AIN SHAMS UNIVERSITY

FACULTY OF ENGINEERING

CSE211s: Introduction to Embedded Systems

Final Milestone Delivery Report

Team Members

Name	ID
Moustafa Ahmed Hashem	2100467
Kareem Ehab Ibrahim	2100913
Michael George Naem Negeb	2100709
Karen Maurice William Adeeb	2100748
Mayar Walid Said Mostafa	2100953

Components used:

Components	Use
Tiva C	Micro-controller board that stores our code and controls its functionality.
GPS module	It is used to communicate with the global satellite positioning system.
Arduino	Used as a power supply.
Cables/Wires	Cables for the UART and wires to connect the inputs/outputs.
LCD	Displays the output we want the user to see (like distance and speed).
Variable Resister	Control the contrast of LCD
Computer	For writing the code that we're uploading in Tiva C

Explanation/Supported features:

- **GPS**

- We wrote a function that reads data from GPS and another function that filters data (latitude-longitude-speed) and store it.
- Then another function that finds the distance and speed between two points (that will lead us to find the distance between the start and end points as well as know when we exceed 100m and calculate AVG speed).
- **Future enhancement:**
  Use more modern GPS to Send more accurate points.

- **LEDs**

- We wrote a function that lights the LED:
  1- the green LED when the distance is greater than 100m.

2- the blue LED when between 95 to 100m.
3- the red LED when less than 95m.

- **Future enhancement:**
  We can make a function that blinks a white LED when it is loaded to send points to draw.

- **EEPROM**

- We created a temporary one-dimensional array that holds four words representing two points of the path read from GPS where each point takes two consecutive words, the first for the latitude and the second for the longitude. We can control the time of each point received by changing the delay function from SYSTICTIMER to decrease/increase the rate of received points.
- After each loop, the two points stored in this temporary array are transferred to be stored in EEPROM and then a variable is incremented to store the next two points read by GPS in the consecutive places in EEPROM.
- Max point we can store 8 points only because EEPROM max 16 words and for each point we need 2 words.
- **Future enhancement**:
  Use a memory SD card to store more points with more accuracy.

- **the trajectory**

1- **send trajectory:** We created a function “draw_path” that receives two points at a time from those stored in EEPROM and sends each two received points to IntelliJ character by character and in the same point between latitude and longitude we send “,” and after each point we send “@” and IntelliJ keeps receiving the characters sent by the “draw_path” function till it receives “^” which means that it has received all 8 points stored in EEPROM which defines the trajectory path.
2- **receive trajectory:** We receive character by character from each point store it in a string and replace each “@” with the new line we store each point alone in a new variable and we make a condition to

check if any point that is not valid was neglected so the trajectory will be drawn without error.

3- **draw trajectory**: IntelliJ uses these points and rewrites them in JavaScript code to be uploaded through a browser and shows the trajectory path on a map on the browser.

- **Future enhancement**:
  We can after showing trajectory on pc use TFT to show the trajectory on it.

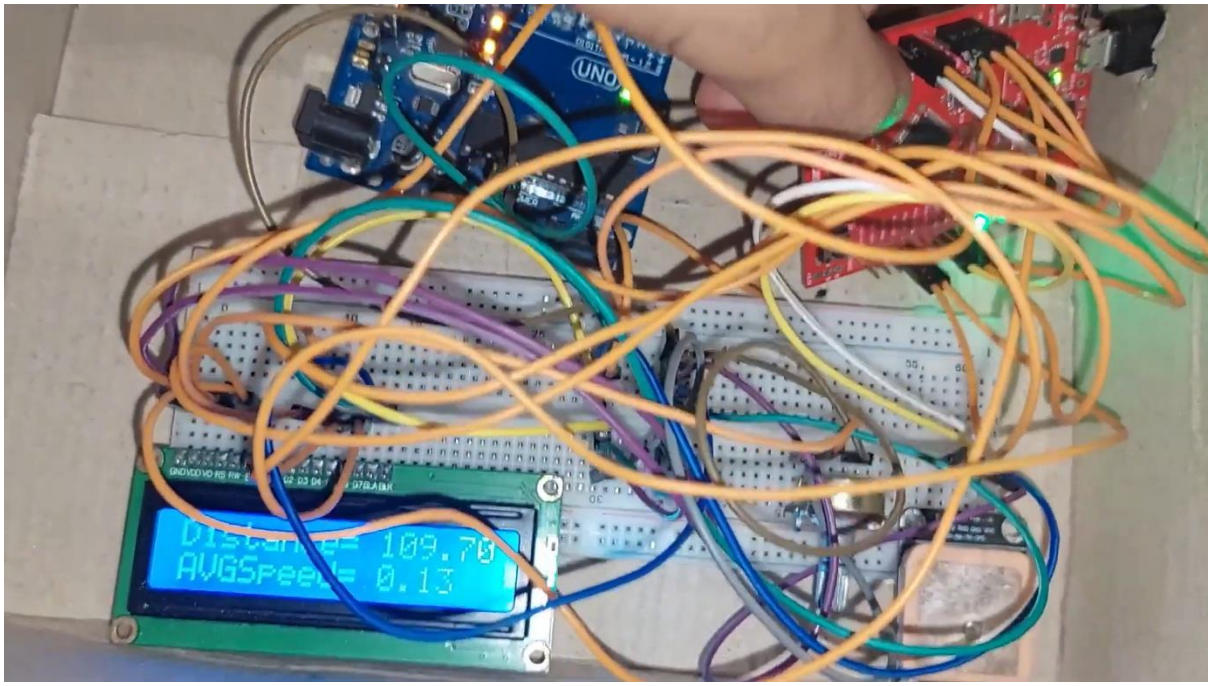
Connections:

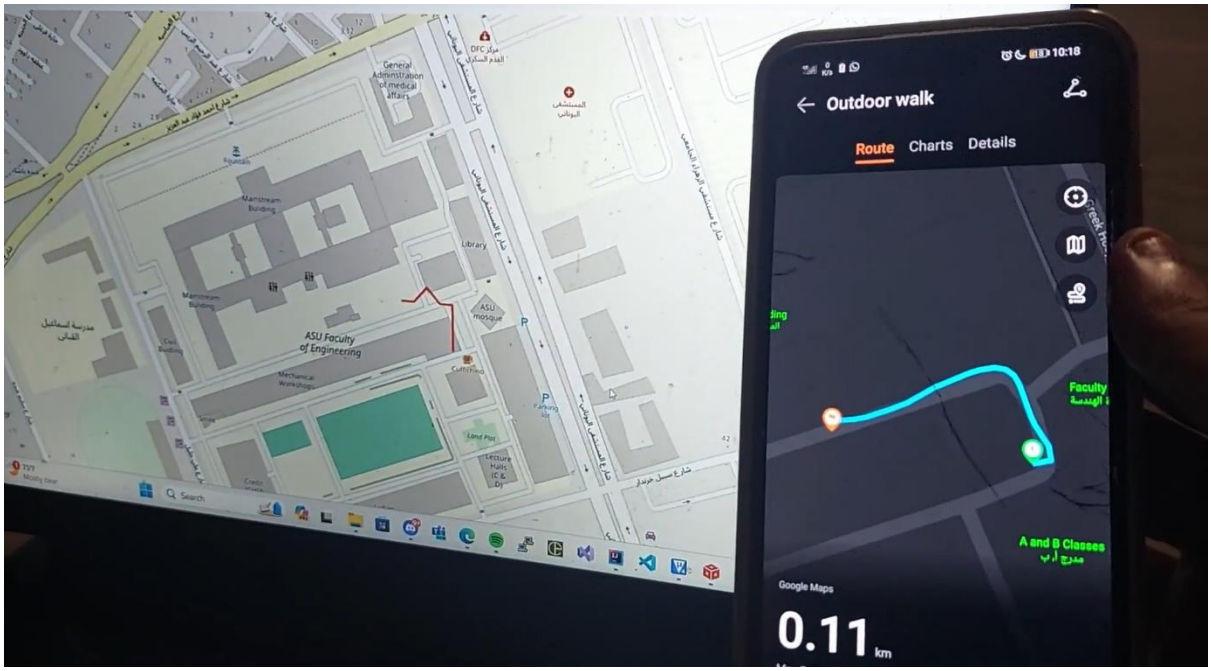
- **GPS**
  - **Connection setup:** We chose PORTE of Tiva C for the use of GPS.
  - **Configuration steps:** We made an initialization code and a driver for GPS.
  - **Additional functions/code that supports GPS:** functions "read_Data" &"do_Filter" that read and receive the data from GPS and then filter it to get the desirable meaningful data (latitude, longitude, directions(E, W, N, S) and speed) and finally store them in temporary variables that later will be sent to a function that stores them in EEPROM.
    A function "calculate_Destination" is added to calculate the distance between each two points sent to it and returns this distance to be added to a total distance accumulator.
- **LCD**
  - **Connection setup:** Port B was chosen for the LCD connection.
  - **Configuration steps:** We made a driver that initializes LCD.
  - **Additional functionalities that support LCD:** a function that writes in LCD, one that displays what's written, functions that adjust formatting, and delay functions for better observation of shown messages.

- **Tiva C**
  - **Connection setup:** UART 0 in port A connects Tiva C to pc and UART7 in port E connects Tiva C to the GPS module and port B to LCD.

- **All component was supplied by Arduino.**

Screenshots:





Link of video:

https://www.youtube.com/watch?v=u4c-sM-THLI

Link of GitHub repository:

https://github.com/MoustafaHashem/GPS_Tracking_System.git

Notes:

- We used the IDE code composer studio.
- We used IntelliJ as a serial port.
- We used JavaScript to upload code to API on the internet.
- We use Arduino as a power supply only.

Contribution:

Moustafa Ahmed Hashem	Main file
Kareem Ehab Ibrahim	LCD file & Trajectory file (JavaScript)
Michael George Naem Negeb	UART file & Trajectory file (java)
Karen Maurice William Adeeb	GPIO file & Trajectory file
Mayar Walid Said Mostafa	GPS file & SYSCTIMER file