

| **Course Title:** | **Microprocessor Systems** |
| --- | --- |
| **Course Number:** | **COE538** |
| **Semester/Year (e.g.F2016)** | **F2024** |

| **Instructor:** | **Dr. Lev Kirischian** |
| --- | --- |

| *Assignment/Lab Title:* | **Final Project** |
| --- | --- |

| *Submission Date:* | **Nov 26, 2024 12:00 PM** |
| --- | --- |
| *Due Date:* | **Nov 26, 2024 12:00 PM** |

| **Student**  **LAST Name** | **Student**  **FIRST Name** | **Student**  **Number** | **Section** | **Signature\*** |
| --- | --- | --- | --- | --- |
| **Ahmad** | **Mariam** | **501180966** | **04** |  |
| **Kaden** | **Yu** | **501110592** | **04** | **ky** |

\*By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a “0” on the work, an “F” in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: <https://www.torontomu.ca/content/dam/senate/policies/pol60.pdf>

**Project Overview**

In this report for the Robot Guidance Challenge, we explore the complexities of fundamental autonomous guidance systems, guidance logic hierarchies, and the troubleshooting associated with a robot interacting with its surroundings. This lab's major goal was to fully program an eebot bot that has an HCS12 microcontroller and cadmium sulfide photoresistor sensors, which serve as the bot's principal navigation mechanism. As depicted in Figure 1, the eebot must use these sensors to find its way through a maze that has several junctions, curves, and dead-end walls.

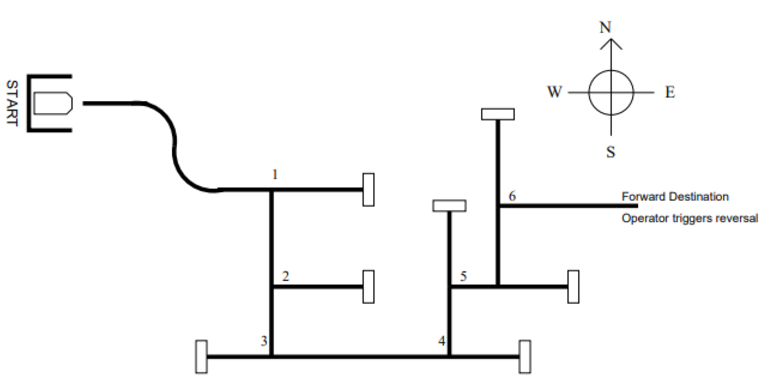


Figure 1: eebot maze

Examples of autonomous devices and navigation in the actual world can be used to fully understand the reasoning behind the robot guiding system that was studied in this lab. The first step in doing so is to comprehend the eebot's states. Figure 2 shows a state machine diagram that depicts the states. Given the current action or state, state machines give a visual depiction of how the robot "thinks" and decides what to do next.

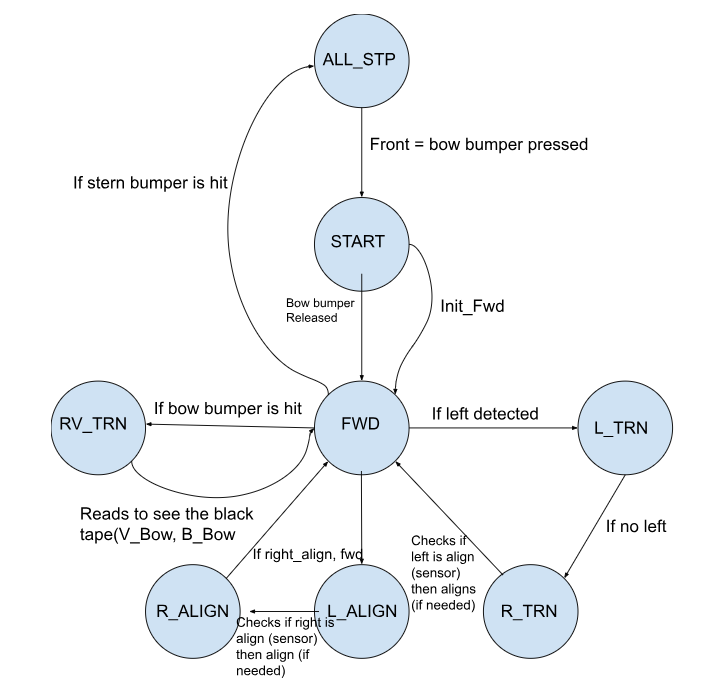


Figure 2: State Machine Diagram

The state machine illustrated in Figure 2 was designed specifically to guide the eebot's navigation through the maze. It outlines all possible actions the bot can perform, represented as individual states, along with the conditions and inputs that trigger state transitions. The operational states include: Start (START), Forward (FWD), Left Turn (L TURN), Right Turn (R TURN), Left Align (L ALIGN), Right Align (R ALIGN), Reverse Turn (REV TURN), and All Stop (ALL STOP). These states form the basis of the bot's decision-making and navigation strategy.

The diagram's arrows indicate the bot’s next state, provided specific conditions are met. Upon being powered on and ready to begin, the bot enters the START state. When the front bumper is activated, it transitions to the FWD state and starts navigating the maze. Subsequently, the bot’s state changes are dictated by sensor and bumper inputs. For instance, when the sensors detect an intersection or turn, the bot determines the direction of open paths and follows the appropriate line. At a dead end, the front bumper triggers as it contacts a wall, prompting the bot to execute a 180-degree turn before pursuing an unexplored path. To stop the bot, operators can activate the rear bumper, transitioning it to the ALL STOP state, halting all motor functions.

**Code Implementation and Troubleshooting**

A crucial element of this project was ensuring seamless communication between sensor readings, enabling the bot to make well-informed decisions while navigating the maze. To identify potential paths along the black line, inputs from multiple sensors needed to be processed simultaneously. Each unique combination of the six sensors, shown in Figure 3, corresponded to a specific route option.

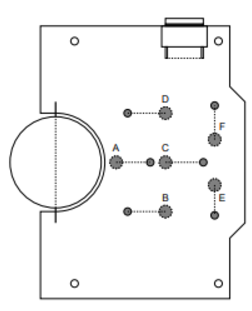


Figure 3: eebot Photresistor Sensor Layout

For instance, when the bot was in the forward state and sensor B detected a black line, it indicated an available left path. In our code, we prioritized left turns over other directions, as the correct maze path required more left turns than right. Early challenges arose from this approach, as the bot initially struggled to follow the line accurately and recognize necessary turns. The bot frequently veered off course, leading us to suspect that the sensor values and thresholds were not properly calibrated. After extensive trial and error, we optimized these values, allowing the bot to remain on the line.

Another issue we encountered was the bot overshooting turns and failing to adjust properly on curved sections of the maze. This was resolved relatively easily by fine-tuning timing values and sensor input logic to help the bot realign with the line. Once these adjustments were implemented, the bot successfully navigated straight sections, executed turns, and maintained alignment throughout. The 180-degree turns posed minimal difficulty, as they had been addressed in earlier lab exercises.

**Conclusion**

Overall, the lab was a success, as our robot was able to navigate the maze without veering off the path or getting lost. While the final version of the code performed effectively, achieving this result required countless hours of iterations and trial and error. Through this process, we gained invaluable experience in correlating inputs and outputs and had our first hands-on interaction with calcium sulfide photoresistor sensors. Additionally, working with assembly as a coding language was a new and challenging experience, requiring us to learn its syntax and software intricacies.

Although we achieved the project’s main objective, there are areas we would improve if given the chance to continue refining it. One issue we identified was the robot’s relatively slow speed while navigating the maze. While the bot’s accuracy was excellent, its slow pace made it take a considerable amount of time to complete the course. During development, we tried increasing its speed, but this came at the cost of reduced accuracy in turns and line recognition—trade-offs we felt were unacceptable. If revisiting the project, we would focus on further optimizing our navigation algorithms and thresholds to ensure the bot could maintain accuracy at higher speeds.

**References**

[1] Hiscocks, P, COE538 Microprocessor Systems Lab Project Robot Guidance Challenge.

[2] Hiscocks, P, COE538 Microprocessor Systems The eebot Guider.