# Coursework – Self-driving cars

Aidan Goldie [40402670]

[1] Edinburgh Napier University, 10 Colington Road, EH10 5DT, Scotland

**Abstract.** With the recurring advancements in automated technology over the years has allowed individuals in today's world to witness the first of its kind in automated driving systems. This project aims to explore this area of software development with the creation of an automated car system to alleviate some of the human errors made while driving.

This report will cover sections such as the different procedures needed to create an automated driving system, the structure of the controller for such a system, extensions to the current specification and further development. This project was carried out using Ada SPARK on GNAT Community.

**Keywords:** Ada, SPARK, Automated Driving, Car, Software Engineering.

## 1 Introduction

The aim of this project was to design and develop an automated driving assistant for an electric car which allowed for some of the stresses and errors of regular human driving to be alleviated. A few of the more basic specifications for a project such as this one can be seen in the list below;

- The car cannot be turned on unless in parked mode
- The car cannot be driven unless there is a minimum charge in the battery
- Once in motion, the system will warn of low charge
- The speed limit cannot be exceeded, ever
- The speed of the car must be zero in order to change gear.
- If the cars sensors detect an object, then the car cannot move towards that object.
- The car must have a diagnostic mode which renders the car incapable of any other operations (maintenance mode)

Along with these basic specifications being implemented, further implementation must be undertaken in order to ensure the complexity of the system is of a viable status in order to properly alleviate a great number of human errors which can occur on the road and therefore serve as a trusted automated driving assistance system. These extensions of the minimum requirements stated above will be spoken about in greater detail in section 5.

To implement this system, the Ada programming language was used in alignment with SPARK to provide evidence and proof of the functionality of the project worked in accordance with the five different levels, stone, bronze, silver, gold and platinum.

## 2 Controller Structure

This section of the report provides a high level overview of the general structure of the project and details the purpose of each section. The implementation was carried out within three Ada packages;

- Main.adb
- Car.adb
- Car.ads

A **main.adb** file which will act as a front-end user interface page, consisting of an array of different options that the user can chose through the input of a character or number on the cars interface system. A simulation of how the cars interface will look is provided through a terminal run to help visualize the end product. This page will also provide if statements to determine things like battery level before starting the car system.

The remaining two packages, **car.adb** and **car.ads** will serve as the back end of the project and ensure all procedures can take place through the use of various if, then, else statements to determine that all pre conditions of the car are met first before a post condition can be given. The **car.adb** package is where the main bulk of the runnable code goes whereas the **car.ads** package acts as a specification file and ensures that all the pre conditions along with post conditions are met when each procedure or function is ran.

A **record** was generateds to represent a car and an instance of this named CarSystem was also created which included all the types within to ensure for fast development and overall readability within the project. The types used within this instance was that of all the properties required to ensure the the car could perform all the different kinds of functions and procedures. Use of an instance of this record was used in the event of further development where a new car would be introduced with different specifications and different functionality requirements.

## 3 Description of procedures and functions

This section will provide a more detailed look into each component utilized in order to ensure each of the minimum requirements were met.

This section details the procedure(s) carried out for each requirement within the system along with its specification detailing its pre and post conditions.

The following table, Table 3.1, shows each procedure used in the system and the grade in which it was able to achieve in regard to the SPARK implementation levels.

| **Procedure Number** | **Title** | **SPARK Grade Awarded** |
|---|---|---|
| 3.1 | EngineOn | GOLD |
| 3.2 | EngineOff | GOLD |
| 3.3 | ParkedOn | GOLD |
| 3.4 | ParkedOff | GOLD |
| 3.5 | DiagnosticOn | GOLD |
| 3.6 | DiagnosticOff | GOLD |
| 3.7 | SpeedUp | GOLD |
| 3.8 | SlowDown | GOLD |
| 3.9 | GearUp | GOLD |
| 3.10 | GearDown | GOLD |

Table 3.1. Procedure SPARK grades

### 3.1 EngineOn Procedure

This procedure was implemented to ensure that the engine of the car could be turned on by the user given that the following pre- conditions were met;

The PowerStatus along with the Maintenance mode were required to be Off, parked mode on along with the battery status being of a value greater than or equal to 10 and less than or equal to 100

Given that these preconditions were met, the post-conditions received were that the PowerStatus and parked mode were on, maintenance mode was off and the battery status was between the values of 10 and 100.

### 3.2 EngineOff

Similar to EngineOn, in regards to pre and post conditions, this procedure allowed for the user to turn off the car's engine provided that the correct pre-conditions were met. The preconditions were that the power status of the car was on along with parked mode also being on, maintenance mode off, car speed of 0mph and the car gear set to a value of 0 (neutral)

In return, the post conditions of this procedure are that the car engine will be powered off, parked mode activated and maintenance mode also off.

### 3.3 ParkedOn

ParkedOn dealt with the Parked Mode feature of this system and required the preconditions that parked mode was off, the speed of the car was 0mph and the maintenance mode was also off. The post conditions for this procedure was that of parked mode being on.

## 3.4 ParkedOff

Performing the opposite to that of ParkedOn, this procedure required the preconditions of the Parked mode being on, the car speed being set to 0mph and the maintenance mode being off. The resulting post condition of this is that the parked mode is turned off.

## 3.5 DiagnosticOn

Also known as the maintenance mode, this procedure required the following preconditions in order to turn diagnostic mode on. The car must be parked, not moving e.g. have a speed of 0mph and the maintenance mode must also be turned off. The post conditions following these  precondtions was that maintenance mode was turned on.

## 3.6 DiagnosticOff

Allowing the driver to turn the maintenance mode off within the car, this procedure required all of the same preconditions to that of the DiagnosticOn procedure. The only change to the code of this procedure compared to that of the DiagnosticOn procedure was that the maintenance mode was turned on.

## 3.7 SpeedUp

Allowing a user to increase the speed of the vehicle in 10mph intervals, this procedure required that the car was not already exceeding the set speed limit of 80mph. in the event that this 80mph speed limit was already being achieved by the vehicle then the utilization of the acceleration command, command ‘a’, was rendered useless and would not allow the user to exceed the limit, as required. Other than not exceeding the speed limit, the preconditions of this procedure were that the car engine was turned on and the parked mode was set to off. The resulting post conditions of this was that of the powerStatus of the vehicle being on and the parked mode was off along with the car gears being lower than or equal to 5, the maximum gear level.

## 3.8 SlowDown

Allowing the user to slow the vehicle down in increments of 10mph, this procedure required that the cars speed was over 10mph so that the speed would not go out of bounds and become a negative value. This procedure also required that

## 3.9 GearUp

This procedure allowed for the user to increase the gear of the vehicle in increments of 1. This procedure was important as without it, the user would be unable to increase the speed of the car. The precondition of this procedure were that the cars power was on, the speed was 0mph, the parked mode was off and the gear was less than the value 5 as if it was 5 it would increase the gear out of bounds. The post condition for this prodecure was that the car was on, parked mode was off and the gear was still less than the value of 5 for reasons stated previously.

### 3.10 GearDown

The GearDown Procedure ensured that the user would have the ability to shift the gear down. This procedure was also in increments of 1. The preconditions of this procedure were that the car power was on, speed at 0mph, parked mode on and the car gear had to be greater than the value of -1 (reverse gear).

The resuting postconditions for this procedure is that the car power remains on along with parked mode remaining off and the final post condition being that the new gear will be set to a value which is greater than or equal to -1, the reverse gear.

## 4 Proof of Consistency

The ability for the system to determine if run time errors will occur is crucial to the overall development process as, if the system could not detect these run time errors, errors could be presented when driving and therefore malfunctions and dangerous situations could arise. To guarantee absence of run time errors, the following calculation is undertaken with an example procedure for battery;

*b:= b-2 and b := 2b*

$$\begin{aligned}
&= \{b := b-2; (b := 2b\}(0 \le b \le 100)) \\
&= \{b := b-2; ((0 \le b \le 100)\left[\tfrac{2b}{b}\right]) \\
&= \{b := b-2\}(0 \le 2b \le 100) \\
&= \{b := b-2\}(0 \le b \le 50) \\
&= (0 \le b \le 50)(\left[\tfrac{b-2}{b}\right]) \\
&= (0 \le b-2 \le 50) \\
&= (2 \le b \le 52)
\end{aligned}$$

The procedure used to shift the gear up by an increment of 1 is calculated by spark In the following way, where x is equal to the value of the gear and y is equal to the value of the cars speed;

$$
\dfrac{
  \dfrac{
    \dfrac{
      \dfrac{
        \dfrac{
          \dfrac{}{1 \le x \le 4, .. \Rightarrow 1 \le x \le 4}\,Ax
        }{1 \le x \le 4, .. \Rightarrow 0 \le x \le 4}\,Arith-R
      }{1 \le x \le 4, .. \Rightarrow 1 \le x \le 5}\,Arith-R
    }{\Gamma \ge x+1}\,Arith-
    \qquad
    \dfrac{}{\Gamma \ge x+1 > y}\,Ax
  }{x\,,\,y\,,\,x \le 5\,,\,x > y-2 \Rightarrow x+1 \wedge x+1 > y}\,R\wedge
  \qquad
  \dfrac{}{x \wedge y\,,\,x \le 5\,,\,x > y-2 \Rightarrow y}\,Ax
}{
  \dfrac{
    \dfrac{
      \dfrac{
        x\,,\,y\,,\,x \le 5\,,\,x > y-2 \Rightarrow x+1 \wedge x+1 > y \wedge y
      }{x \wedge y\,,\,x \le 5\,,\,x > y-2 \Rightarrow x+1 \wedge x+1 > y \wedge y}\,L\wedge
    }{x \wedge y \wedge x \le 5\,,\,x > y-2 \Rightarrow x+1 \wedge x+1 > y \wedge y}\,L\wedge
  }{
    \dfrac{x \wedge y \wedge x \le 5 \wedge x > y-2 \Rightarrow x+1 \wedge x+1 > y \wedge y}{\Rightarrow (x \wedge y \wedge x \le 5 \wedge x > y-2) \supset (x+1 \wedge x+1 > y \wedge y)}\,R\supset
  }\,L\wedge
}\,R\wedge
$$

The code could not be copy and pasted from the latex editor and so the code to write this can be seen in the Appenix.

## 5 Extensions

Although this project and its specification requirements were completed as required, the overall system does not give much complexity in terms of automated driving experiences. It is for this reason that extensions of the minimum requirements, seen in section 1, must be implemented to ensure that a greater complexity of features is achieved within the system. The following table, Table 5.1 depicts the extensions which were made to this project;

| **Extension Number** | **Extension Name** |
|---|---|
| 1 | Car Charge Feature |
| 2 | Automatic Gear Shifting |
| 3 | Car Status Menu |
| 4 | Command Menu |
| 5 | Battery Usage Simulation |

Table 5.1. Extension

### 5.1 Car Charge Feature

This feature allows for the user to charge the car when appropriate. For example, one of the minimum requirements of this system, shown in Section 1, was that it could not be driven unless there is a minimum charge in the battery. Due to this requirement being present, along with the extension of the battery usage simulation, it was discovered that, without a procedure such as this one, there would not be a way to recharge the battery in the event that the battery status would fall below 10% which was the minimum charge required in order to drive the car.

This procedure requires the following pre conditions;

The car power must be off, parked mode on, car speed should be 0mph and the Boolean result of if the car is at a charging station should be set to Yes.

The resulting post condition of this procedure should be that of the car power status being off, parked on, and the car speed should remain at 0mph.

### 5.2 Automatic Gear Shifting

This feature was implemented and later redacted from the system after the realization that the speed of the car must be 0mph in order to change the gear and so having an automatic gear shifter procedure was not feasible for this system. Having brought this to light, this feature is spoken about in greater detail within the further workings and implementations section of the conclusion of this report, Section 6.

### 5.3 Car Status Menu

This menu is provided to the user upon every command that is entered. It was decided that this was a more logical approach as it allows for the user to view the status of all attributes of the car system frequently and eliminates the extra step of entering a command to view the status of the car. The car status menu holds important information about the car that the user may need to know quickly under specific circumstances and therefore was implemented to shown constantly throughout the driving process. Key attributes of this menu would be that of the battery status, speed, current gear and if diagnostic mode is on or not.

### 5.4 Command Menu

To provide further readability and usability to the system, a menu is presented to the user on first viewing of the system. This menu details all the commands in which a user can enter to achieve their specific goals. This menu is split into two separate lists, one for general procedures of the automated car system such as turning on and off parked or diagnostics mode, turning on and off the cars power, viewing the menu again or charging the car. The other list revolves around procedures used to physically drive the car and perform automated driving actions. Examples of items in this drivers assistance list is that of increasing or decreasing the speed of the car or shifting the gears up or down.

### 5.5 Battery Usage Simulation

As this project is not being run in a real time environment with actual sensors and readings, the battery of the car was simulated to decrease in status by 2% for every command entered by the user. Once a command is entered, the main.adb file is primarily in a loop to keep a constant flow of information retrieval available to the user and so each time the loop is ran e.g. a command is entered, the battery status is lowered by 2%.

# 6 Conclusion

To conclude this project, the minimum specification requirements, detailed in Section 1, were met for this system. These requirements were implemented to ensure a safer driving environment and experience could be achieved by the system and to alleviate the stresses and potential errors of regular human interaction driving. The functionality of these requirements was validated and proved through the user of Ada SPARK as the code was able to be tested against the five different levels and as a result of this was able to achieve a functionality level of GOLD.

Although the requirements shown in section 1 are of beneficial use to an automated car system, they are not enough to achieve a fully automated car driving system and so a number of extensions were implemented throughout the system to further aid in the overall complexity, safety and usability of the ‘self-driving’ car system. These extensions included the addition of a car status feature, detailing all the current values of the car in real time of when the command was entered, addition of a ‘charge car’ command that, when the correct pre-conditions were met, allowed the user to charge their car in the event that it was low on charge due to extensive driving and command entering. The final extension gave the user the ease of mind when it came to gear shifting. Although the user was able to manually enter the command to change gears when they so desired, an extension of this was also implemented to ensure that, in the event that the user wanted to increase their speed to that of which would exceed the speed available for the currently set gear, the system would automatically shift the gear up or down depending on the current speed of the car, therefore alleviating the need to change gear every time the user would like to change speeds. The utilization of these extensions allowed for a higher complexity of the system to be achieved and therefore can argue a greater use case for a system such as this one in the modern world of driving.

## 6.1 Further work

In regard to the automated driving system and the conclusions drawn above, further workings and implementations must be applied to ensure that the system can gain complete or near to complete automation control and allow for minimal user input. Appropriate examples of further work could be the addition of a graphical user interface to aid in the readability of the system and different sections such as general commands and driving commands can be separated to ensure that the driving experience is as smooth as possible with only the most important information being displayed to the user, dialog boxes would be included to further highlight warning signals and messages to the user. Furthermore, advancing the sensors within the system to determine the speed and direction of objects could be vital to the improvement of such a system as it would then be able to detect objects coming toward the vehicle from any direction and determine if a collision may occur and act accordingly based on this information e.g. speed up or slow down to avoid objects. An addition of real time traffic knowledge could also be implemented to determine the fastest route to the required destination and the system can suggest these routes to the user to then decide if they'd like to redirect their journey. The requirement that the speed of the car must be at 0mph before a gear change can occur is illogical and so the removal of this feature should be pursued. Furthermore, if this feature of the system was removed, an automatic gear shifting procedure could be put in place depending on the speed in which the car is going which in turn will alleviate the driver from changing gears as they increase or decrease in speed.

# 7 Appendix

## 7.1 Run Time Errors Code

```
\{b := b-2; b := 2b\} (0 \leq b \leq 100)
\\
\\
$= $ \{b := b-2; (b := 2b\} (0 \leq b \leq 100))
\\
$= $ \{b := b-2; ((0 \leq b \leq 100) \left[\frac{2b}{b}\right])
%\\
$= $ \{b := b-2\} (0 \leq 2b \leq 100)
\\
$= $ \{b := b-2\} (0 \leq b \leq 50)
\\
$= $ (0 \leq b \leq 50) (\left[\frac{b-2}{b}\right])
\\
$= $ (0 \leq b - 2 \leq 50)
\\
$= $ (2 \leq b \leq 52)
```

## 7.2 Sequent Calculus

```
\infer[R\hook]{\Rightarrow(x \wedge y \wedge x \leq 5 \wedge x > y-2) \hook (x+1
\wedge x+1 > y \wedge y)}
{\infer[L\wedge]{x \wedge y \wedge x \leq 5 \wedge x > y-2 \Rightarrow x+1 \wedge
x+1 > y \wedge y}
{\infer[L\wedge]{x \wedge y \wedge x \leq 5 $ , $ x > y-2 \Rightarrow x+1 \wedge x+1
> y \wedge y}
{\infer[L\wedge]{x \wedge y $ , $ x \leq 5 $ , $ x > y-2 \Rightarrow x+1 \wedge x+1 >
y \wedge y}
{\infer[R\wedge]{x $ , $ y $ , $ x \leq 5 $ , $ x > y-2 \Rightarrow x+1 \wedge x+1 > y
\wedge y}
{\infer[R\wedge]{x $ , $ y $ , $ x \leq 5 $ , $ x > y-2 \Rightarrow x+1 \wedge x+1 > y}
{\infer[Arith-]{\displaystyle \Gamma \geq x+1}{\infer[Arith-R]{1 \leq x \leq 4,..
\Rightarrow 1 \leq x \leq 5}{\infer[Arith-R]{1 \leq x \leq 4,.. \Rightarrow 0 \leq x \leq
4}{\infer[Ax]{1 \leq x \leq 4,.. \Rightarrow 1 \leq x \leq 4}{}}}}
& \infer[Ax]{\displaystyle \Gamma \geq x+1 > y}{}{}{}{}}
& \infer[Ax]{x \wedge y $ , $ x \leq 5 $ , $ x > y-2 \Rightarrow y}{}{}}{}}
{}{}}}}
```