## CS723 Assignment 2 Report

Designing a Cruise Control System in Esterel

Sylvain Bechet, Mark Yep (Group 14)

### Abstract

We present the design of a cruise control system in Esterel. Based on a set of requirements, we first developed a set of functional specifications for a cruise control system. We then implemented these specifications using the synchronous programming language Esterel. This allowed us to develop an executable reactive program that fulfils the given requirements for a cruise control system. We then tested our solution with a variety of inputs to ensure it was working as intended.

### Introduction

This solution is based on an automobile cruise controller system with given specifications. The purpose of this system is to maintain the car at a constant speed without the user pressing the accelerator or brake, while incorporating basic security features. As such, the system allows the car to maintain a constant speed as long as it does not exceed a specific threshold or falls under another one. Furthermore, in order to help the system stay stable, the system does not operate while either the brake or the accelerator pedals are pressed. This solution is implemented using the Esterel language as it offers many features which facilitate the development of the system, the prominent one being its ease of implementing concurrency. Esterel also provides a deterministic solution which ensures that we do not get unexpected output for a tested input sequence.

### Specifications

The overall behaviour of the cruise controller is described using the context diagram shown in Figure 1. The system takes inputs from the driver through pedals, buttons and the car’s current speed. After processing the inputs, the cruise controller then outputs three signals to other parts of the cars.

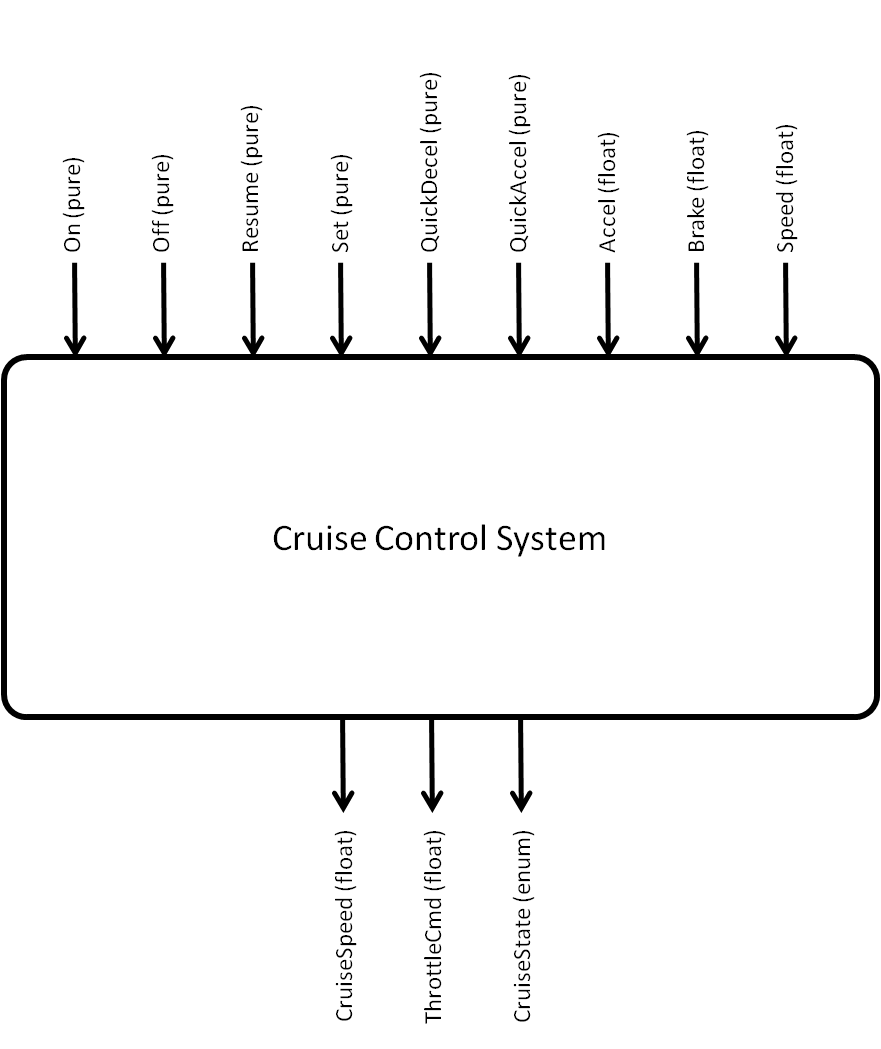


Figure 1: Overall Cruise Controller Interface

The inputs to the cruise controller consist of the acceleration pedal, the brake pedal, six different buttons – Off, On, Resume, Set, QuickAccel and QuickDecel, and the current speed of the car. The Off and On buttons are used to turn the system off and on accordingly while the Set button takes the current speed of the car and sets it as the desired cruising speed. As the cruise controller is disabled after the brake is pressed, the Resume button is used to re-enable it without having to restart the system. This also conserves the previously saved cruise speed. While the cruise controller is turned on, the driver has the possibility of pressing the QuickAccel or QuickDecel buttons. These buttons increase and decrease the current cruise speed by a set value accordingly. If either the accelerator pedal or brake pedals are sufficiently pressed, they affect the state of operation of the system. The brake pedal will change the system to a standby state and the accelerator pedal will change the system to a disable state. These states are held until both pedals are released. Finally, the cars current speed is sampled when setting the cruising speed. This is also used to ensure that the system does not operate while outside a range of reasonable speeds. This implies that the cruise controller will be disabled as long as the car is above or below the threshold speed values.

The three signals which are outputs of the cruise controller – the cruise speed, the throttle command and the cruise state – are used by other parts of the car in order to apply the functions of the cruise controller. The cruise speed is used to give feedback to the driver about the currently set speed of the cruise controller. The throttle command is related to the engine while the cruise state informs the driver about which of the four states the cruise controller is in: off, on, disabled or standing by.

The top-level context diagram in Figure 1 can be further detailed into the lower-level context diagram shown in Figure 2 to highlight how each module in the system relates with system inputs, outputs, and other modules. Each of the three modules focuses on a single output, with some dependent on other modules while running concurrently. Firstly, the cruise control system module – the only independent module – processes the inputs On, Off, Resume, Accel, Brake, and Speed to determine the state in which the cruise controller should be in. Then the cruise speed management module uses some of the information processed in the cruise system control as well as the inputs Set, Resume, QuickAccel, and QuickDecel to determine the value of the cruising speed. Finally the car driving control module uses information processed in both previously described modules to calculate the value to pass to the throttle command.

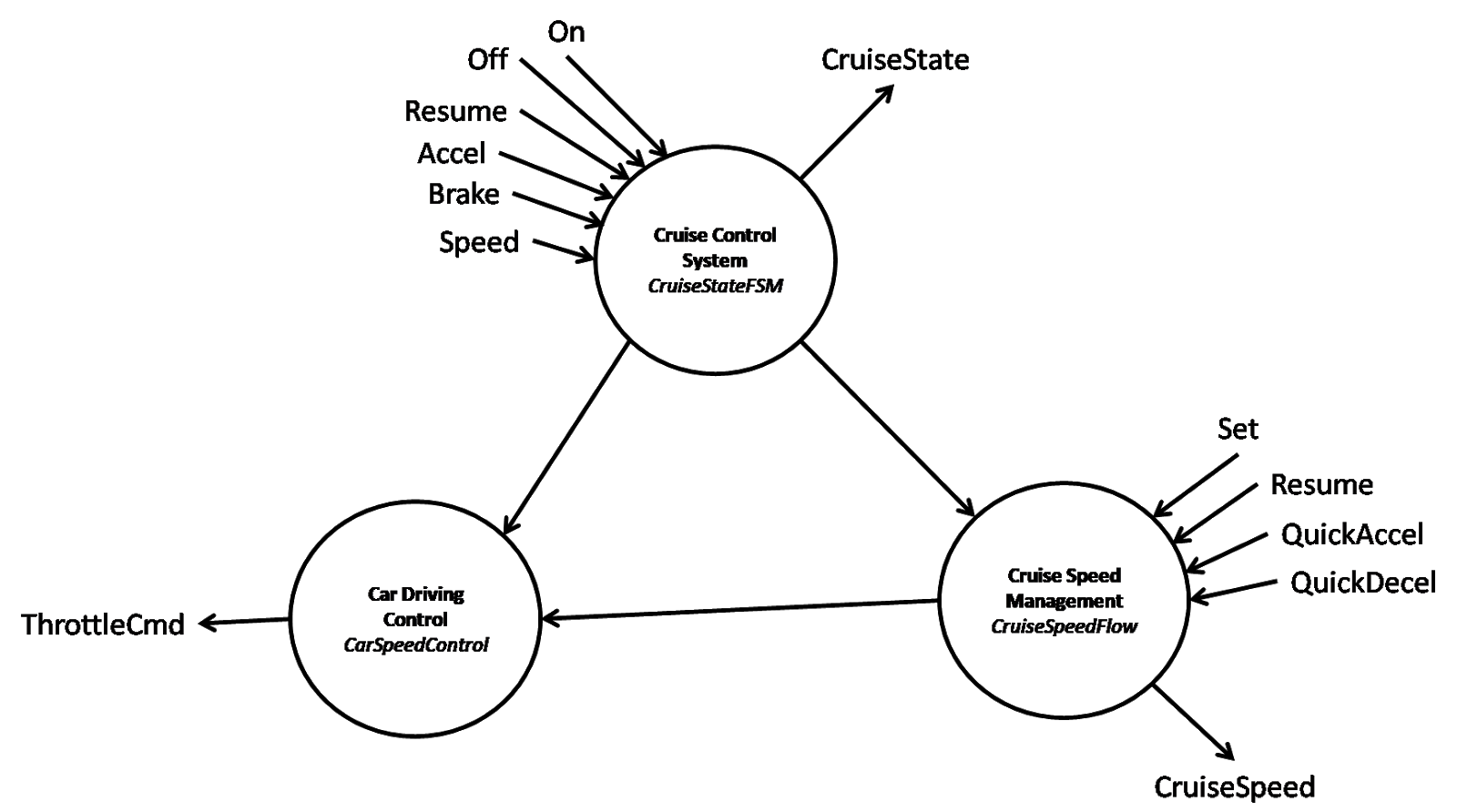


Figure 2: Overall System Relation Diagram

The cruise control system module operates according to a Mealy finite state machine (FSM) detailed in Figure 3 in order to determine the current state the system should be in. The cruise speed management module – detailed in Figure 4 – operates more like a flow chart, processing information through a series of conditional statements in order to determine the cruise speed. As for the car driving control module, it behaves more like a function dependent on the presence of a few conditions.

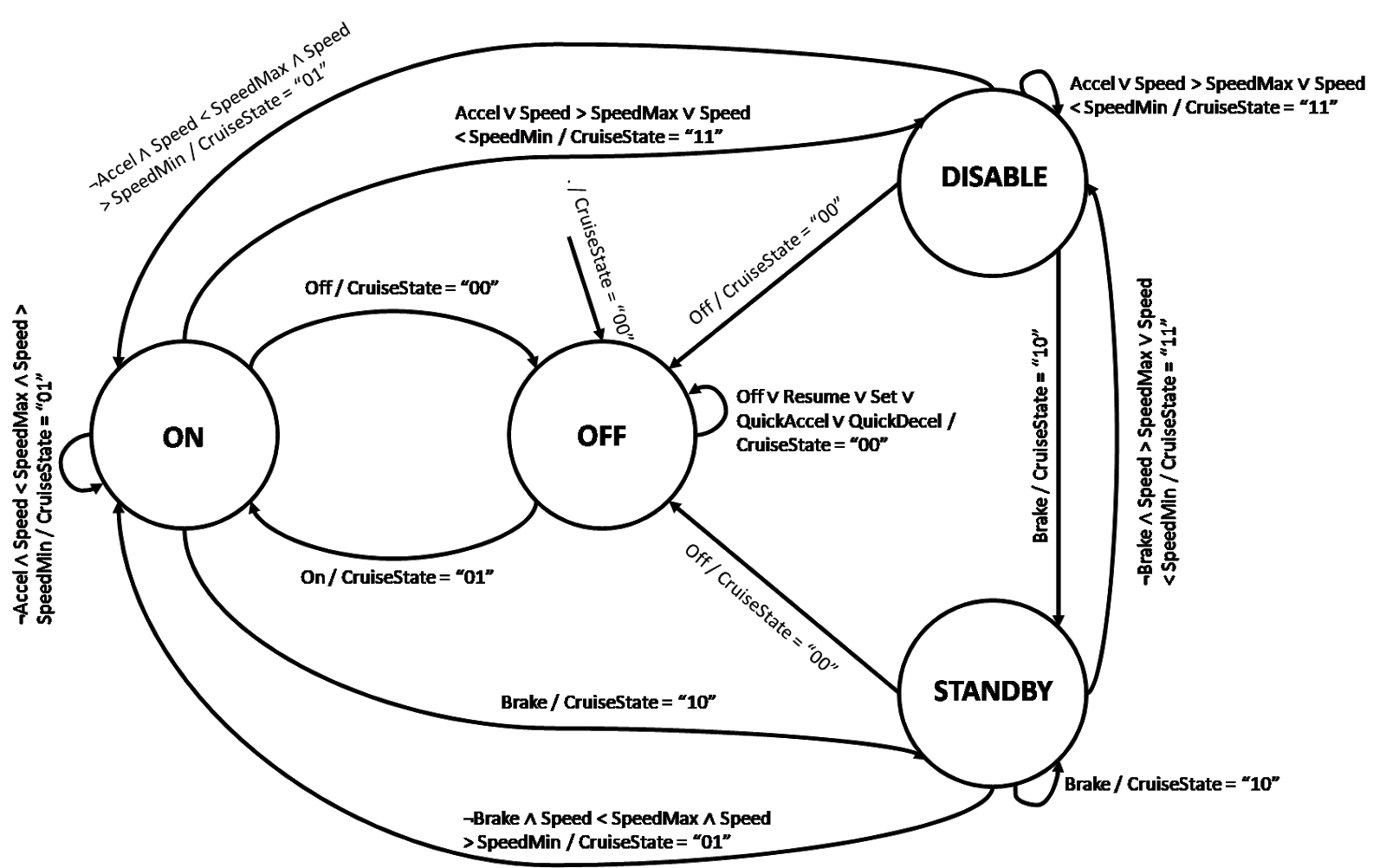


Figure 3: FSM of Cruise State Controller

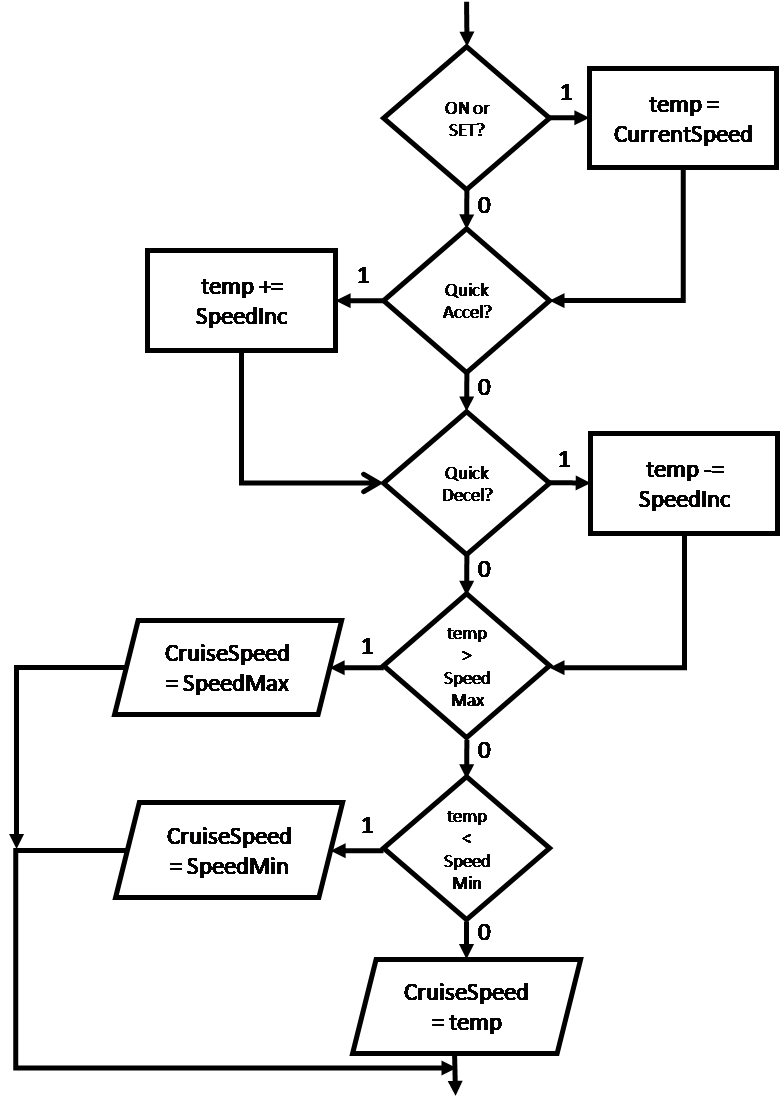


Figure 4: Flowchart of Cruise Speed Controller

### Esterel design

The cruise controller system has been designed based on the specifications mentioned in the previous section using the Esterel language. Three modules are used to define the behaviour of the system – the CruiseStateFSM module, the CruiseSpeedFlow module, and the CarSpeedControl module. All of the modules are encapsulated under the top-level module called CruiseController.

The cruise control system or CruiseStateFSM executes an FSM that handles the inputs On, Off, Resume, Accel, Brake and Speed to determine the state in which the cruise controller should be in. As described in Figure 3, the system can either be off, on, disabled or standing by. In order to minimise unexpected behaviour, each state was given a priority relative to the other ones: the off state has the highest priority, followed by the standby state, the disable state, and then the on state. The reason behind this decision was that turning on the system should be an immediate action. Then we decided that standby state should have a higher priority than the disable state. This is because the standby state is dependent on the brake being pressed while the disable state is dependent on the accelerator being pressed. As a result, we decided that any actions relating to the brakes should have a higher priority than any actions relating to the accelerator. Finally the on state was given the lowest priority as the system should only be active if it is safe for it to do so.

The cruise speed management or CruiseSpeedFlow module is detailed in Figure 4. It follows a flow chart design, processing information through a series of conditional statements in order to determine the cruise speed. It uses some of the information processed as well as the inputs Set, Resume, QuickAccel and QuickDecel to determine the value of the cruising speed. This module only runs when the cruise control system model is not in the off state.

The car driving control or CarSpeedControl module is a basic module which is used to calculate the value to pass to the throttle command. It uses information from the cruise control system module and cruise speed management module to select whether the throttle command output is dependent on the accelerator pedal or on the cruise controller value. This module uses the external saturateThrottle and regulateThrottle functions defined in the corresponding C code. This defines an integral based control system to regulate the car throttle.

The cruise controller module or CruiseController is the top-level module which runs the CruiseStateFSM, CruiseSpeedFlow and CarSpeedControl modules concurrently, while also linking all the signals correctly.

### Conclusion

This report details the implementation of a cruise controller with given specifications. These specifications are detailed and shown how they have shaped the design of the cruise control system in Esterel, in order to give an understanding of the development process undergone to create it.