Department of Computer Science & Engineering
The University of Texas at Arlington

Detailed Design Specification
CSE 4317: Senior Design II
Fall 2019

Team Rocket
Rocket Board

Darion Adams
Rahasya Chandan
Alexander Pavlak
Nathan Regan
Brianca Washington

Revision History

Revision	Date	Author(s)	Description
0.1	09.10.2019	AP	document creation

CONTENTS

List of Figures

List of Tables

1 Introduction

Rocket Board is a system that collects flight data and provides tracking capabilities to model rockets. Rocket Board will enable users to gather flight information about their rocket and provide GPS tracking capabilities for rocket recovery.

Rocket Board is a sensor suite intended collect data on board a model rocket during flight. Rocket Board will be accompanied by a separate base receiver to receive flight data live. The receiver will then re- transmit the data to a companion mobile app which will provide data visualization and GPS tracking capabilities. The data collected from the sensor suite can be used to analyze flight performance and inform future launch decisions.

Rocket Board is intended for use by hobby model rocketeers as well as in STEM classrooms. Model rocketeers will gain access to a wealth of information about the performance of their rocket previously unavailable to them. Educators in STEM classrooms will gain a fun and interactive tool to help teach concepts of aerospace engineering and data science.

2 System Overview

The Rocket Board will collect and send data at regular intervals from a model rocket while in flight. The data will be collected from various sensors on the product and will be transmitted over FM radio. A USB receiver connected to an Android smart phone will receive the data and will display it on the accompanying application along with visualization of the flight path. A GPS module in the app will allow the user to find the location of the rocket for easier retrieval. Rocket Board will not give constant real time data, nor will it directly guide a user to their rocket. The whole system will involve three primary parts: the physical board connected to the model rocket, the FM receiver, and the accompanying application.

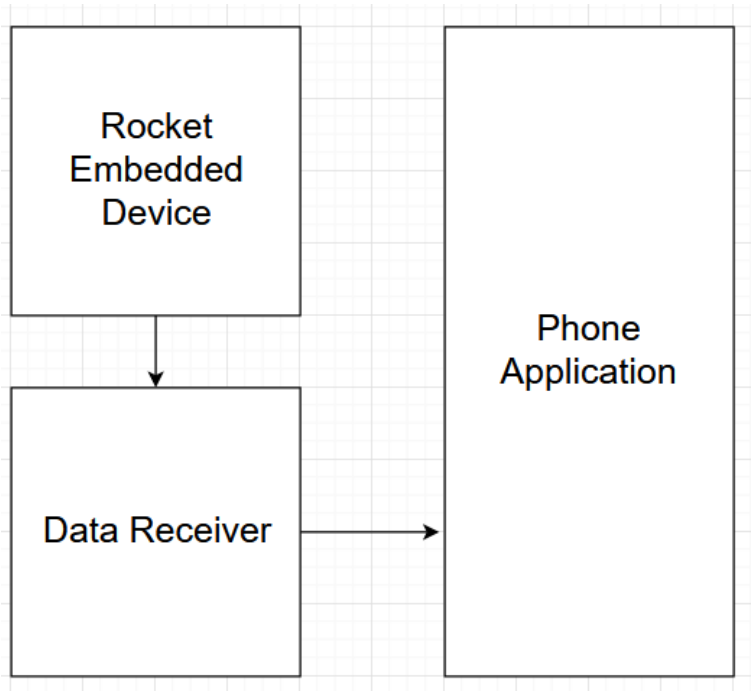

Figure 1: architectural layer diagram

The embedded device serves to collect data and transmit the packaged data to the local receiver. The inertial measurement unit (IMU), GPS tracker, and altimeter collect data and send it to a central

microcontroller. The microcontroller then validates and interprets the data from the raw voltage values to be sent to the FM transmitter. This does not include filtering of any kind, as the data will be interpreted later by the android application. The final packaged data will be sent to the FM transmitter to be received by the local receiver.

The local FM receiver exists purely to receive and transmit data from the rocket to the phone application. Data will be streamed straight from the FM receiver to the bluetooth transmitter, which contains its own protocol to be used with the Android application. No microcontroller exists between the two devices as data is streamed via I2C.

The phone application receives, interprets, and displays the data to the end user. Once the data is received through the bluetooth receiver, the raw data is stored. This data is then circularly filtered to provide a singular data measurement for each aspect at a rate of four times per second. The data is then used in two ways. In the first, the data is shown as a visual meter for properties such as pitch, roll, etc. This is to give a strict visual to the user for interpreting the rocket's current properties. This is then also used to control the rocket's 3D visualization within the application. The second aspect is GPS tracking. The live coordinates will be displayed on the application, but once the rocket's flight is completed, Google Maps' API will be used to track the rocket's end position through exporting the coordinates. The user will then be directed to the rocket for retrieval.

3 Rocket Board Subsystems

The rocket board subsystem is responsible for collecting and packaging the data. The data collected comes from an Inertial Measurement Unit (IMU), Altimeter, and a GPS Unit. Once the data is collected, the data is packaged into a single string and transmitted at a rate of 4 Hz to the FM Transmitter (described in a later subsystem).

3.1 Layer Hardware

The hardware used is the 3 chips and central microcontroller. The Altimeter Unit is the MPL3115A2, which is designed as a barometric sensor, but then calculated into the altitude. The data is collected through a Uart. The IMU is the MPU-9250, which sends the data through I2C in 2 separate registers. The setting used on the IMU is the 9-Axis output. Finally, the GPS unit is the Adafruit 790, which uses 4 satellites to receive data at a 1 Hz rate.

The central microcontroller is the ARM TM4C123GHM, which is used in a MIKROE-1595 mini development board for ease of access. The microcontroller has 5 available UARTs and the necessary processing power to transmit and collect data at far above the necessary rate.

3.2 Layer Software Dependencies

Software dependency is the existing C90 library used in the microcontroller along with the tm4c123ghm library.

3.3 Sensors

The sensors are responsible for collecting the data and sending it to the microcontroller through either a UART or I2C port. Data filtering and validation are completed at a later stage.

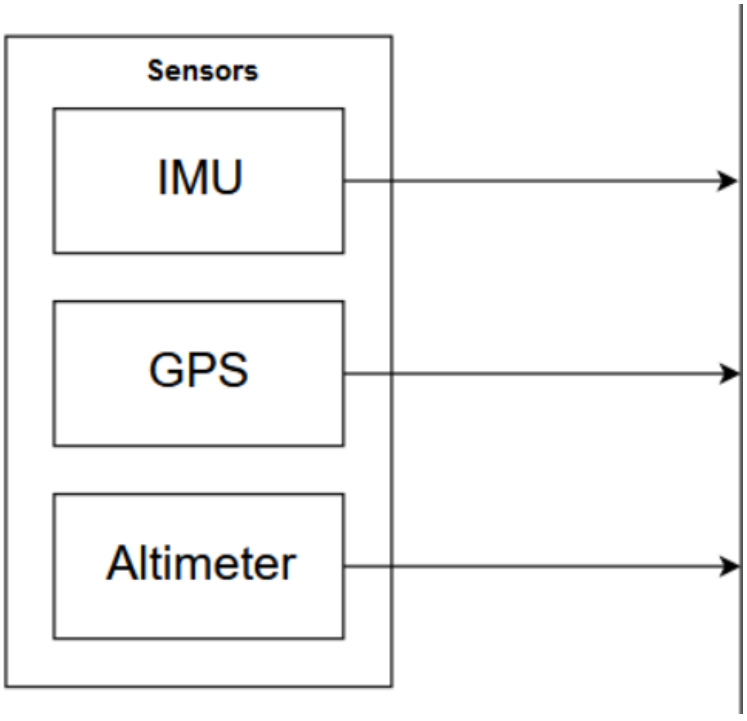

Figure 2: Rocket Subsystem

3.3.1 Subsystem Hardware

MPL3115A2 is used to collect the altitude, MPU-9250 is used to collect 9-axis measurements, and the Adafruit 790 is used to find the GPS coordinates.

3.3.2 Subsystem Data Structures

The MPL3115A2 and Adafruit 790 both use serial communication through the UART to transmit data. Data is therefore transmitted in 8-bit characters, with each having a start and stop bit.

The MPU-9250 uses I2C to transmit the bytes, so the data is stored in 2 different registers, discussed in a later subsystem.

3.4 Microcontroller Data Collection

The central microcontroller is responsible for collecting the data at regular intervals, then processing it (if necessary) and packaging the data to be sent to the FM Transmitter. The 3 sensors are connected to the UARTs of the central microcontroller.

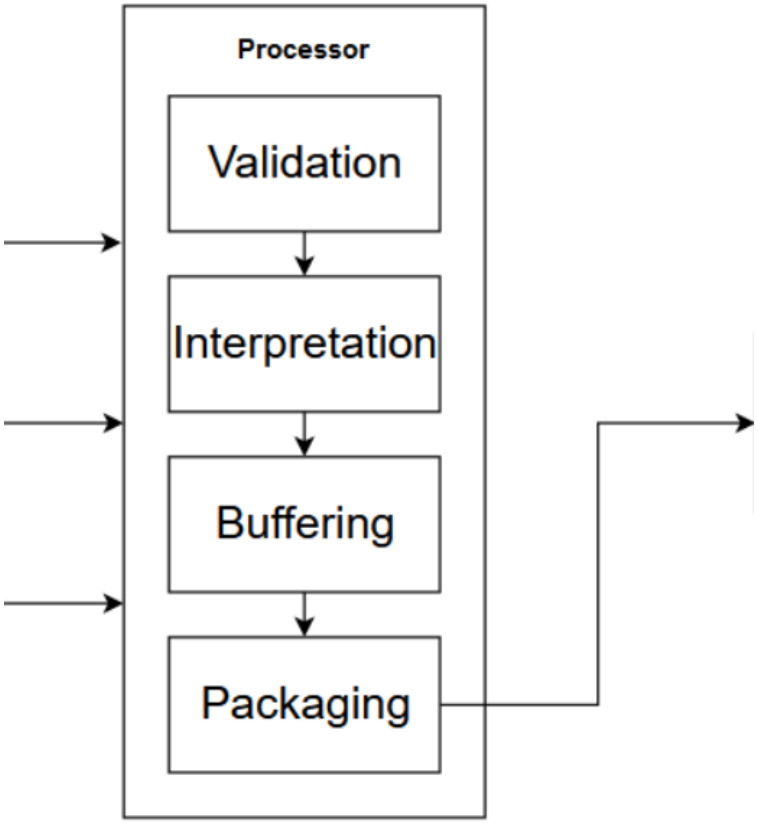

Figure 3: Rocket Subsystem

3.4.1 Subsystem Hardware

The TM4C123GH ARM microcontroller is used due its substantial processing power and excess UART units.

3.4.2 Subsystem Software Dependencies

Software dependency is the existing C90 library used in the microcontroller along with the tm4c123ghm library.

3.4.3 Subsystem Programming Languages

C (for the TM4C123 microcontroller).

3.4.4 Subsystem Data Structures

Packets are strings (continuous chars) with commas separating each data figure. The string format is as follows:

1. Time
2. Latitude
3. North/South
4. Longitude
5. East/West
6. Acceleration X
7. Acceleration Y
8. Acceleration Z
9. Temperature
10. Gyro X
11. Gyro Y
12. Gyro Z
13. Altitude

Additionally, each byte of the string is sent out in 8N1 format, with 8 bytes of data and 1 stop byte.

4 Receiver Transmitter Subsystems

The receiver and transmitter subsystem is responsible for getting the data collected from the device to the end Android application. This layer is largely focused on data packaging and transmission over a wireless medium. Both the software and hardware aspects are included in this section, as the embedded device's software aspects are entirely dependent on the hardware it resides in.

4.1 Layer Hardware

The hardware used is two FM Transmitter/Receivers, and a Bluetooth chip module. The FM chip used is the XBee Pro900 HP, which is capable of simultaneously transmitting and receiving an input signal on the 900 MHz public wave. The XBee is connected to the UART peripheral of the microcontroller and receives its serial communication. It then transmits through an antenna up to a distance of two to three miles.

The Bluetooth chip used is a RN42XVP-I/RM device. It is capable at transmitting at low distances (approximately 30 feet) at a reliable rate, which is enough for the receiver module to transmit to the Android phone. The chip uses serial transmission to receive its input.

4.2 Layer Software Dependencies

Software dependency is the existing C90 library used in the microcontroller along with the tm4c123ghm library.

4.3 FM Transmitter

The purpose of this subsystem is to send data from the rocket to the FM receiver. This includes the necessary hardware peripherals and code found on the embedded device to transmit over the 900 MHz radio wave.

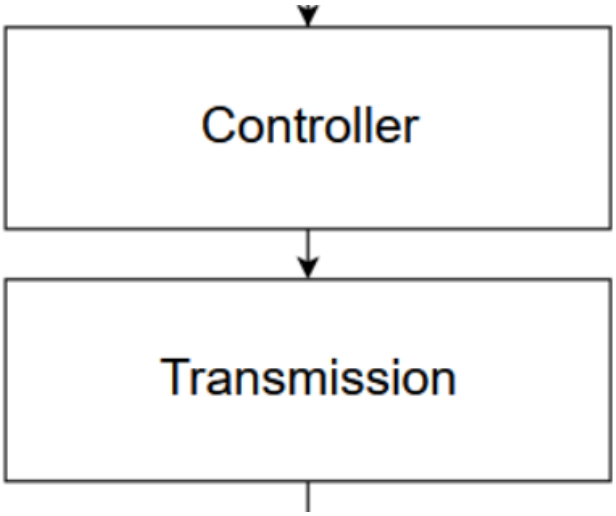

Figure 4: Transmission Subsystem

4.3.1 Subsystem Hardware

This uses the TM4C123 microcontroller (used for the prior sensors) to gather the data through the UART at regular intervals, as well as a single Xbee Pro900 HP FM Transmitter for the FM Transmission.

4.3.2 Subsystem Software Dependencies

Software dependency is the existing C90 library used in the microcontroller along with the tm4c123ghm library.

4.3.3 Subsystem Programming Languages

C (for the TM4C123 microcontroller).

4.3.4 Subsystem Data Structures

Packets are strings with commas separating each data figure. The string format is as follows:

1. Time
2. Latitude
3. North/South
4. Longitude
5. East/West
6. Acceleration X
7. Acceleration Y
8. Acceleration Z
9. Temperature
10. Gyro X
11. Gyro Y
12. Gyro Z
13. Altitude

Additionally, each byte of the string is sent out in 8N1 format, with 8 bytes of data and 1 stop byte.

4.4 FM Receiver/Bluetooth Transmitter

The receiving device, placed next to the phone, uses an FM receiver to receive the signal, then passes the output directly to the Bluetooth module. This transmits the signal to the phone for the end application to process.

4.4.1 Subsystem Hardware

The receiver uses the XBee Pro900 HP to receive the signal, which then passes the received signal to the RN42XVP-I/RM Bluetooth module.

4.4.2 Subsystem Data Structures

The packet structure is the same as the prior described in the Transmission subsystem.

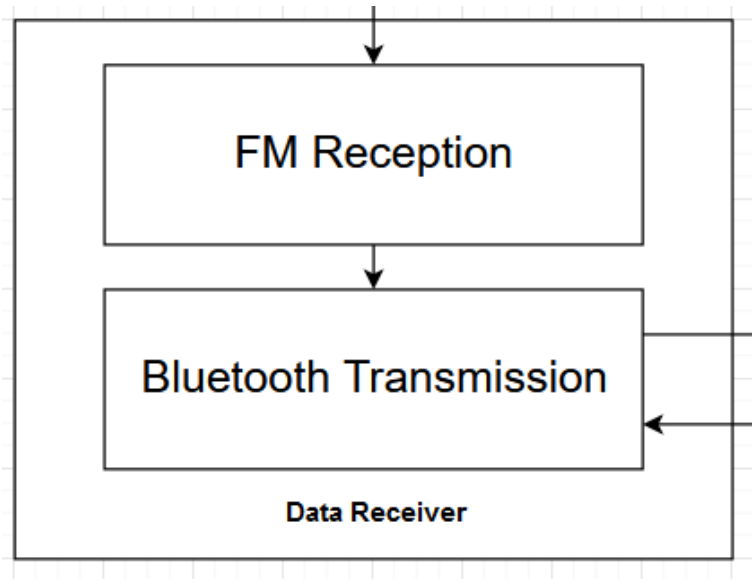

Figure 5: Receiver Subsystem

5 Android Application Subsystems

The phone application receives, interprets, and displays the data to the end user. Once the data is received through the bluetooth receiver, the raw data is stored. This data is then circularly filtered to provide a singular data measurement for each aspect at a rate of four times per second. The data is then used in two ways. In the first, the data is shown as a visual meter for properties such as pitch, roll, etc. This is to give a strict visual to the user for interpreting the rocket's current properties. This is then also used to control the rocket's 3D visualization within the application. The second aspect is GPS tracking. The live coordinates will be displayed on the application, but once the rocket's flight is completed, Google Maps' API will be used to track the rocket's end position through exporting the coordinates. The user will then be directed to the rocket for retrieval.

5.1 Layer Hardware

The Rocket Board application will run on a reasonably modern Android mobile phone. The phone must be equipped with a bluetooth radio and a GPS unit in order to function properly.

5.2 Layer Operating System

The Rocket Board application will target the Android operating system version 8. It will also make use of the Android API level 26. This is to comply with Google's guidelines for apps that are released on the Google Play Store.

5.3 Layer Software Dependencies

A description of any software dependencies (libraries, frameworks, etc) required by the layer.

5.4 Subsystem 1

Descibe at a high level the purpose and basic design of this subsystem. Is it a piece of hardware, a class, a web service, or something else? Note that each of the subsystem items below are meant to be specific to that subystem and not a repeat of anything discussed above for the overall layer.

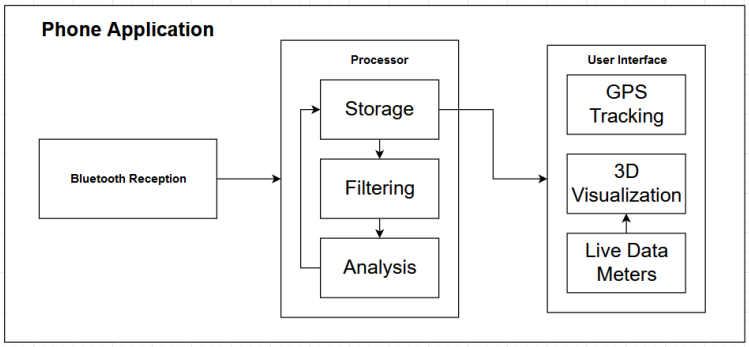

Figure 6: Phone Application Subsystem

5.4.1 Subsystem Hardware

The Rocket Board application will make use of several of the phones on board sensors. Primarily the GPS and bluetooth radio. The application will access these devices through androids provided API's. Bluetooth radio will be responsible for retrieving data from the transmitter device described in section 4 of this guide. Once received the data will be handed off to the application for further processing and display. The GPS will be used primarily for tracking. It will plot the users current position on a map and show the rockets current position. In this way the user can use the application to track down and find their rocket should it become lost.

5.4.2 Subsystem Software Dependencies

A description of any software dependencies (libraries, frameworks, design software for mechanical parts or circuits, etc) required by the subsystem.

5.4.3 Subsystem Programming Languages

All code for the Android application will be written in Java, using Android API level 26 as mentioned above. All code must compile and run in an Android Studio project.

5.4.4 Subsystem Data Structures

A description of any classes or other data structures that are worth discussing for the subsystem. For example, data being transmitted from a microcontroller to a PC via USB should be first be assembled into packets. What is the structure of the packets? Data transmitted from the rocket and received by the bluetooth receiver will come in the form of a packet. A Java class will be created for each packet received to store and process the data in the program. It will have the following fields and types.

- **GPS Data**
- Time (String)
- Latitude(Float)
- North/South(Char)
- Longitude(Float)
- East/West(Char)
- **IMU Data**

- AccellerationX(Float)
- AccelerationY(Float)
- AccelerationZ(Float)
- Temperature(Float)
- GyroX (Float)
- GyroY (Float)
- GyroZ (Float)
- **Altimiter Data**
- Altitude(Float)

  All fields will have appropriate getters and setters provided for accessing the data. The class will also provide a to string method to facilitate communicating and storing an entire packet within the app. Once a data packet is received it will be saved to local file storage for further processing.

5.4.5 SUBSYSTEM DATA PROCESSING

A description of any algorithms or processing strategies that are worth discussing for the subsystem. If you are implementing a well-known algorithm, list it. If it is something unique to this project, discuss it in greater detail.

6 APPENDIX A

Include any additional documents (CAD design, circuit schematics, etc) as an appendix as necessary.

References