ESB Homework Rookie's Bootcamp

Sanjana Taware

EECS Homework

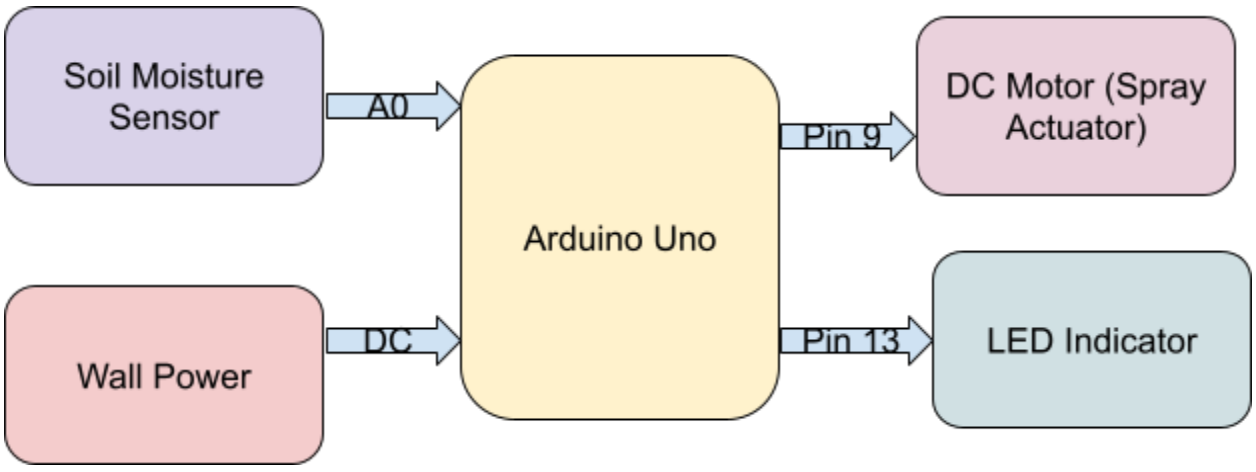
Vedang has to take care of his roommate's plant (true story), however he tends to overwater or underwater the plant too often. Design a plant monitoring and watering system for him. You will be required to deliver the following developed in Tinkercad:

- A block diagram of the system (and a 1 paragraph explanation of how the system interacts with the outside world)
- A TinkerCAD simulation of the breadboarded device. (Use an Arduino Microcontroller Board as a starting point)
- Data outputs and key points for analysis (1 paragraph)

Key Design Points:

- This plant will be misted with a spray bottle (already purchased) which has an actuation mechanism in place that connects with a motor.
- Ideally this system should get power from a standard wall supply (Stumped? Read up more on the Arduino Uno)
- The system should have some indicator of being on when it's not actuating the spray bottle.

Block Diagram:



The system interacts with the environment by continuously monitoring soil moisture via the sensor. If moisture drops below a set threshold, the Arduino activates the motor to mist the plant. The LED stays on when the system is idle (powered) but turns off during watering to indicate

activity. Power is supplied via a wall adapter connected to the Arduino, ensuring continuous operation.

Data Outputs & Key Analysis Points

The system generates three primary data outputs: soil moisture levels (analog values from 0–1023), motor activation timestamps/duration (tracking when and how long the spray bottle mists the plant), and LED status changes (indicating idle vs. active states). Key analysis focuses on calibrating the moisture threshold to match the plant's needs (a lower threshold for drought-tolerant plants, higher for moisture-loving varieties), optimizing watering frequency to prevent over/underwatering by adjusting code delays or sensor placement, and ensuring consistent power delivery by monitoring the LED's idle-state illumination. For example, if the motor activates too frequently, increasing the threshold or extending the delay between checks can reduce water usage. Additionally, logging moisture values via the Arduino's serial monitor can help refine the system's responsiveness to environmental changes.