GPS Tracking System

Project Description

The goal of this project is to develop an embedded system using C programming that gathers real-time positional coordinates while a microcontroller is in motion (GPS tracking system using TM4C123G LaunchPad) after power-on until a destination point is reached. The collected data will then be efficiently transferred to a personal computer (PC) and visualized on a map application.

There are three possible options for the definition of the destination point:

1. Once the moved distance exceeds 100m, stop adding new points to the trajectory and the last point added to the trajectory is the target destination.
2. Stop adding new points when a push button is pressed, and the last point added to the trajectory is the last destination.
3. Predefine the destination point in your code and when the system reaches this point, stop adding new points to the trajectory.

Other functions required by the system:

1. When the destination point is reached the build-in LED of the launchpad should be turned on.

2. The system should check for UART commands from the PC and if the PC sends the command ‘U’ then the system should send the stored trajectory to the PC.

Project Requirements:

Hardware:

1. A microcontroller development board (e.g., TivaC)
2. A GPS module (or a compatible sensor that provides positional data)
3. A personal computer (PC)
4. Connecting cables (USB, serial, etc.)

Software:

5. A C-development environment (e.g., Keil)
6. A PC-based application development environment suitable for data visualization (e.g., Python with Matplotlib or a mapping API)

Functional Requirements

1. After power-on, the system should configure the following interfaces:
    1. UART to interface with the GPS module
    2. UART to interface with PC
    3. Digital Output for built-in LED

2. The system should read GPS module data and wait until there is a GPS fix (Check GPS module datasheet to check how can this be done).

3. The first point read with GPS fix should be appended to the trajectory as the start point.

4. Your system needs to read the coordinates from the GPS module in a periodic manner to get your trajectory. Systick timer should be used to control rate of execution of main function depending on the update rate of the GPS module.

5. After reaching the destination point, the system should stop appending new points.

6. The trajectory of the distance will be drawn using software

7. The trajectory of the distance should satisfy the following criteria:
    1. The total distance between the start and the end point should be > 100 meters.
    2. The path from the start point to the end point should form a non-straight line that is similar to the provided baseline path below.

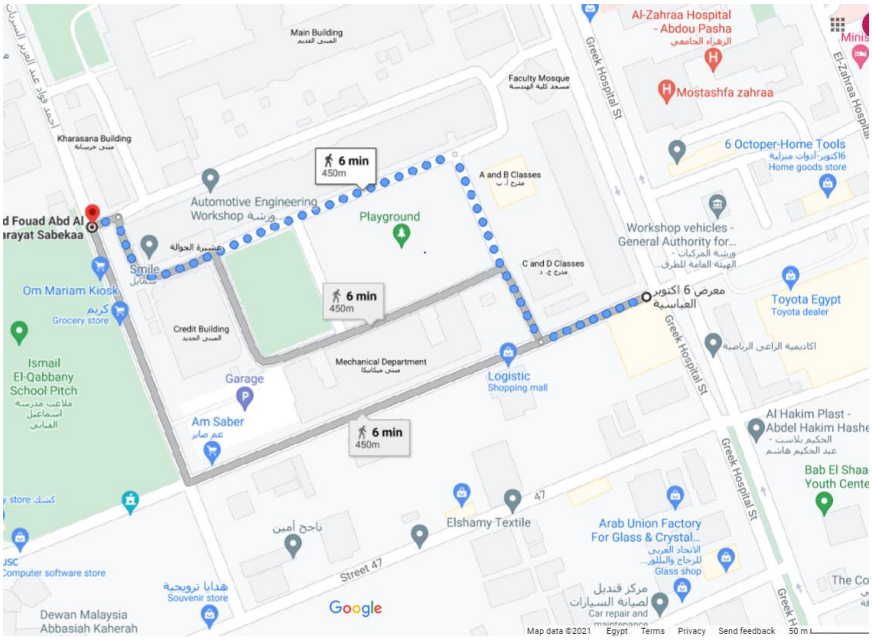

Figure 1:Basline path that you should follow

3. You have the freedom to select any starting point on google maps.

8. The scenario will be as the follows:
    1. The user will open google maps and start his program running on TivaC at the same time.
    2. The user should walk holding the kit and the opened google maps and walk till he/she exceeds the 100 meters.
    3. The user should follow the path which is given in Figure1 and walk in non-straight line.

    4. After reaching the specified destination, connect the launchpad to the PC and wait for ‘U’ command from the PC and then send your saved trajectory to the PC to be drawn. The process of storing and sending the coordinates can be done in one of the following steps, listed in order of preference from most to least preferred.
        1. After reaching the destination point, press on a switch to store the trajectory in EEPROM and change the color of the LED after data is saved to EEPROM and then you can power-down the kit. Whenever you power on the kit, connect it to PC. Your kit should listen and waits for a ‘U’ command to read the trajectory from EEPROM and send the trajectory to the PC to be drawn.

2. Before powering down the kit, connect the launchpad to the PC and when ‘U’ command is received, send the trajectory to PC.
3. Sending the coordinates live from the kit to the PC while walking (less preferable choice). In this case, you do not need to check for the ‘U’ command as you will send the coordinates continuously as you read them from the GPS module.

Milestones

First milestone

- You should make sure that you can flash your code from the IDE (e.g., Keil) to your kit.
- You should implement a function that initializes the ports of your microcontroller (e.g., configure GPIO ports).
- You should write a function that configures the UART of your kit to communicate properly with the GPS subsystem.
- You should implement a function that turns on the LED when the destination is reached (e.g., when the push button is pressed or when the distance exceeds 100 meters).

Second milestone (Final)

- You should write the function that parses the coordinates sent from the GPS in the form of ASCII and stores the coordinates, while the microcontroller is in motion.
- You should implement the software part that draws the trajectory.
- You should integrate the developed functions to form your program and test it.

Number of Students

The project team should be between 6-8 members.

Project Instructions

1. Download the kit header file from the below link to include it to use its defined macros in your code.

https://drive.google.com/file/d/1Gyt1VkYgfyEYHeF1VL6ivI9W2FQkB-GQ/view?usp=sharing

2. Your implementation should be in embedded C.
3. The demo video should be taken as one shot without cuts or edits.

First Milestone Delivery

1. The first milestone is expected to be delivered on 27th April at 11:59 pm.
2. The expected to be delivered:
   - Source code compressed in one zip file.
   - The team should push their codes on the GitHub repository. Each team member should contribute and push his/her part of the code on GitHub.
   - The team leader should attach the GitHub repository link of the team in the report.
   - A pdf file (report) contains the team member names and their IDs and screenshots for the output of testing of the LED.
   - This will be submitted on LMS.

Final Delivery (second milestone) and project discussion

1. The team should deliver source codes compressed in one zip file.
2. PDF document contains the explanation of supported features and what has been done to support such features (e.g., components used to support those features, connection setup, configuration steps, and explaining the addition of extra code/functions to support certain functionalities).
3. The team should deliver a video for the project. Upload your video on the drive/YouTube or any social platform and attach the **video link to the report**.
4. Snapshots of the software drawing as output
5. The video should be one shot showing you when you are walking from the start point to the target point showing that
   a. you exceeded the 100 meters,
   b. you walked in a non-straight line
   c. the output of the LED,
   d. and the drawing part of the trajectory on any software.

6. The team should push their codes on **GitHub repository**. Each team member should **contribute** and push **his/her part of code on GitHub**.
7. The team leader should attach the **GitHub repository link of the team in the report**.
8. A project discussion will be held.

Final Delivery Deadline (second milestone)

1. The deadline of the submission will be 12th May at 11:59 pm.
2. The project delivery files will be submitted on LMS.
3. The initial dates for the live demo will be held between 13th May and 14th May.

Evaluation

1. 25% of the marks for **individual** contribution especially the GitHub repository contribution.
2. 75% of the marks for the project team.

Note: A team member without contribution on the GitHub repo will get ZERO.