TBRe Design and build an EV tuning system based on universal calibration protocol (XCP)

BEng Project Scoping and Planning Report

submitted in partial fulfillment of the requirements

for the degree of

BEng Computer Systems Engineering (with Hons)

by

Seena Shafai

(Candidate Number: 199268920)

Under the supervision of

Dr. Yunjie Gu

Department of Electrical and Electronic Engineering
University of Bath

Academic Year 2021-22
Word Count: 1782

Contents

1 Scope and Aims **1**

1.1 Introduction 1

1.2 Specification of Aims 1

2 Project Plan **2**

2.1 Breakdown of Tasks 2

2.1.1 Task 0 (Setup task) 2

2.1.2 Task 1 [Key Milestone] 2

2.1.3 Task 2 2

2.1.4 Task 3 [Key Milestone] 2

2.1.5 Task 4 2

2.1.6 Task 5 3

2.1.7 Task 6 [Key Milestone] 3

2.2 Gantt Chart 3

3 Resources Required **4**

3.1 Hardware 4

3.2 Software 4

4 Risk Analysis **5**

Bibliography **6**

1. Scope and Aims

1.1. Introduction

This project aims to design and build a platform to tune the ECU (Engine Control Unit) of electric vehicles operated by Team Bath Racing Electric (TBRe). Unlike the traditional internal combustion engine, an EV (Electric Vehicle) relies on battery power, and the power delivery can be tuned to optimise the delivery of torque from the motors, producing a more competitive vehicle.

As the name suggests, Electric Vehicles are wholly controlled by electronic means. As a result, every feature of the vehicle can be manipulated by the ECU, and this in turn can be programmed in Simulink, via CCP (CAN Calibration Protocol) [1]. CCP gives calibration and data acquisition capabilities to engineers. In this project, CCP will allow us to override the current ECU package and apply our own modifications, as per the needs of TBRe.

For development of the solution, I will be using the MUXlab4 ECU [2], the same device used by TBRe in their race vehicles. The MUXlab4 is effectively a microprocessor with configurable CAN interfaces [3]; this is what allows it to communicate with the vehicle's CAN bus. The CAN (Control Area Network) bus is a connection bus linking all of the vehicles electronics to a central control unit (the ECU)

1.2. Specification of Aims

The table below outlines the key aims of this project. Identifying these aims is crucial, as it greatly assists in creating a plan for the project and leads to breaking down each tasks individually. Furthermore, success will be measured on the completion of these aims, allowing for a quantifiable measure of success at the end of the project's timeframe.

No.	Aim	Description
1	Unlock CCP Capabilities	We can extend the capabilities of the MUXlab 4 beyond that of a standard microprocessor, by utilising the CCP capabilities to communicate with the electronic systems of the vehicle.
2.	Circumvent licensing requirements	The MUXlab4 currently requires a USB flash drive to be inserted when programming. As a secondary objective, revoking this requirement would make rapid prototyping of a solution much more efficient.
3	Override Muxlink blocks	The MUXlab4 comes with a Simulink block package called Muxlink; a pre-set range of blocks which can be programmed onto the ECU. We can modify these blocks, or create our own for greater versatility
4	Build platform for real-time tuning	With the preceding aims achieved, a Simulink environment can be created, allowing the TBRe team to easily tune vehicle parameters in real time.

Table 1: Specification of aims

2. Project Plan

2.1. Breakdown of Tasks

2.1.1 Task 0 (Setup task)

Aim: Acquire the Muxlink library any other tools provided by the manufacturer. Either from the manufacturer or from a previous use of the MUXlab4, ideally from the TBRe members.

Note that this is a prerequisite for the entire project, and that no development or further analysis can begin without this software library and driver tools to interface the physical ECU device with our computer.

2.1.2 Task 1 [Key Milestone]

Aim: Configure Simulink environment with the Muxlink library, to allow communication with the MUXlab4 MP via the CAN Calibration Protocol.

This means; plugging in the ECU device to the computer, and running Simulink with the correct software packages and drivers installed. Success in this task requires evidence that Simulink has recognised the external ECU device, and communication between Simulink and the ECU has been achieved.

mm!

2.1.3 Task 2

Aim: Interface the MUXlab4 with the calibration data logger tool via USB connection.

Before beginning to reprogram the ECU, we must first be able to read the data coming from the device and ensure that this data is correct. If there are any discrepancies, the ECU can be recalibrated for the correct responses in a controlled environment.

2.1.4 Task 3 [Key Milestone]

Aim: Reprogram the MUXlab4 with predetermined blocks provided by the Muxlink library, and test the functionality in simulations.

Using Simulink blocks we can model the operation of the ECU, and then download these blocks into the ECU hardware. We can then observe the effects of these blocks and any parameters to determine whether they are working as expected, or if any further modifications are required.

2.1.5 Task 4

Aim: Reprogram individual Muxlink blocks to achieve greater customiseability for the system, rather than relying on the basic tools provided by the manufacturer.

Some blocks provided may not work as expected, as the ECU device itself is a beta product, and the limited manufacturer's documentation is outdated. We will inspect the code behind these blocks, and make the necessary modifications to ensure that they can be used by the team in the planned interface.

Furthermore, some blocks may have limited functionality, either through intended design or through missed considerations. Regardless, we can use the same process to investigate the code that makes up these blocks, and make modifications to align with the needs of the team.

2.1.6 Task 5

Aim: We can create our own blocks for the MUXlab4 if required, to greatly extend the functionality of the ECU.

The need for this task will greatly depend on what can be achieved in the previous tasks, and may even be unnecessary, if it is determined that a suitable range of modifications can be made without the need for entirely new software blocks to be downloaded onto the ECU.

2.1.7 Task 6 [Key Milestone]

Aim: Combine the previous tasks into some form of a User Interface which will allow TBRe to observe the effects of real-time ECU modifications, and tweak any vehicle parameters as desired.

2.2. Gantt Chart

Below is a Gantt chart, collecting the above tasks into a format depicting the available time frame for each task. This is a key piece of planning information, and will be constantly referenced throughout the duration of this project to ensure that we are on track to deliver a timely solution.

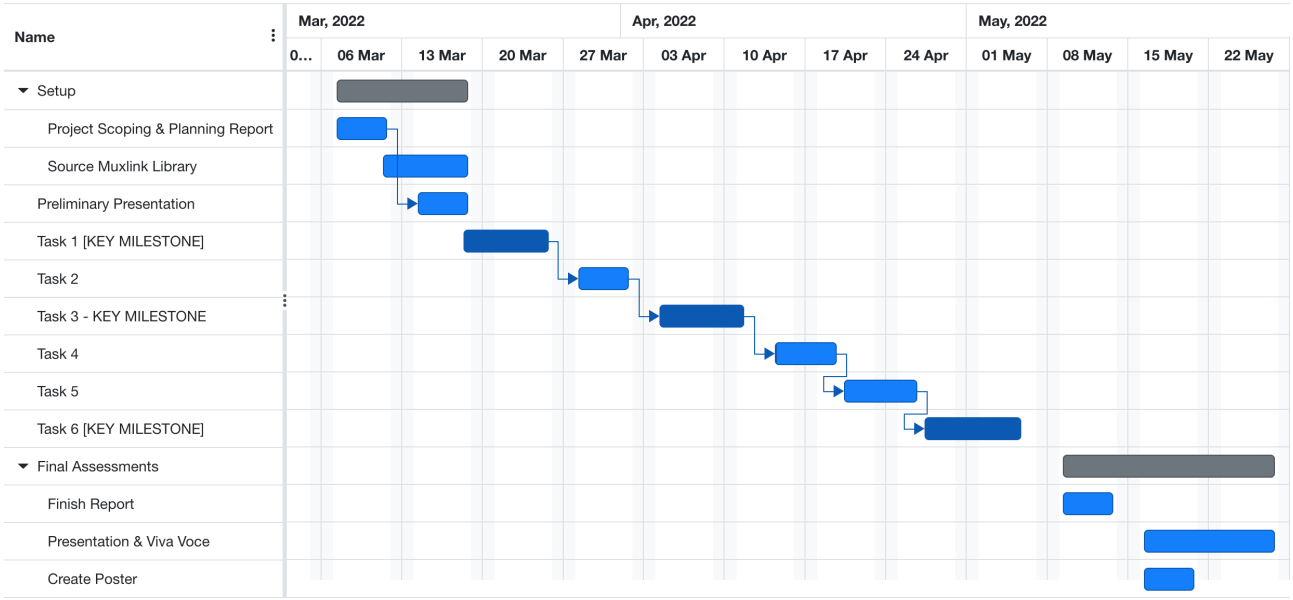

Figure 1: Gantt chart of the timings for each task in the project

3. Resources Required

3.1. Hardware

- MUXlab4 ECU
- MUXlab Licence USB Dongle

The hardware has very kindly been provided to us by Team Bath Racing Electric, as they had a spare ECU which they were not developing on at the time. However, this has imposed some constraints on the project. TBRe may request their ECU back at any time, leaving me without a microprocessor to perform my investigations on. Furthermore, as a key piece of technology for the TBRe vehicle, I will have to ensure that the ECU does not undergo any electrical damage, as this unit will eventually be required by the team for racing and development.

Therefore, the ideal solution would be to acquire a fresh MUXlab4 package from Altran (now Capgemini Engineering) which would come packaged with all the required software and drivers which would allow me to begin development immediately. There is no information as to how much this would cost, as these units were provided to TBRe in 2018 without charge, due to the sponsorial relationship between Altran and TBRe at the time. I have contacted the sponsor to gain some more insight into this relationship, and to determine if they would be interested in continuing their partnership. This would allow me to acquire a MUXlab4 solely for this project.

3.2. Software

- Full MATLAB License with Simulink
- Latest version of Muxlink library
- **Maybe**: Vector CANape

The MATLAB and Simulink environments have been supplied via a University-wide license, thus the software has been acquired from the supplier already. However, the Muxlink library is a proprietary package provided by the MUXlab4 manufacturer, and so this is a key requirement for the development of a solution. This software was provided free of charge as part of the ECU package from the manufacturer. Therefore, re-acquiring this software would be preferable, as we can keep costs to a minimum.

If, however, we are unable to source the original datalogging and real-time parameter modification tools from the manufacturer, we can purchase third-party software to accommodate this. CANape [4], by Vector Informatik GmbH is an ECU analysis tool which can be integrated into the Simulink environment. This will combine the code generation of Simulink's blocks, with the measurement and calibration tools of CANape, allowing us to create a more stable model for deployment to TBRe.

4. Risk Analysis

Risk	Impact	Mitigation
Missing software packages from Altran	This will slow down development, and increase costs as we will have to look elsewhere to source the software.	Start researching into Vector CANape and CANoe software in case third-party software is required.
Third-party software too expensive	Could result in a less stable end product, and thus more difficult to deliver to TBRe as working evidence outside of the vehicle will be limited.	The testing can all be performed in MATLAB Simulink, with the consequence of increased development times and limited testing capabilities.
Not enough time to develop all features	Product may not be adopted by TBRe as the solution will be incomplete.	Meet with TBRe to discuss priority of aims. Schedule work based on these priorities, so an acceptable solution is more likely within the given timeframe.
Electronically damaging the Microprocessor	This will halt the entire project until another ECU can be sourced, which may not be identical to those used by TBRe	Specific attention will be paid to voltage ratings of the MP before overriding any Simulink blocks for custom development
Damage to the TBRe Vehicle	The MUXlab4 is the heart of the electrical communications bus in the vehicle. If programmed incorrectly, it could lead to battery overheating which would permanently damage the vehicle.	Extensive simulations in both digital and physical environments are required. We must demonstrate the safety of the vehicle with the modified engine software.
Injury to the TBRe Driver	As the ECU translates pedal position to acceleration via a throttle curve, an incorrectly defined behaviour could result in too much/too little power being delivered at a specific moment, leading to a crash.	As above; extensive simulation is required to ensure that the ECU will send the correct signals to the vehicle, and that the vehicle will respond as expected.
MATLAB/Simulink crashing or producing corrupt files	Due to the intensive simulations in conjunction with an outdated external device, there is a non-negligible chance of file corruption, resulting in loss of work.	All work on this project will be backed up in multiple locations; the main project file will also be saved in cloud storage after each session of development on MATLAB. Furthermore, at the completion of each task, the project file will be saved onto a flash drive.

Table 2: Risk Analysis Table

References

[1] NI Engineering (Accessed: 6/3/22). CAN Calibration Protocol (CCP) Overview [Online]. Available: https://zone.ni.com/reference/en-XX/help/371602T-01/niecumchelp/ccp_overview/

[2] Mathworks (Accessed: 7/3/22). Rapid prototyping solution for embedded applications [Online]. Available: https://uk.mathworks.com/products/connections/product_detail/MUXlab4.html

[3] Capgemini Engineering (Accessed: 7/3/22). Catalogue Produits 2017 [Online]. Available: https://capgemini-engineering.com/as-content/uploads/sites/5/2017/06/france_electronics_catalogue_produit_2017.pdf

[4] Vector Informatik GmbH (Accessed: 10/3/22). ECU Calibration with CANape [Online]. Available: https://www.vector.com/gb/en/products/products-a-z/software/canape/