**Technical Design Document**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| **Date** | **Revision** | **Release Notes** |
| 4/26/24 | 01 | Initial Release |
|  |  |  |
|  |  |  |
|  |  |  |

Contents

[1 Mechanical Documentation 3](#_Toc165129194)

[1.1 System Level Assembly Drawing 1: Interactive Urban Track 3](#_Toc165129195)

[1.2 System Level Bill of Materials 3](#_Toc165129196)

[1.3 Part Drawings 4](#_Toc165129197)

[1.3.1 Part 2004: ESP-32 Case 4](#_Toc165129198)

[1.3.2 Part 3003: Single Mounting Point 5](#_Toc165129199)

[1.3.3 Part 3004: Double Mounting Point 5](#_Toc165129200)

[1.3.4 Part 4002: Yield Sign 6](#_Toc165129201)

[1.3.5 Part 4003: Stop Sign 6](#_Toc165129202)

[1.3.6 Part 4004: Stop Light 7](#_Toc165129203)

[1.3.7 Part 4005: Railroad Crossing 7](#_Toc165129204)

[2 Electrical Documentation 8](#_Toc165129205)

[2.1 System Level Wiring Diagram Drawing 8](#_Toc165129206)

[2.2 Wiring Diagram BoM 8](#_Toc165129207)

[2.3 Printed Circuit Boards (PCB) 8](#_Toc165129208)

[2.3.1 Stop Light PCB Schematic 8](#_Toc165129209)

[2.3.2 Stop Light PCB Layout Images 9](#_Toc165129210)

[2.3.3 Stop Light PCB Bill of Materials 10](#_Toc165129211)

[2.3.4 Railroad Crossing PCB Schematic 10](#_Toc165129212)

[2.3.5 Railroad Crossing PCB Layout Images 10](#_Toc165129213)

[2.3.6 Interactive Element Receiver PCB Schematic 11](#_Toc165129214)

[2.3.7 Interactive Element Receiver PCB Layout Images 12](#_Toc165129215)

[2.3.8 Interactive Element Receiver PCB Bill of Materials 13](#_Toc165129216)

[3 Software Documentation 14](#_Toc165129217)

[3.1 ESP Interactive element network 14](#_Toc165129218)

[3.1.1 Instructions to Compile and Use 14](#_Toc165129219)

[3.1.2 File Names and Purpose 15](#_Toc165129220)

[3.2 ROS Code. 16](#_Toc165129221)

[3.2.1 Directions for Initial Build 16](#_Toc165129222)

[3.2.2 Basic Use Instructions 16](#_Toc165129223)

[3.2.3 Code Structure 17](#_Toc165129224)

[4 Design Files 19](#_Toc165129225)

[5 User Operation Video 20](#_Toc165129226)

# Mechanical Documentation

## System Level Assembly Drawing 1: Interactive Urban Track

The System Level Assembly drawing of the project is shown in Figure 1 from the “System Level Diagram v1.dwg”.

A diagram of a device

Description automatically generated

Figure 1 - System Level Assembly Drawing

## System Level Bill of Materials

The System Level Bill of Materials is in Table 1, and also in the file “BOM.xlsx”.

Table 1 - Bill of Materials for Interactive Urban Track

A table of information with text

Description automatically generated with medium confidence

## Part Drawings

### Part 2004: ESP-32 Case

Part 2004: ESP-32 case is shown in Figure 2 from the file “ESP Controller Box Drawing.dwg”.

A blueprint of a computer component

Description automatically generated

Figure 2 - ESP-32 Case Drawing

### Part 3003: Single Mounting Point

Part 3003: Single Mounting Point is shown in Figure 3 from the file “Single Mounting Point Drawing v2.dwg”.

A blueprint of a vent hood

Description automatically generated

Figure 3 - Single Mounting Point Drawing

### Part 3004: Double Mounting Point

Part 3004: Double Mounting Point is shown in Figure 4 from the file “Double Mounting Point Drawing v4.dwg”.

A blueprint of a mechanical design

Description automatically generated with medium confidence

Figure 4 - Double Mounting Point Drawing

### Part 4002: Yield Sign

Part 4002: Yield Sign is shown in Figure 5 from the file “Yield Sign Drawing.dwg”.

A blueprint of a triangle

Description automatically generated

Figure 5 - Yield Sign Drawing

### Part 4003: Stop Sign

Part 4003: Stop Sign is shown in Figure 6 from the file “Stop Sign Drawing.dwg”.

A blueprint of a stop sign

Description automatically generated

Figure 6 - Stop Sign Drawing

### Part 4004: Stop Light

Part 4004: Stop Light is shown in Figure 7 from the file “Assembled Light Drawing v3.dwg”.

A blueprint of a mechanical design

Description automatically generated

Figure 7 - Stop Light Drawing

### Part 4005: Railroad Crossing

Part 4005: Railroad Crossing is shown in Figure 8 from the file “Assembled RR Barrier Drawing v2.dwg”.

A drawing of a rail road crossing

Description automatically generated

Figure 8 - Railroad Crossing Drawing

# Electrical Documentation

## System Level Wiring Diagram Drawing

The system level diagram is shown in

A computer screen shot of a circuit board

Description automatically generated

## Wiring Diagram BoM

The wiring diagram BOM is shown in Table 2 from the file “Wiring\_Diagram\_BOM.xlsx”.

Table 2 - Wiring Diagram BOM

A close-up of a computer screen

Description automatically generated

## Printed Circuit Boards (PCB)

### Stop Light PCB Schematic

Three custom PCBs were created for this project. The first one is the Stop Light PCB whose schematic can be seen in Figure 9 from the file “Light Pcb.kicad\_sch”.

A screenshot of a computer

Description automatically generated

Figure 9 - Stop Light PCB Schematic

### Stop Light PCB Layout Images

The overall layout of the Stop Light PCB can be seen in Figure 10 from the file “Light Pcb.kicad\_pcb”.

A computer screen shot of a computer

Description automatically generated

Figure 10 - Stop Light PCB Layout

### Stop Light PCB Bill of Materials

The BOM for the Railroad Crossing PCB is shown in Table 3 from the file “StopLight \_PCB\_BOM.xlsx”.

Table 3 - Stop Light PCB Bill of Materials

A white background with black lines

Description automatically generated with medium confidence

### Railroad Crossing PCB Schematic

The second custom PCB is the Railroad Crossing PCB whose schematic can be seen in Figure 11 from the file “Servo\_Adapt.kicad\_sch”.

A computer screen shot of a computer

Description automatically generated

Figure 11 - Servo Adapter Schematic

### Railroad Crossing PCB Layout Images

The overall layout of the Servo Adapter PCB can be seen in Figure 12 from the file “Servo\_Adapt.kicad\_pcb”.

A computer screen shot of a computer

Description automatically generated

Figure 12 - Servo Adapter PCB Layout

### Railroad Crossing PCB Bill of Materials

The BOM for the Railroad Crossing PCB is shown in Table 4 from the file “Railroad\_Crossing\_PCB\_BOM.xlsx”.

Table 4 - Railroad Crossing PCB Bill of Materials

A close-up of a computer screen

Description automatically generated

### Interactive Element Receiver PCB Schematic

The final custom PCB that was made is the Interactive element PCB, whose schematic can be seen in Figure 13 from the file “Esp Controller.kicad\_sch”.

A screenshot of a computer program

Description automatically generated

Figure 13 - Interactive Element Receiver Schematic

### Interactive Element Receiver PCB Layout Images

The overall layout of the Interactive Element Receiver Board PCB can be seen in Figure 14 from the file “Esp Controller.kicad\_pcb”.

A computer screen shot of a computer

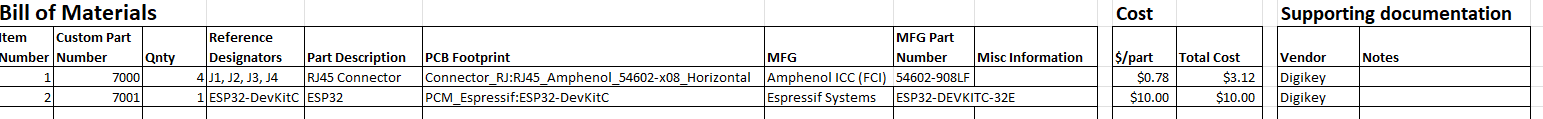
Description automatically generated

Figure 14 - Interactive Element Receiver Board

### Interactive Element Receiver PCB Bill of Materials

The BOM for the Railroad Crossing PCB is shown in Table 5 from the file “Interactive\_Element\_PCB\_BOM.xlsx”.

Table 5 - Interactive Element Receiver PCB Bill of Materials



# Software Documentation

## ESP Interactive element network

The interactive elements in the physical space are linked via a network of ESP-32 boards, connected via ESP-NOW protocol and serial communications via USB.

### Instructions to Compile and Use

To Flash the Interactive Element Receiver (ESP Receiver) Simply select the desired function, eg Train Crossing. Open the file in Arduino IDE, with the Espressif Board Library installed as shown in Figure 15.

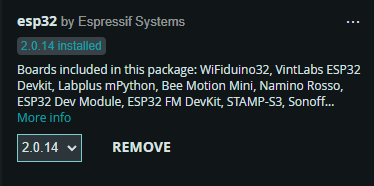


Figure 15 - Installed Espressif Board Library

Select the appropriate COM port from the top drop down menu as shown in Figure 16.

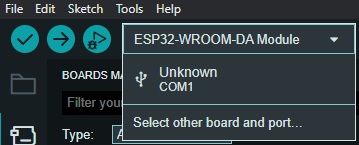


Figure 16 - COM Port Drop Down Menu

Type “wroom” in the boards search bar. Select the ESP32-WROOM-DA module, and the Com port. Select “Ok” as shown in Figure 17.

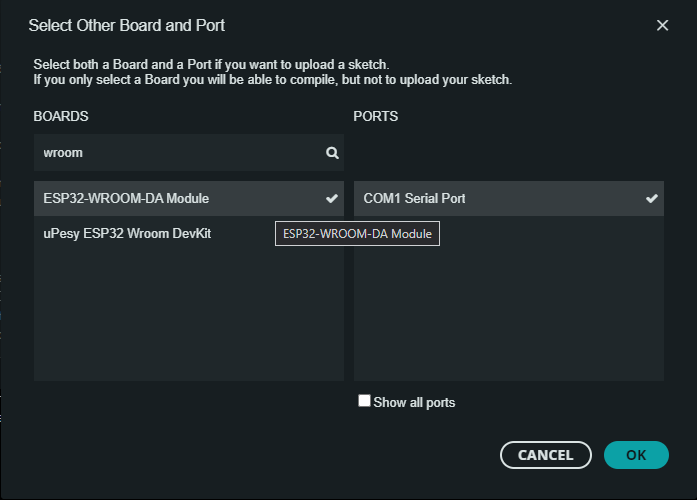


Figure 17 - Select OK

Then select “upload” (The Arrow button) as shown in Figure 18.

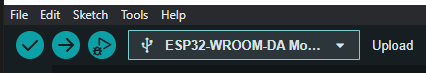


Figure 18 – Select Upload

Wait for the upload to complete, and that’s it.

Should the process fail, check the com port is correct, and in some cases the “boot” button on the ESP32 may need to be held down to load.

### File Names and Purpose

* + Interactive Elements: Contains the Arduino code files, and PCBs for the physical interactive elements.
    - Animals: Contains the animal models for the track.
    - Board Design and IO: PCBs and associated Gerbers can be found here
    - DirectRemote: Code file for a remote control using the ESP Network, No ROS integration (Optional extra)
    - Driver for 38 pin board: Driver files to connect to the serial board on the ESP32
    - Files to start: Template Arduino files for ESP 32 receiver nodes.
    - Sender: The serial to ESP network bridge code. Connected to Pi via USB cable
    - Stoplight: Code for the ESP receiver for both 3 and 4 way light elements.
    - Train Crossing: Code for the ESP receiver for the movable train crossing element
    - Room dimensions v1: A 1:1 scale model in Fusion of Votey 308, with some immovable tables included.

## ROS Code.

### Directions for Initial Build

After installing Ubuntu 22.04 on the machine of choice.

1. Connect computer to Internet
2. Download and install NoMachine for the system architecture (AMD64 or ARM64)
3. Open a new terminal: Run this command set: this will pull down the github repo, with all the code for the car.
   1. git clone https://github.com/UVMAirLabTrack/Track23 RaspberryCore
   2. cd home/[user]/CoreRaspberry/Setup\_files
   3. sudo chmod a+x control-install.sh Installs base software, suitable to a SBC or Pi
   4. Or sudo chmod a+x base-install.sh This installs a full software loadout, better suited to an x86 machine.
   5. Sudo chmod a+x ros-humble-std.sh This should install the full ROS2 Humble distribution
   6. cd /home/[user],RaspberryCore/CoreRaspberry/ Change your user to match
   7. colcon build --symlink-install This should build out the packages for the Pi. Once complete, you’re ready to run the worlds!
   8. Close the terminal. If you don’t things will appear to not work, the terminal does not live update settings.

### Basic Use Instructions

Provided the setup has gone correctly, (This is open source software, compatibility is impossible to guarantee) These basic commands will get a world up and running. Note: All launch files are located under world\_gen for simplicity. And for most commands, typing the first 3 letters then hitting tab for autocomplete is helpful.

1. Open a new terminal window for each separate command.
   1. ros2 launch world\_gen track\_launch Launches the Interactive network and serial coms.
   2. Ros2 launch world\_gen all\_viewer\_launch Loads up RVIZ and the marker communication systems. Note, if the car marker is not visible, select “car” in the left pane, and adjust the topic to “carmesh\_1” It should be visible in the pulldown menu.
   3. If you want to change the base track. Navigate to CoreRasperry/worlds and open world\_select.txt. move the “y” to the desired world map. Then reload the all\_viewer\_launch command. (CTRL+C cancels running commands) A screenshot of a computer program

      Description automatically generated
   4. Ros2 run map\_transforms reset\_car. This activates the ability to reset the car’s positional odometry with respect to the map. Press enter on the terminal window to activate, after 5 seconds the reset will stop collecting data and reflect the car’s return to the start location.
   5. To relocate interactive elements to different track positions open the respective [world]\_markers.txt file. The structure used is [Topic,zone,location] A screenshot of a computer program

      Description automatically generated . so the first element is which marker to display at a given location. Fourway\_1 for example is the fourway light, element 1. 3way is the zone for the marker to be located in, in this case the 3 way intersection. The final location tag is which of the marker locations available at the 3 way to place it.

### Code Structure

* Github Repository:
  + Car Commands: A set of text instruction files to deploy basic functions on the Yahboom R2 Car, commands are stock.
  + CoreRaspberry: This contains the code ROS code.
    - Build: Empty until package is deployed, will contain system level files
    - Control: The text files used to control the light and train states can be found here.
    - Example\_package: exactly what it says, for building new packages in the future, might be better to reference an existing package at this point.
    - Install: empty until deployed, only for advanced users, direct edits not recommended
    - Log: log files
    - Src: source code. The python files and most reference files can be found here. Packages overview document lists the included packages and their file dependencies.
    - Worlds: contains the world mesh files, and the textual control files. For full details see packages overview document. For each world, here are the main files:
      * World.dae The actual world mesh file. New ones can be created by generating a 3d model in CAD of choice, exporting as a .obj file with meter scaling, then using blender to convert from .obj to .dae.
      * World.txt Contains the pose and color data for the world mesh.
      * World\_markers.txt the reference file used to place a virtual element, at a specific location.
      * World\_markers\_loc.txt the coordinate position, angle, and zone for locations in the virtual space.
      * Markers: this folder contains a pose file for each individual marker used in the system.
  + Linux Commands: Some basic Linux script files to simplify deploying the environment.
  + Other Code: Python files not attached to ROS, such as HSV decoders, and Object detection files.
  + Setup Files: Text files for initial configuring of the Raspberry Pi.
  + Dockerfiles (not in a subfolder)
    - Dockerfile-r2-base: The base image provided with the R2 car. Here for reference.
    - Dockerfile-viz-r2-humble: Deploys the track visualizer environment on the R2 car. Pairs with the viz\_bringup shell script in Linux commands.
    - Dockerfile-Viz-Humble: For deployment of the visualizer environment. Currently works with Linux, Networking with windows not yet functional, but otherwise can be run on windows as well.

# Design Files

Table 6 - Design Files

|  |  |  |
| --- | --- | --- |
| **Location** | **File Name** | **Description** |
| EduSourced/Files/3D Models | AllTrackModels.zip | All solid models for the different track layouts for the project |
| EduSourced/Files/Design Files | AllTrackSectionModels.zip | All solid models for the track sections in the project |
| EduSourced/Files/Design Files | AllOtherModels.zip | All other solid models included in the track, including interactive elements and cases. |
| EduSourced/Files/Design Files | Drawings.zip | All drawing files for custom parts and system level overview. |
| EduSourced/Files/BOM | Interactive\_Element\_ Receiver\_PCB\_BOM.xlsx | BOM for the Interactive Element Receiver PCB |
| EduSourced/Files/BOM | Railroad\_Crossing\_ PCB\_BOM.xlsx | BOM for the Railroad Crossing PCB |
| EduSourced/Files/BOM | StopLight\_PCB\_BOM.xlsx | BOM for the Stop Light PCB |
| EduSourced/Files/BOM | Wiring\_Diagram\_BOM.xlsx | BOM for the Wiring Diagram |
| EduSourced/Files/BOM | BOM.xlsx | BOM for entire system |
| EduSourced/Files/PCBs | InteractiveElement Receiver.zip | All design and Gerber files for Interactive Element Receiver PCB |
| EduSourced/Files/PCBs | RailroadCrossing.zip | All design and Gerber files for Railroad Crossing PCB |
| EduSourced/Files/PCBs | StopLightPCB.zip | All design and Gerber files for Stop Light PCB |
| EduSourced/Files/PCBs | System Diagram.zip | Wiring Diagram schematic |
| <https://github.com/>  UVMAirLabTrack/Track23.git | All Code | All files and packages listed in the Software section and in the user manual. |
| EduSourced/Files/ | Setup Manual.docx | User Manual |

# User Operation Manual

A User Operation Manual has been written to instruct future users how to set up and operate the project. This manual is the file “Setup Manual.docx”. It contains all physical track setup information and a more in depth explanation of the software section in this document.