aid lent by caring companions can be unreliable as everyone has their own schedule and may not be available at the necessary time. Those in need of refuse removal can be assisted using *Trash·E*. *Trash·E* will take the strenuous part of taking the barrel to the curb out of the process; it is a Wi-Fi enabled robotic system to move garbage from its near-home collection site to the curb on a path and at a specified time.

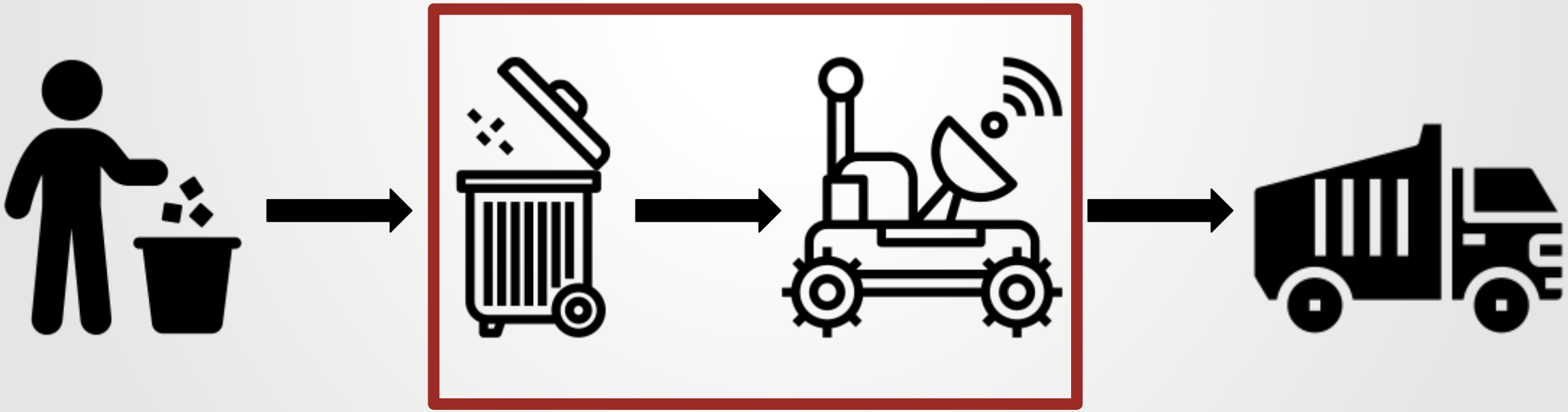
BE REVOLUTIONARY™

DESIGN GOALS, SPECIFICATIONS, AND TESTING PLAN

University of Massachusetts Amherst

GOAL

The goal of the *Trash·E* is to provide an automated and convenient way to move trash from its place of collection to the location needed for removal by a trash service with minimal user intervention.



Achieved/Not Yet Achieved Deliverables

Large Motor Enclosure	✅
Antenna Design + Research	❌
PCB Design	✅
Migration to Embedded C	✅?
GPS Migration	❌
IMU Migration	❌
Scheduler	❌
UI	✅/2

Design Goals

- The System shall
  - specify what days and times *Trash·E* will run
  - wait for a preset amount of time at the pickup location before returning to its starting location
  - detect when trash is picked up
- Navigation
  - follow designated line forward and backwards
  - return to line if deviation occurs
- Obstacle Detection
  - detect and stop before obstacles in front of the system
  - alert the user of an obstacle
- Environmental Restrictions
  - fully functional on paved surfaces

System Specs (Updated)

Requirement	Specification	Value
Arrive before specified pickup time	User-inputted day/time	Arrive 10 minutes before time specified, alert user
Payload of average trash bag	Pounds	Greater than or equal to 25 lbs.*
Battery Life	Trips / Time	Two round trips on one full charge
Wait at Pickup Location	Time / Load Verification	Wait until load is taken away or EOD
Object Detection	Distance	Detect objects <= 1 meter from rover (90% accuracy)
Navigation [Primary]	Line Tracking	Follow line from start to pickup and back (90% accuracy)
Navigation [Secondary]	Find Relative Home	Alert user after disturbance or error within five minutes (80% accuracy)
Movement	Surface, Slope	Work on paved surfaces up to 10° slope
WiFi Connectivity	Distance	20 meters, LOS
Arrive at Starting & Pickup Location	Error Tolerance	Arrive within 2 meters of final destination (90% accuracy)
User Alert [Obstacle]	Time	Alert the user of an obstacle within one minute (90% accuracy)

Testing Plan

Navigation [Primary]

- Design test track with varying designs (straight, curvy, angled, etc.)
- Designate start and pickup locations
- Test 10 times per track
- Measure distance from pickup location, emulate trash pickup
- Measure distance from start location
- Repeat for different layouts
- Repeat process outdoors

Object Detection

- Design test track of varying designs
- Place obstacle on line, mark location
- Run Trash-E on the track, mark where system stops
- Begin timer when object detected
- End timer when alert is received
- Measure distance between obstacle and rover
- Repeat 10 times

Load Detection

- Obtain varying weights up to 25 lbs
- Place weight on sensor
- Remove
- Verify detection
- Repeat 10 times

Navigation [Secondary]

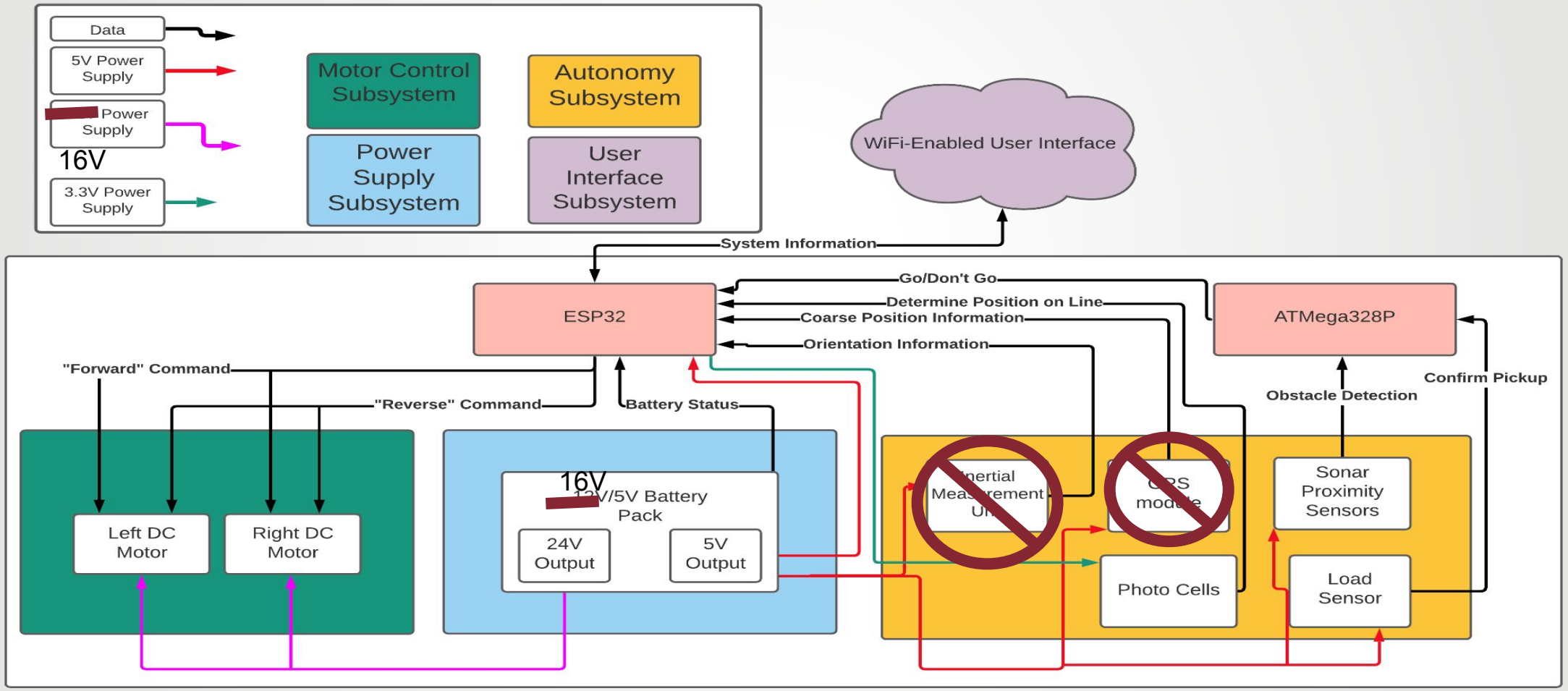
- Trash-E follows line correctly, stores data
- Emulate outside force (place rover off line)
- Start timer
- Wait for Trash-E to alert user
- Stop timer, verify under 5 minutes
- Repeat 10 times

BE REVOLUTIONARY™

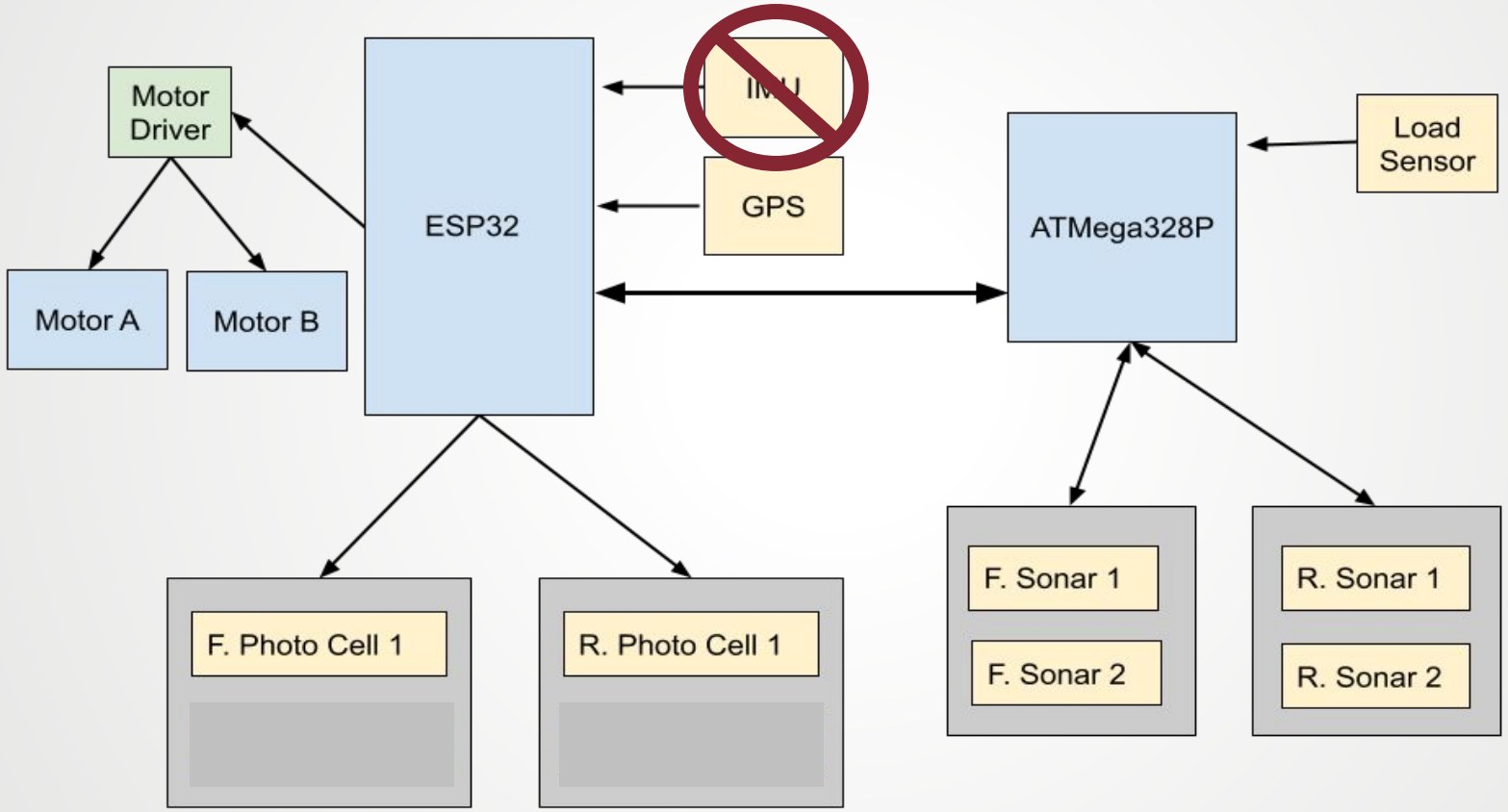
BLOCK DIAGRAMS & PCB DESIGN

University of Massachusetts Amherst

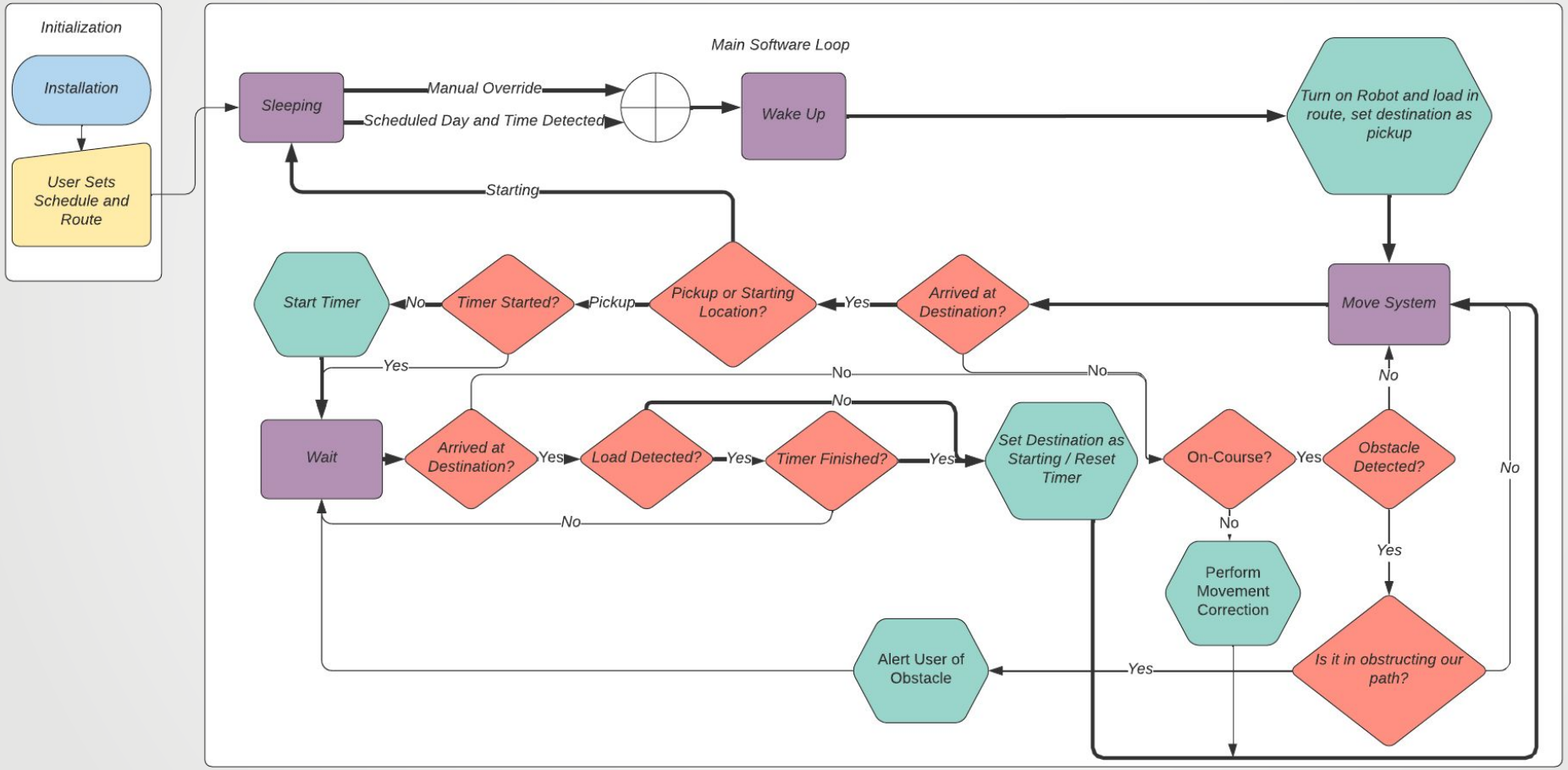
(UPDATED) SYSTEM BLOCK DIAGRAM



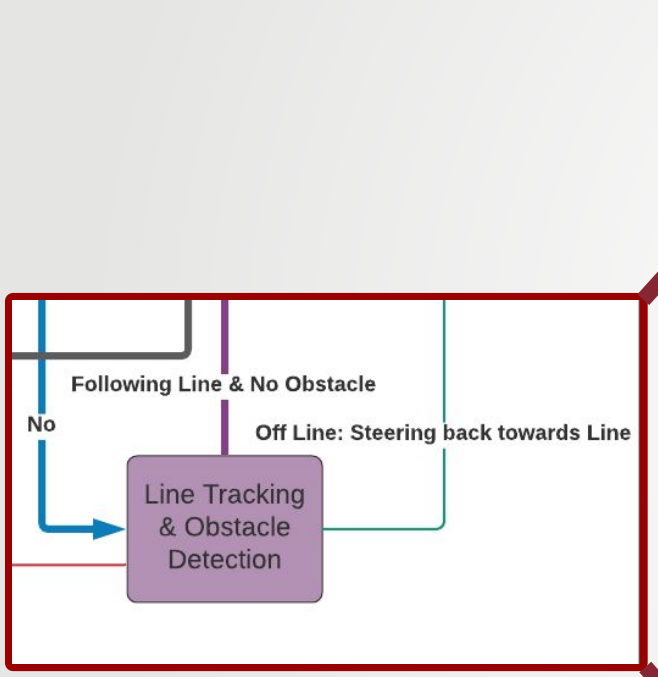
Hardware Block Diagram

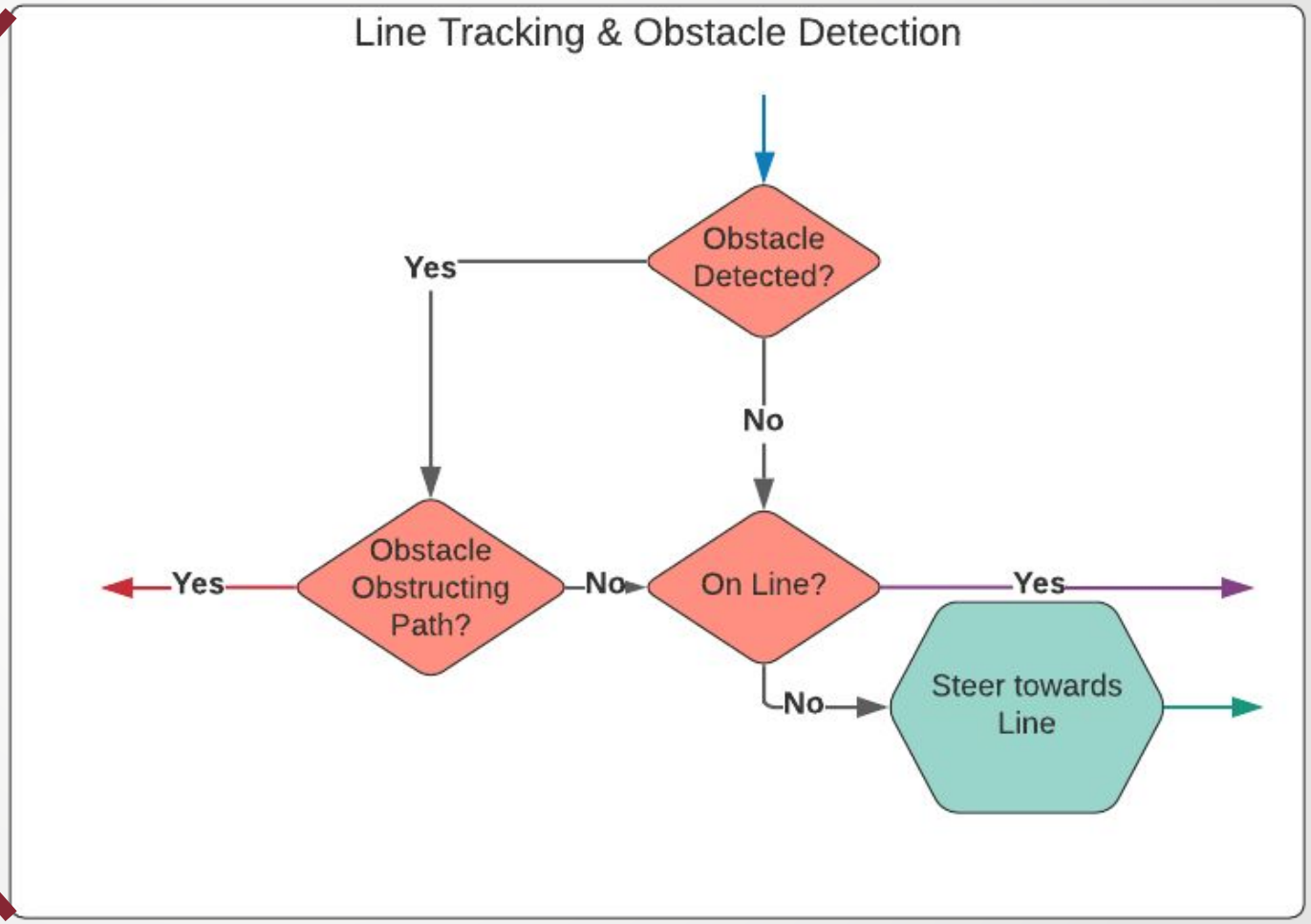


Software Block Diagram



Deeper Look: Navigation and Obstacle Detection





The Scale Up

- 19” x 19” x 6” 80/20 aluminum frame
  - lightweight and sturdy
  - conjoined with corner brackets on all joints
- 4.5” driver wheels
  - connected to motor shafts
  - motors are held in place with L brackets and pipe strap
- 3” castor (lazy) wheels
  - in all four corners for stability
- Everything is mounted to a wooden baseboard
- Polycarbonate siding for weather proofing
  - will be permanently caulking the edges once finished prototyping
- Total weight is about 25 pounds

Subsystem Analysis: Motor Control

```
void A_Stop(void)
{
    digitalWrite(MA_IN1, 0);
    digitalWrite(MA_IN2, 0);
    analogWrite(MA_PWM, 0, MAX_SPEED);
}
```

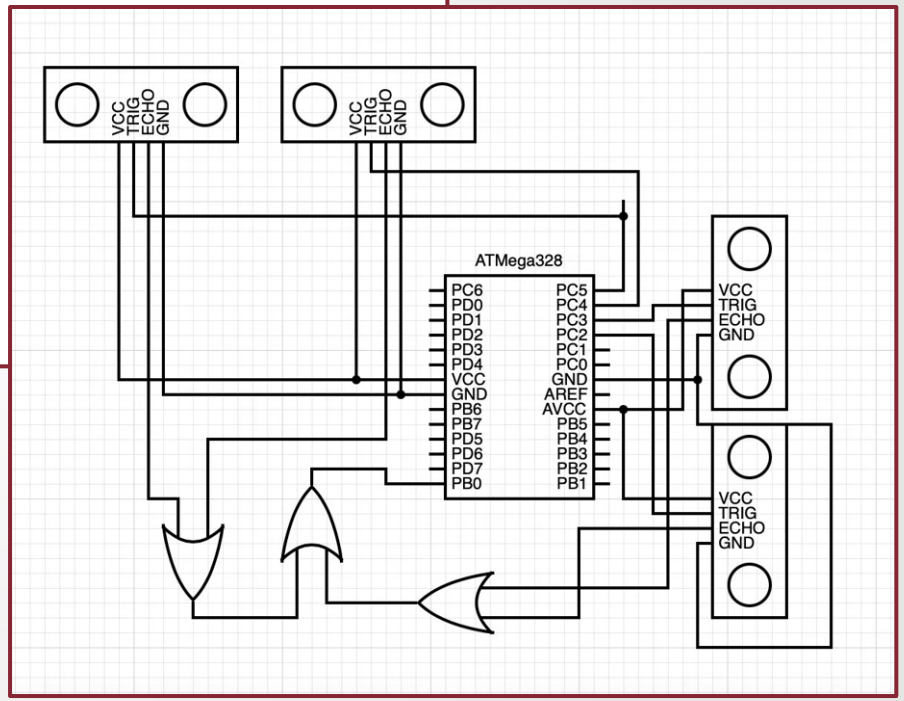
```
void A_CCW(int speed)
{
    digitalWrite(MA_IN1, 0);
    digitalWrite(MA_IN2, 1);
    if(is_stopped)
    {
        for(int i=0; i<speed; i++)
        {
            analogWrite(MA_PWM, i, MAX_SPEED);
        }
    }

    else analogWrite(MA_PWM, speed, MAX_SPEED);

    is_stopped = false;
}
```

```
void A_CW(int speed)
{
    digitalWrite(MA_IN1, 1);
    digitalWrite(MA_IN2, 0);
    if(is_stopped)
    {
        for(int i=0; i<speed; i++)
        {
            analogWrite(MA_PWM, i, MAX_SPEED);
        }
    }

    else analogWrite(MA_PWM, speed, MAX_SPEED);

    is_stopped = false;
}
```

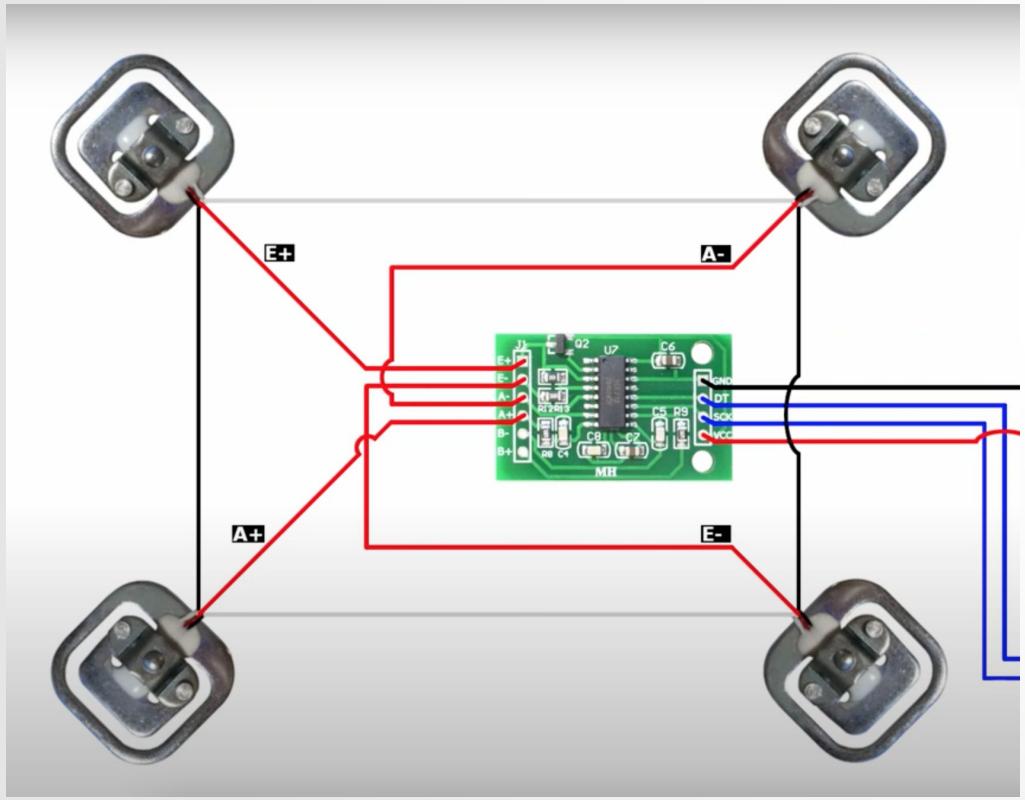
Subsystem Analysis: Sonar Sensors

```c
float measureDistance(int pinNum) {
    float distance = 0;
    unsigned int clockTotal = 0, startTime = 0, endTime = 0;

    PORTC = (0<<pinNum);
    _delay_us(2);
    PORTC = (1<<pinNum);
    _delay_us(10);
    PORTC = (0<<pinNum);

    TCNT1 = 0; //Reset Counter
    TCCR1B = 0x45;

    while(!(TIFR1 & (1<<ICF1)));

    startTime = ICR1;
    TIFR1 = (1<<ICF1);
    TCCR1B = 0x05;

    while(!(TIFR1 & (1<<ICF1)));
    endTime = ICR1;
    TIFR1 = (1<<ICF1);

    clockTotal = endTime - startTime;
    distance = clockTotal * 1.098;

    return distance;
}
```

```c
int main(void)
{
    /* Replace with your application code */
    DDRC = (1<<FRONT_LEFT_ULTRA) | (1<<FRONT_RIGHT_ULTRA) | (1<<BACK_LEFT_ULTRA) | (1<<BACK_RIGHT_ULTRA);
    DDRB = (0<<0) | (1<<FRONT_LEFT_LED) | (1<<FRONT_RIGHT_LED) | (1<<BACK_RIGHT_LED) | (1<<BACK_LEFT_LED);
    DDRD = (0<<0);
    PORTD = 0;

    float FR = 0, FL = 0, BR = 0, BL = 0;
    while (1)
    {
        if (PIND & (1<<PIND0)) {
            FR = measureDistance(FRONT_RIGHT_ULTRA);
            FL = measureDistance(FRONT_LEFT_ULTRA);
            if (FR < 5) {PORTB |= (1<<FRONT_RIGHT_LED);}
            else if (FL < 5) {PORTB |= (1<<FRONT_LEFT_LED);}
            else {PORTB = 0;}
        }
        else {
            BR = measureDistance(BACK_RIGHT_ULTRA);
            BL = measureDistance(BACK_LEFT_ULTRA);
            if (BR < 5) {PORTB |= (1<<BACK_RIGHT_LED);}
            else if (BL < 5) {PORTB |= (1<<BACK_LEFT_LED);}
            else {PORTB = 0;}
        }
```



Subsystem Analysis: Load Sensors




```
scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
scale.set_scale(calibration_factor); //This value is obtained by using the SparkFun_HX711_Calibration sketch
scale.tare(); //Assuming there is no weight on the scale at start up, reset the scale to 0
```

```
if (scale.get_units() > 150) {   //150 grams ~ .3 lbs
 digitalWrite(10, HIGH);
  }
else {
     digitalWrite(10, LOW);
    }
```

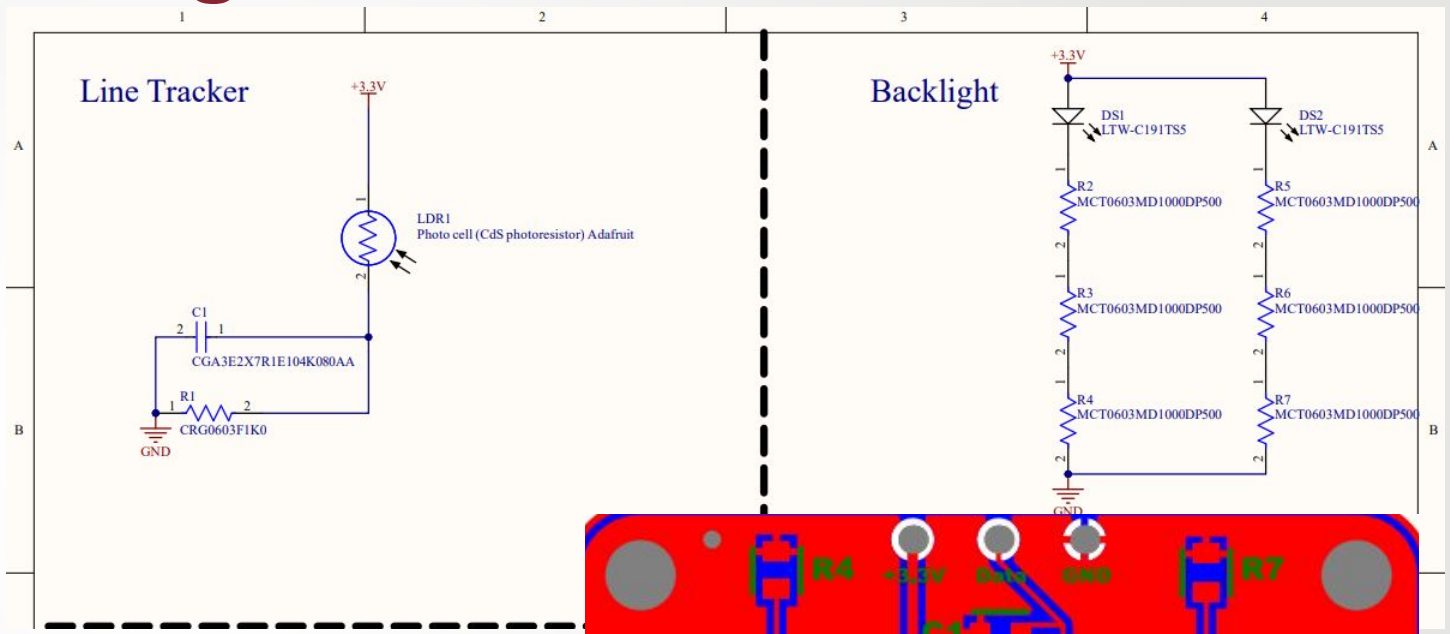
HX711 load cell amplifier:

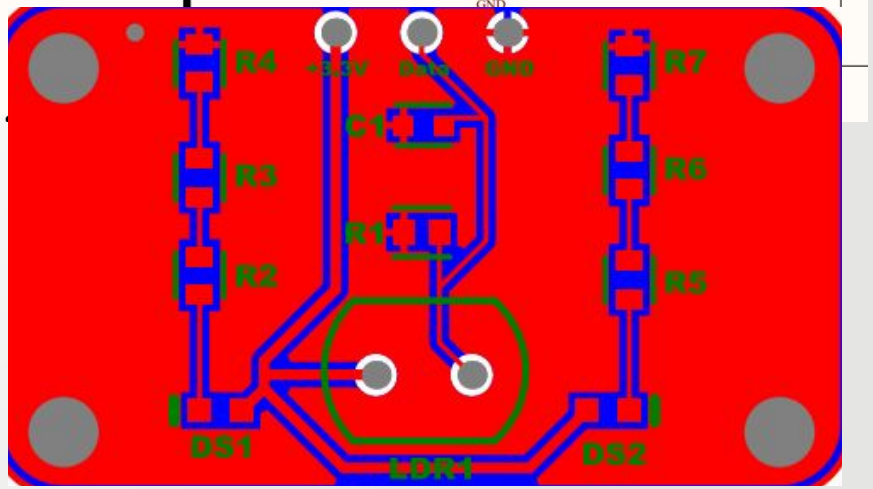
- gets measurable data from load cells

Subsystem: Line Tracking via Photoresistor

Step by Step Process:

1. Calibrate Photoresistors
   a. Find brightest color -> highest voltage value
   b. Find darkest color -> lowest voltage value
2. Perform Analog Read via ADC on voltage divider
3. Squeeze newly read value into our calibrated range
   a. Convert to a boolean value
4. Translate Boolean Values into human understandable cases
5. Repeat steps 2 - 4





$$V_{out} = V_{in} \cdot \frac{R_2}{R_1 + R_2}$$

$V_{upper\ out}$ = 3.3 * 1000 / (1000 + 1000) = 1.65V

$V_{lower\ out}$ = 3.3 * 1000 / (10000 + 1000) = 0.3V

Subsystem: Line Return Algorithm Overview

```
switch(trackLine(ADC1_CHANNEL_3, ADC1_CHANNEL_6, false, &photoValStruct, true))
{
    case 1: //left side triggered
        stop();
        delay(100);
        while(trackLine(ADC1_CHANNEL_3, ADC1_CHANNEL_6, false, &photoValStruct, true) == 1)
        {
            reverse(motorSpeed);
            wasForward = false;
            wasRev = true;
        }
        delay(800);
        stop();
        delay(1000);
        rotate_right(motorSpeed);
        delay(500);
        stop();
        findAttempt = false;
        break;
```

Subsystem: Messaging System on ESP32

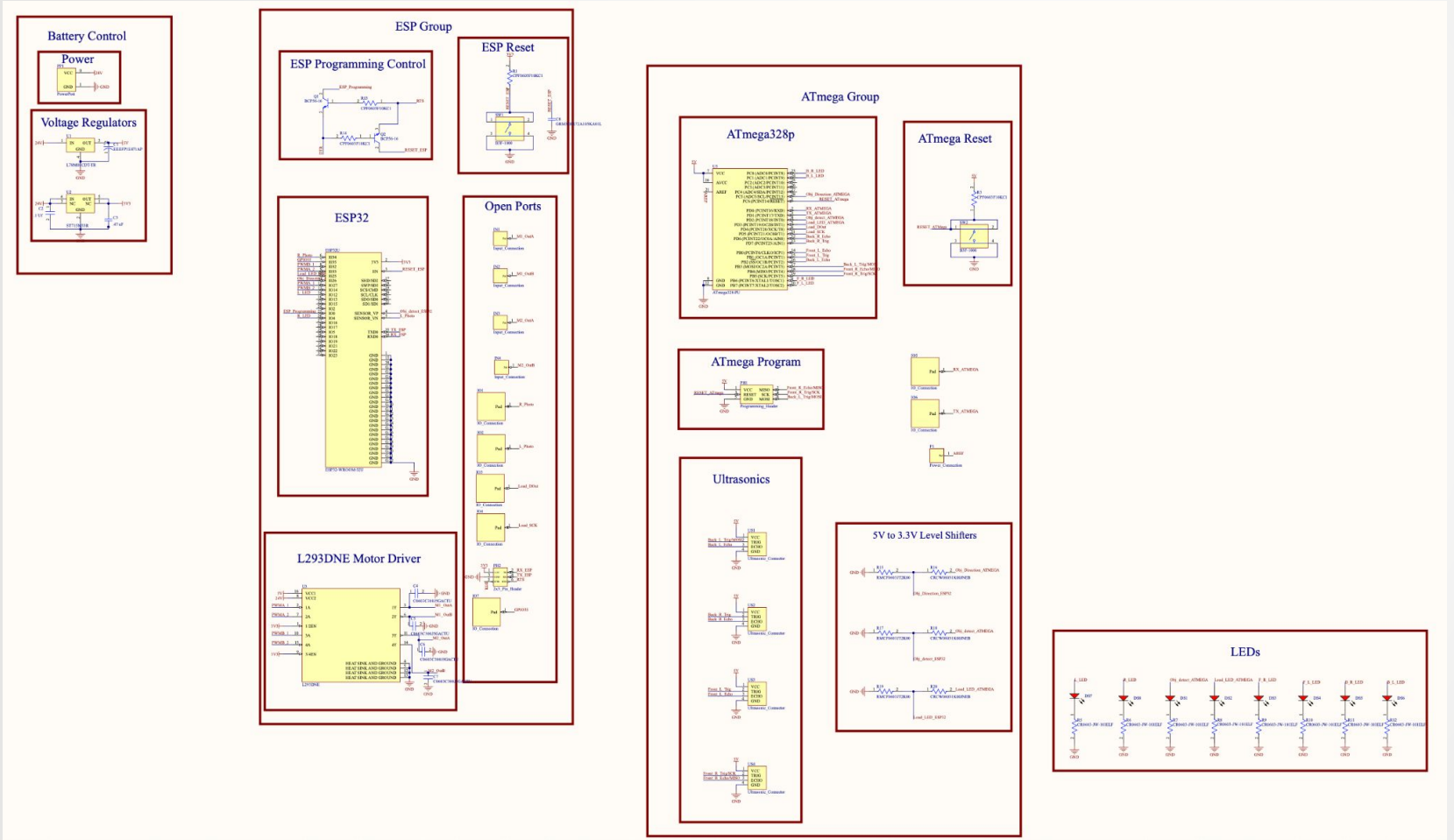
General Process:

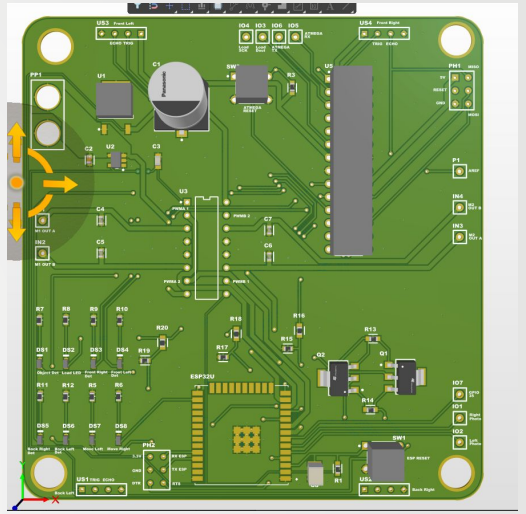
1. Connect to Internet connected Network
2. Connect to Trash-E Telegram Bot
3. Call send message function whenever we need to send the user a message.

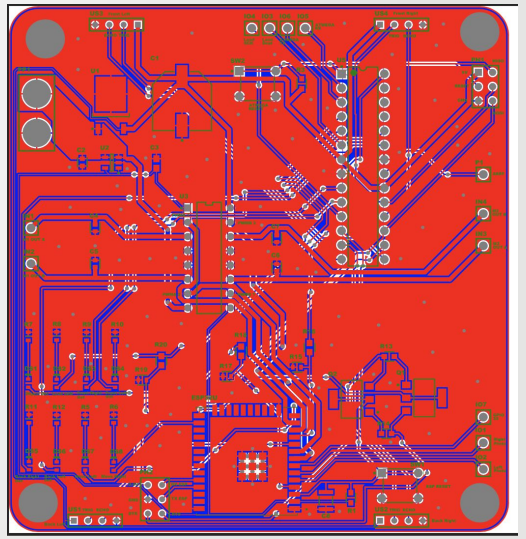
```
void send_to_bot(void)
{
    esp_err_t ret = nvs_flash_init();
    if (ret == ESP_ERR_NVS_NO_FREE_PAGES || ret == ESP_ERR_NVS_NEW_VERSION_FOUND) {
      ESP_ERROR_CHECK(nvs_flash_erase());
      ret = nvs_flash_init();
    }
    ESP_ERROR_CHECK(ret);

    wifi_init_sta();

    xTaskCreatePinnedToCore(&http_test_task, "http_test_task", 8192*4, NULL, 5, NULL,1);
}
```

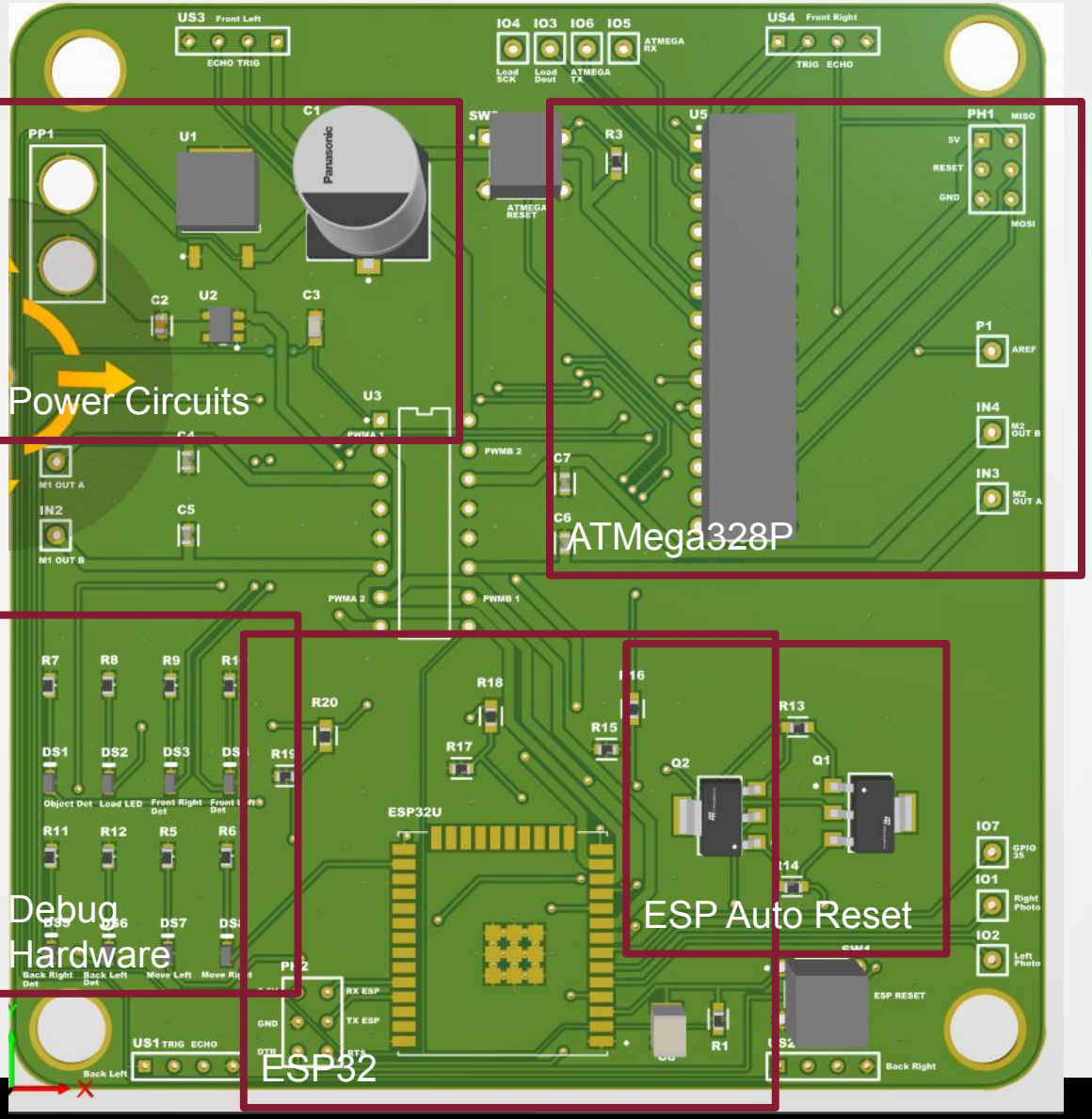
```
    if(objectDetected)
    {
        int x = 0;
        stop();
        while(objectDetected)
        {
            if(x >= 5000)
            {
                send_to_bot();
            }
            stop();
            objectDetected = digitalRead(ULTRA_DETECTION);
            x++;
        }
    }
}
```

PCB Design


Battery Control
Power
Voltage Regulators
ESP Group
ESP Programming Control
ESP Reset
ESP32
Open Ports
L293DNE Motor Driver
ATmega Group
ATmega328p
ATmega Reset
ATmega Program
Ultrasonics
5V to 3.3V Level Shifters
LEDs






Power Circuits
ATMega328P
Debug Hardware
ESP Auto Reset
ESP32

BE REVOLUTIONARY™

Challenges

University of Massachusetts Amherst

Challenges Faced During CDR

- Mechanical
  - Lack of tracks
  - Caster wheels
  - Motor limitations
  - Chassis alignment
  - Custom part sourcing
- Technical
  - Underpowered motor driver
  - Migration of load cell code
  - Fine-tuning motor driver
  - GPS connectivity (inside)
  - Bare ESP32 Programming
- Logistical
  - Shipping delays
  - Slow ramp-up to PCB design
  - Lack of testing phase
  - Lack of group cohesion

BE REVOLUTIONARY™

COST ESTIMATE

University of Massachusetts Amherst

Cost Estimate

Costs (Up to CDR)

Component	Predicted Cost
GPS	$39.06
Motor Driver	$5.45
Battery	$34.95
Huzzah32 Feather*	$19.95
Proximity Sensor x4	$25.71
Testing Platform	$29.99
Load Sensor x4	$31.72
Load Cell Amplifier	$6.05
Photoresistor PCB Parts	$26.20
ESP32 + Adapter	$21.94
Main PCB Parts	$81.48
Antenna	$11.98
IC Regulator	$0.76
Motors*	$40.00
Aluminum Frame*	$38.00
Castor Wheels*	$20.00
Total	**$433.24**

*Items sourced from M5 so actual cost is zero

Costs Estimate (CDR to FPR)

Component	Predicted Cost
Main PCB	$30
Trash Bin	$20
Total	**$50**

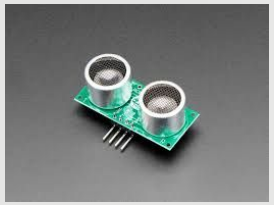
Total Cost Estimate: $483.24
Total Cost Without M5 Items: $365.29

GPS - Adafruit MTK3339 B.O.B

- Location of system
- High-sensitivity receiver
- Low power

Photocell - CdS photoresistor

- Light sensor
- Light = ~1KΩ
- Dark = ~10KΩ



Ultrasonic Sensor - HC-SR04 x 4

- Very General Purpose
- Can measure distance

Lithium ion Battery Pack

- 5000mAh
- 14.8 V
- ~ 5 hours run time

Motors x2

- recommended by course staff
- found in M5
- manually tested

Load Sensor x 4

- Cheap
- High Measurement range (up to 110kg or ~242.5 lbs)

ESP32 MCU
- previously was ESP32 feather board
- Wifi enabled
- Many GPIO pins
- Many ADC I/O pins
- SPI / I^2C (For GPS and Communication across boards/chips)

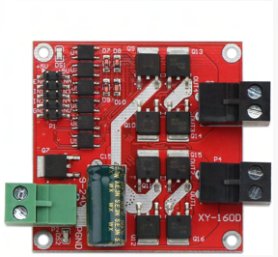
ESP32 Antenna

- 2.4GHz Antenna
- 20 meters added distance

Atmega328-PU MCU
- previously was RedBoard Qwiic
- General purpose MCU
- Cheap
- Analog & Digital IO
- Easy to prototype

Load Cell Amplifier - HX711

- 24 bit precision
- Read the changes in the resistance



Motor Driver - Dual L298

- Up to 27VDC & 7A
- H-Bridge for precise motor control

BE REVOLUTIONARY™

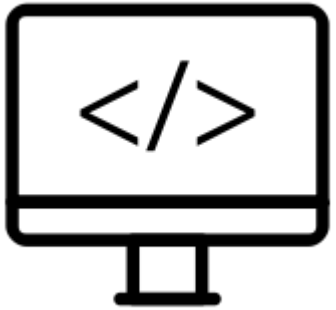
Responsibilities

University of Massachusetts Amherst

Team Member Responsibilities

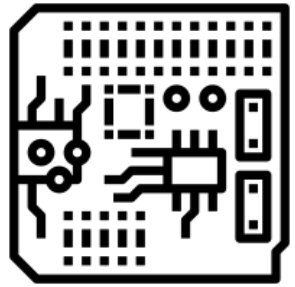
John Diep

- Software Lead
- Migration to Embedded C
- User Interface & Scheduler
- Systems Integration



Jasmine Hickey

- PCB Lead
- PCB Design
- Systems Integration



Smit Patel

- Budget Lead
- Motor Enclosure Scale-Up
- Antenna R&D
- GPS Navigation Design & Integration
- Systems Integration

Stephen Townsend

- Team Coordinator
- Motor Enclosure Scale Up
- GPS Navigation Design & Integration
- Systems Integration

BE REVOLUTIONARY™

FPR Deliverables

University of Massachusetts Amherst

FPR Plan

Changes to be made for FPR:

- PCB Revisions
  - Updated motor driver
  - Load cell amplifier
  - Or-gate addition
- Mechanical Changes
  - Worm drive gearbox
- Design Changes
  - Add trash receptacle
- Software Changes
  - Implement GPS secondary navigation and load detection
  - User scheduling of pickup time
  - Differentiation of home and pickup location

Testing Plan

- Real world testing
- See previous testing plan

- FPR Demonstration
  - Likely outside (weather permitting)
  - Demo of primary and secondary tracking systems
  - Demo of user interface

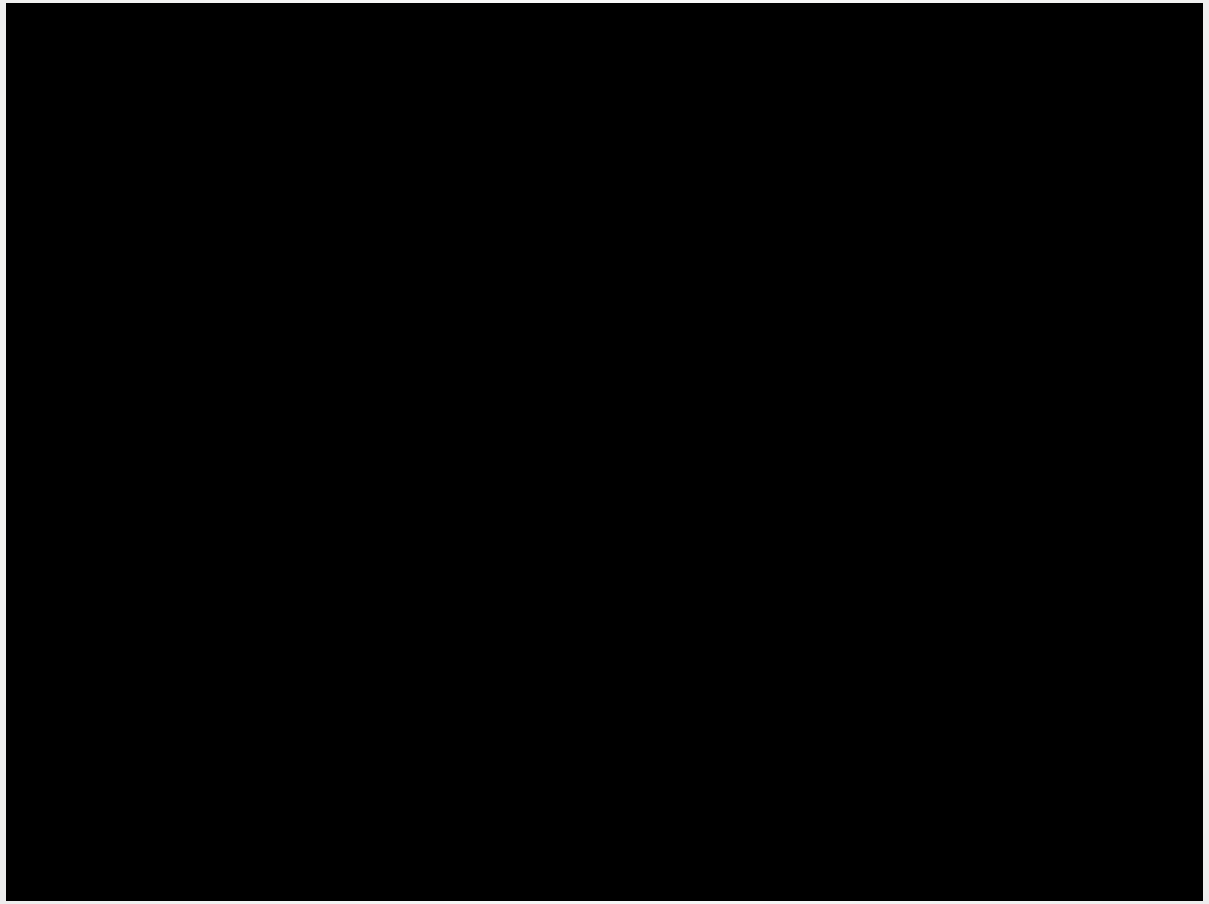
FPR Gantt Chart

GANTT CHART (POST CDR -> FPR)							
		Spring Break	T-4 weeks	T-3 weeks	T-2 weeks	T-1 week	FPR
Task to be Done	Team Members	3/14	3/21	3/28	4/4	4/11	4/18
PCB Revisions	John, Jasmine						
GPS Integration	Smit						
UI Integration	Stephen						
Scheduling	John						
Testing	All						

BE REVOLUTIONARY™

Demonstration

University of Massachusetts Amherst




QUESTIONS?
University of Massachusetts Amherst
BE REVOLUTIONARY™