Design Overview

This project will focus on the construction of a transport logger. This logger will track, among other things, the GPS location, temperature, and acceleration of the tracker. The main idea behind this is to gather this data with the tracker, then present the data in a logical and efficient fashion. The user will be able to control what data appears by adjusting thresholds to display certain events that may have occurred when the tracker was en route to its destination. For example, the user will be able to set a maximum threshold temperature of 30’C. With this threshold, the PC application that processes the tracker’s data will use a red line for the route along parts where the temperature exceeded 30’C. This will provide the user with extremely simple visually feedback about the conditions that the packages went through en route to their destinations, also hopefully allowing them to pin point areas where damaged packages may have initially been damaged.

Several versions are planned for the tracker, starting off with a basic tracker that tracks GPS location, temperature and acceleration. Given more time, two additional possible sensors will be added. These possibilities include pressure, sound, humidity, luminosity, magnetic field and UV light sensors. The last version of the tracker will include added functionality to the PC application so that a user will be able to program in a route that they wish to travel, upload it to the tracker, then have the tracker inform them if they veer off route. An overview of the system is shown below.

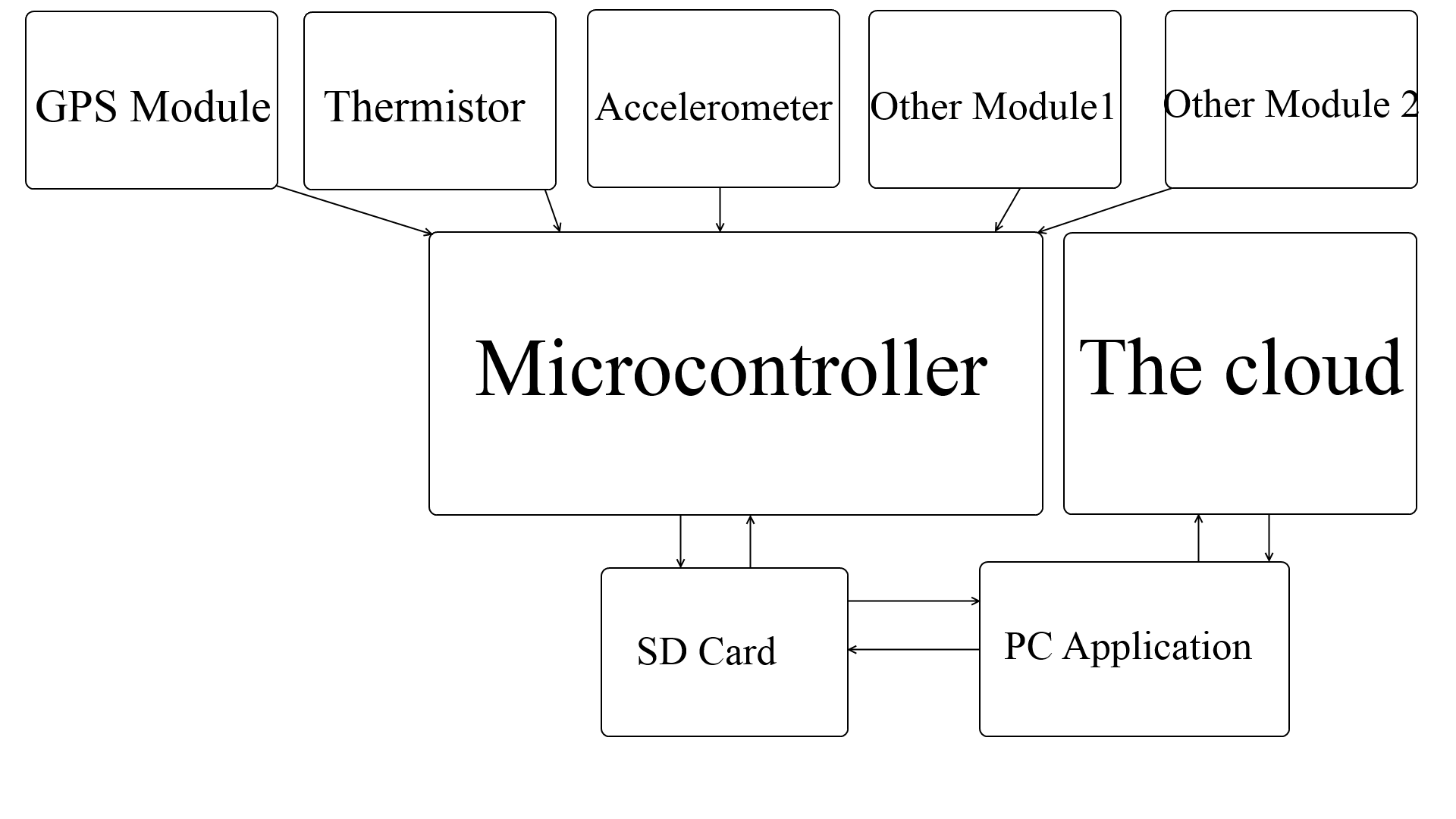


Figure 1: An overview of the system and how each section interacts.

The basis of the tracker will be a TI TIVA C Series microcontroller. This microcontroller will handle the input from the multiple sensors, such as the thermistor and accelerometer. The firmware on the microcontroller will then handle the writing of these readings to a log file stored on an SD card, which will be used to transfer the raw data gathered by the tracker to the PC application for the end user’s inspection.

The software and firmware portions of the project will overlap when the necessity for creating a logging file standard arises. The standard will be mainly restricted by the power cost associated with its being written to the SD card. The standard chosen must not create a significant impact on the battery life of the logger.

With the data transferred to the SD card in the appropriate standard, the SD card is to then be inserted into the computer, where it will be loaded into the PC application. The application will initially have the user find the location of the SD card where it will read in its data from. The application will also create and/or update a local repository of logger data stored on the cloud service. The user will then be able to select the data file they are interested in and load it into the application.

With the tracker data loaded, the user is then able to set about managing the thresholds and displaying the information they are looking for. The application has many sliders which will change the minimum and maximum thresholds for certain readings, allowing the display of a colour or image coded route displaying the areas in which the tracker exceeded these user defined thresholds. The application also allows the visualization of multiple journeys, showing a display of the temporal and special behaviour of multiple tracker units on a single map. The application will also allow the user to preprogram a route that the tracker should follow. The user will add points to a map, then the application will calculate the exact route taken and upload a small configuration file to the tracker. The tracker will then alert the deliverer in the event that the tracker drifts off course.

The design choices made for the PC application were made in order to simplify the application. The map takes up the top half of the application, making it clearly visible to the user. The setting modifiers are grouped together in a logical sense. The sliders that control the temperature thresholds are placed next to each other, as are the sliders that control the acceleration thresholds. This makes the application very intuitive to use.

It was decided that the PC application would use Google Maps as the map service. There were a few reasons behind, mainly the fact that Google Maps is a very recognisable service that people trust and that the Google Maps API is extremely functional and well documented. The first point will give the users trust in the product, while the second means that it makes the programming of the mapping functionality a lot easier compared to what would be expected from the use of an undocumented API. The PC application will be programmed using Python and JavaScript. The GUI for the application will be pure Python, calling upon JavaScript when interaction with the map is required.

Team Roles

Connor Cooper

As I have mainly taken courses regarding desktop programming, I will be undertaking the programming of the PC application for this project as this is where my strengths lie. It will be my responsibility in this project to ensure that the PC application runs without error and accurately displays the information that it is given in an intuitive manner, while also converting the raw information from our own standard into YAML for integration with the cloud service. I will mostly be interacting with the firmware programmer, as it is necessary for us to devise a standard in which the logger information can be written to the SD card with minimal battery expense. Disregarding possible cases where I may be called on to help with the firmware, I will have very little interaction with the electrical members of the group beyond keeping each other informed of our current status regarding the project.

Ge Li Ming

I have finished some electronic courses and gain knowledge from that. So I put my most effort into the hardware designing and making. With my current electronic experience, I will test and improve the current circuit and PCB diagram. And I am currently enrolled in an embedded system design course, which relates to the microcontroller design part. Due to this subject, I got a basic idea of how embedded system works, which benefits me about choosing the correct ports on the TIVA board during circuit design. Continuing with the current embedded system course, I will be better in such microcontroller field and contribute to firmware part with the programming. And I can exploit the knowledge of such two fields to better communicate with the persons who are responsible for firmware and hardware respectively.

Johnathan Wagner

The most relevant subjects that I have taken in regards to this project are CSSE1001, ELEC3400, CSSE2310, CSSE3010 and I am currently enrolled in CSSE2002. These subjects give me a basis in both analogue and digital electronics as well as python and java. However as the only member of the team who has taken CSSE3010, I will focus more on the firmware and circuit connections. When needed, I will be able to provide assistance with both the electrical and computer programming sections of this project.

Xinren Jiang

In this project, I am responsible for the power management circuit as well as PCB design and assembling. The elements and problems need to be solved including:

• An USB rechargeable lithium-polymer battery needs to be selected to provide the system acquire and record for at least 48 hours.

• Power converting and management through the device.

• Limiting the PCB size within 70 \* 51 \* 25 mm (L x W x H).

• PCB assembling with surface mounted components.

• Circuit testing and PCB finalization.

Milestones

For this project, I have set several milestones to aid in breaking down the work needed to accomplish the goals set regarding what will be delivered for the final demonstration. These are as follows.

**Internal data format fully integrated into the software**: The first main milestone I will achieve for this project will be to work with our firmware programmer in order to create a viable internal standard for our tracker’s data files. This standard must conform to several constraints, chief amongst which will be the conservation of battery life on the tracker. The main prerequisites for this milestone are that the tracker has the SD card module installed on it and the firmware supports writing to the SD card. The success of this milestone will be determined by whether or not we can move readings obtained from the tracker unit to the PC application in a usable format.

**Display of the data provided by the tracker unit:** The next milestone will be programming the PC application with the ability to logically and aesthetically display the data provided by the tracker unit. This feature will take the data provided by the tracker unit and, when combined with the user’s input into the constraints interface, display the data accordingly by overlaying it onto the map. The prerequisites for this function is that we will need to have an internal data format designed (Note: The previous milestone does not need to be completed for the implementation of this milestone. Once a format has been designed it will be possible to use falsified data as a testing point.) The success of this milestone will be determined by whether or not the PC application can accurately display the data overlaid on the map.

**Cloud compatibility:** The next milestone will be adding functionality pertaining to the cloud specification. This will consist of four areas:

1. Converting raw data from our internal format to the YAML format for uploading to the cloud,
2. Converting YAML formatted data to our internal format for use with the map display,
3. The uploading of YAML converted data to the cloud service, and
4. The downloading of YAML encoded data from the cloud and into the PC application.

The prerequisites for this milestone will be the full integration and implementation of the data format as per the first milestone. The success of this milestone will be determined by the functionality of the four aspects previous mentioned.

**User route creation:** The final milestone in this project will be the addition of a function in which the PC application can be used to plot a route that the tracker is to follow. Should the tracker go outside a predetermined margin of variance from the planned route, it should start sounding off an alarm. The user will use this additional functionality by adding points in to the PC application. The application will then call the maps API to create a route. The GPS coordinates generated from this will then be exported to the SD card in a configuration file readable by the tracker. This will require all previous milestones to be achieved before implementation of this feature can begin. Whether this milestone has successfully been met will be determined by the tracker’s ability to stay silent while on course, and sound an alarm when it veers off course.

Deliverables

For the second performance review meeting, I will be focusing on ensuring that the follow functionality is deliverable.

**Reading of the tracker data**: The software will be capable of reading raw data provided by the tracker via the SD card.

**Display of the tracker’s GPS data**: The software will be capable of making use of this data to display the route the tracker travelled.

**Display of the tracker’s temperature and acceleration data**: The software will be capable of making use of the temperature and acceleration readings by clearly indicating on the map where and when these readings have exceeded the user’s inputted constraints.

**Conversion of internal data format for cloud use**: The software will be capable of converting the raw data from our own format into a standard YAML file, for use with the cloud services.

**Upload of the converted data**: The software will be capable of uploading this YAML data to the cloud services.

**Download of cloud data**: The software will be capable of downloading YAML data from all available sources on the cloud.

**Use of cloud data**: The software will be capable of using YAML data in the same manner as our internal data format.