**System Requirements**

***SteerTurnIllum***



|  |  |  |
| --- | --- | --- |
| Author: |  | Nicolae-Bogdan Bacrău |
| Version: |  | 1.0 |

Index

1 Glossary 3

2 Introduction 4

2.1 Purpose of the Document 4

2.2 Overview 4

3 Methodology 5

3.1 Steering 5

3.2 Debouncing 8

3.3 Moving Average 9

3.4 Fading 11

4 System Requirements 13

4.1 General 13

4.2 Vehicle Management 14

4.3 Steering 17

4.4 Turning 18

4.5 Illumination 20

5 Information about this Document 22

5.1 Copyright 22

5.2 Version Index 22

# Glossary

This section contains a glossary of all the important terms and acronyms used inside the document.

|  |  |
| --- | --- |
| **Term / Acronym** | **Description** |
| ECU | Electronic Control Unit |
| uC | Microcontroller |
| ADC | Analog Digital Converter |
| PWM | Pulse Width Modulation |
| EMC | Electromagnetic Compatibility |

Table 1 - Glossary.

# Introduction

## Purpose of the Document

The purpose of the document is to define the system requirements of the ***SteerTurnIllum*** embedded academy project.

## Overview

***SteerTurnIllum*** is an abbreviation of **Steering**, **Turning** and **Illumination**, which represent the functional features of the emulated vehicle system:

1. **Steering**:

* Steering control of the vehicle through movements of a joystick, emulating the wheel, and indicated by servomotor angles.
* Horn control of the vehicle through pressing of a dedicated horn button and indicated by a buzzer activation.

1. **Turning**: control of turn signals (left, right and hazard signaling) through pressing three turn indication buttons and indicated on four turn indication LEDs through blinking; the position of the wheel can also stop the left or right turn indications.
2. **Illumination**: control of the vehicle board illumination through a potentiometer emulating the outer luminance and indicated on two illumination fading LEDs.

The system presents one extra feature, named **Vehicle Management**, which enables the user to switch between a normal state, in which the three functional states are active, and a configuration state in which several runtime configuration parameters can be set up which affect the behavior of the steering, turning and illumination features.

# Methodology

## Steering

In the cars, the most conventional steering arrangement is to turn the front wheels using a hand operated steering wheel. Depending on the geometry of the wheels while steering, there are two possible steering control types:

* **Parallel steering geometry**: the wheels turn an identical angle when steering.
* **Ackermann steering geometry**: the inside wheel turns a greater angle than the outside wheel to follow the smaller radius that the inside wheel takes on a turn.

The two steering geometries are presented in **Figure 1**.

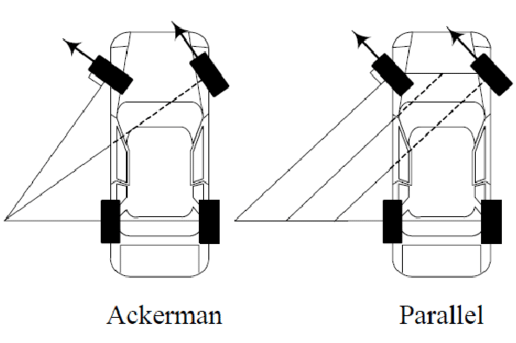


Figure 1 - Parallel vs Ackermann steering geometries.

While the parallel steering geometry is straightforward (the angles of the two wheels are identical), the Ackermann steering geometry defines a mathematical formula between the two angles, named: inner angle, 𝛿𝑖 (left angle when turning left, right angle when turning right) and the outer angle, 𝛿o (right angle when turning left, left angle when turning right). The Ackermann steering geometry is presented in **Figure 2**.

The Ackermann steering mathematical equation is:

Let *w* be half the size of *l*. In this case the equation of 𝛿o depending on 𝛿𝑖 is:

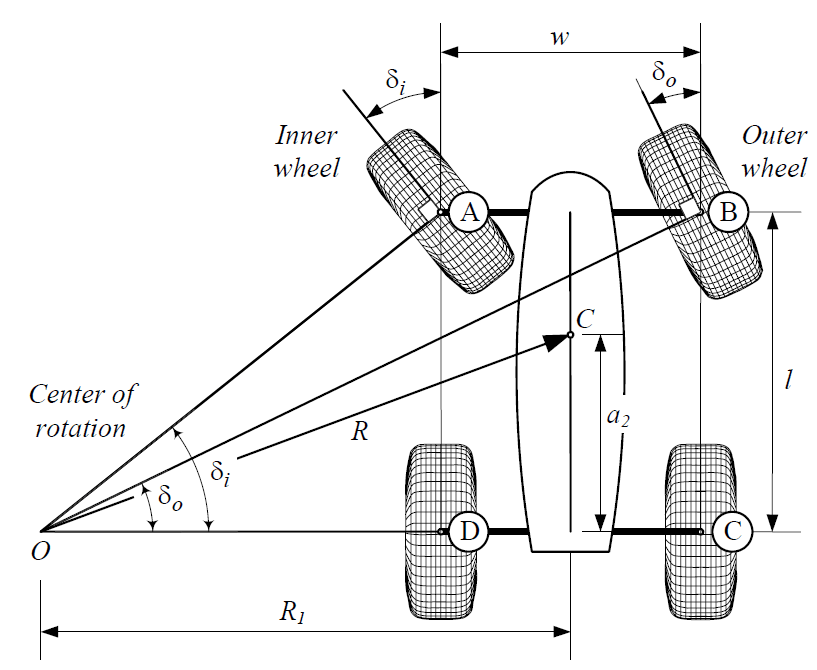


Figure 2 - Ackermann steering geometry.

A plot of outer wheel angles depending on inner wheel values is presented in **Figure 3**.

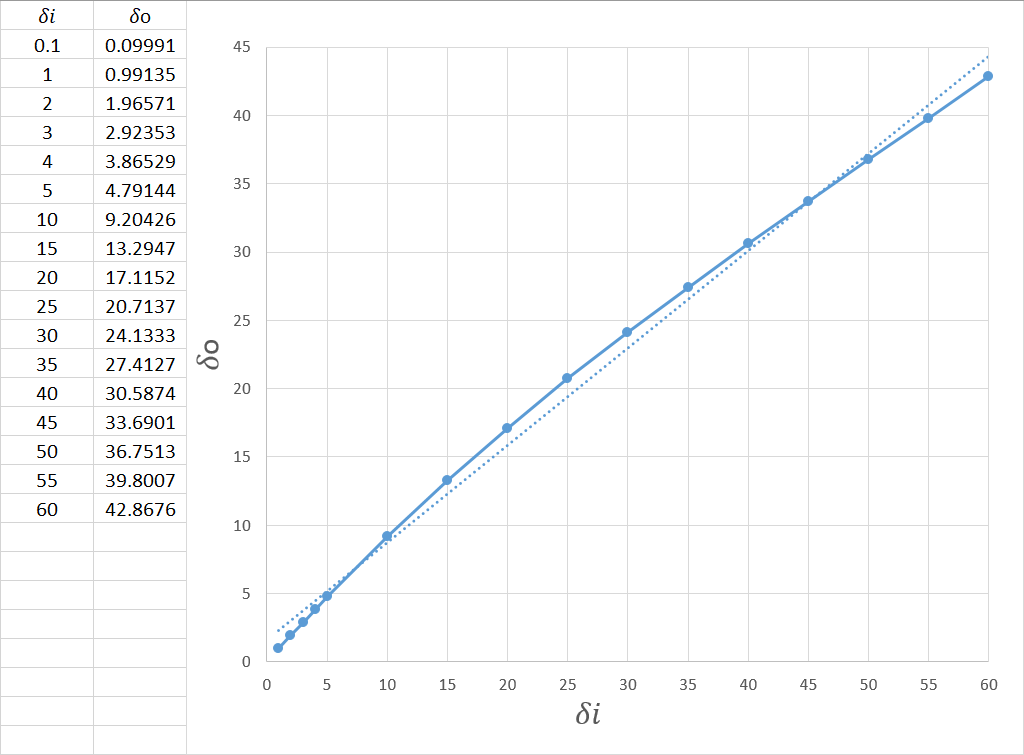


Figure 3 - Ackermann outer angles depending on inner angles.

The equation can be approximated through 4 linear equations, as shown in **Figure 4**.

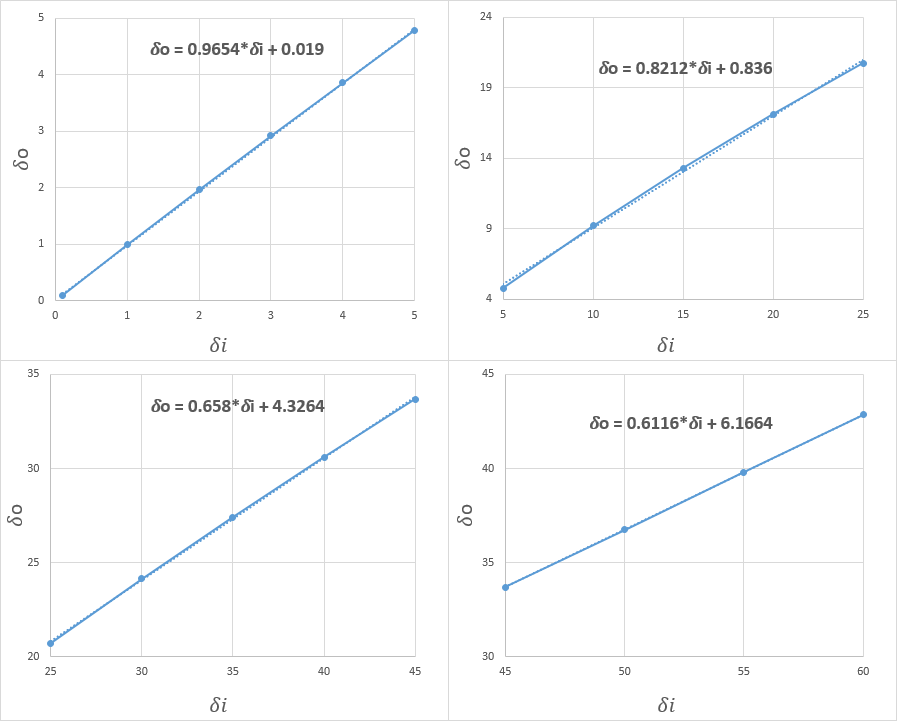


Figure 4 - Ackermann linear functions in the (0..60] degrees interval.

The 4 linear equations can be applied for inner angles in the (0..60] degrees interval. The Ackermann equation does not make sense for an inner angle of 0 degrees (*ctg(0)* is undefined). Therefore, when the inner angle is 0 the outer angle is also 0.

For more details see ./Resources/AqRe\_Graphs.xlsx → Ackermann.

## Debouncing

In order to avoid frequent switching of the output values because of mechanical contact oscillation at button presses and releases, or because of EMC influence, the input values are debounced. This consists of considering a larger time interval than the contact maximum oscillation period or the EMC influence period in which the input values must be stable. By doing so, the probability of considering a wrong button state is greatly reduced.

Debouncing is defined by two values: the debounce window, representing the time period in which an input must be stable, and the debounce depth defining the number of times an input is sampled in the debounce window.

An example showing the result of a debounce with a depth of 4 on a button input signal that is affected by noise is illustrated in **Figure 5.**



Figure 5 - Debounce of a button input affected by noise.

A debounce depth of *N* means a debounce window equal to the sum of *N-1* acquisition time window periodicities. In the given example, if the acquisition time would be *5 ms*, then the debounce depth of 4 would mean a debounce window of *(4-1)\*5 = 15 ms*.

Each mechanical button component has a maximum bouncing period of the contacts that is defined by the manufacturer / supplier of the component. It must be ensured that the chosen debounce time is larger than the defined bouncing period.

A measure for reducing the probability of erroneous button press events is to increase the debounce time of the buttons.

**Observation**: the debounce time greatly affects the response time, therefore it should be kept as low as possible (usually below 20 ms).

## Moving Average

A moving average is a simple Low Pass FIR (Finite Impulse Response) filter commonly used for regulating an array of sampled data. It takes a number of samples of input at a time, defining the *Depth*, and calculates the average of those samples, producing a single output value. As the length of the filter increases, the smoothness of the output increases, whereas the sharp modulations in the data are made increasingly blunt.

A moving average filter is described by the following formula:

* *tx* – a given point in time (timestamp);
* *D* – the filter depth;
* *S(tx)* – the sampled input signal;
* *F(tx)* – the filtered output.

An example of an input signal (*S(tx)*) and the output of a moving average filter (*F(tx)*) with a depth of 3 (*D = 3*) is presented in **Figure 6**.



Figure 6 - Moving average example.

For the given example, the filtered output in *t5* is:

An example of filtering the output of a noisy ADC with a depth of 5 is presented in **Figure 7**.

Figure 7 - Moving average on a noisy ADC output.

The filtering operation adds a delay between the input values and the output values. For example, applying a moving average with *D = 10* over the (*0, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10*) input, results in the (*0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10*) output.

**Observation**: the depth of the moving average greatly affects the response time, therefore it should be kept as low as possible.

For more details see ./Resources/AqRe\_Graphs.xlsx → Moving average.

## Fading

Sudden changes in the illumination levels are very easily observable by the human eye and can distract the driver’s attention. In order to diminish this effect, a slow transition between two illumination levels is needed. The transition, also called fading, consists of a series of intermediate illumination levels over a long period of time.

The illumination level is usually controlled via PWM signals applied on illumination LEDs. The actual illumination levels of the LEDs are not linearly correlated to the PWM duty cycle percentages, the changes in the illumination levels being more sensitive when the duty cycles are low. Therefore, in order to obtain a more linear transition of illumination levels, the used PWM duty cycles shall follow a non-linear equation. A good non-linear equation that satisfies a linear transition of illumination levels is:

*FadingLevel(x)* takes values in the [0, 1]interval, defining PWM duty cycle percentages in the [0, 100]% interval. The graph of *FadingLevel(x)*, using PWM duty cycle percentages, is presented in **Figure 8**.

Figure 8 - Fading levels graph.

Two examples of PWM duty cycle fading using the given equation are presented in **Figure 9** and **Figure 10.**

Figure 9 - Fading PWM duty cycles from 0% to 30% in 150 ms with an update rate of 10 ms.

Figure 10 - Fading PWM duty cycles from 100% to 50% in 100 ms with an update rate of 10 ms.

For more details see ./Resources/AqRe\_Graphs.xlsx → Fading.

# System Requirements

## General

1. The system shall contain 3 x buttons for controlling the turn signals and 1 x button for controlling the horn activation, named and arranged in the following left to right order: BTN\_LEFT, BTN\_RIGHT, BTN\_HAZARD, BTN\_HORN.
2. The system shall contain 1 x joystick for controlling the steering angle, named JOY\_STEER.
3. The JOY\_STEER joystick shall be positioned such that left-right movements shall influence the steering control, left-most position meaning the 0% position and the right-most position meaning the 100% position. Top-bottom movements shall have no effect on the steering control. By default, the stick shall sit in the 50% position.
4. The system shall contain 1 x potentiometer for controlling the luminance, named POT\_LUMINANCE.
5. The system shall contain 1 x buzzer for indicating the activation of the horn: BUZZ\_HORN.
6. The system shall contain 4 x LEDs for indicating the turn signals, arranged in a square shape and named in the following clockwise order, starting from top-left: LED\_TURN\_FRONT\_LEFT, LED\_TURN\_FRONT\_RIGHT, LED\_TURN\_BACK\_RIGHT, LED\_TURN\_BACK\_LEFT.
7. The system shall contain 2 x red LEDs for indicating the board illumination, named and arranged in the following left to right order: LED\_ILLUM\_NIGHT, LED\_ILLUM\_DAY.
8. LED\_ILLUM\_NIGHT and LED\_ILLUM\_DAY shall be controlled via PWM channels with a frequency of 1600 Hz +- 10% and a duty cycle resolution of 0.01%.
9. The system shall contain 1 x dual 7 segments display for indicating the configuration states, named DISP7SEG.
10. The system shall contain 2 x servomotors for indicating the steering of the wheels, named and arranged in the following left to right order: SERVO\_TURN\_LEFT, SERVO\_TURN\_RIGHT.
11. All buttons shall be debounced in a 10 ms debounce window with a minimum depth of 3 (for more details see Chapter 3.2).
12. All analog channels shall be filtered using a moving average with a depth size of 5 over a 10 ms period (for more details see Chapter 3.3).

## Vehicle Management

1. The system shall implement the state machine illustrated in **Figure 11.**

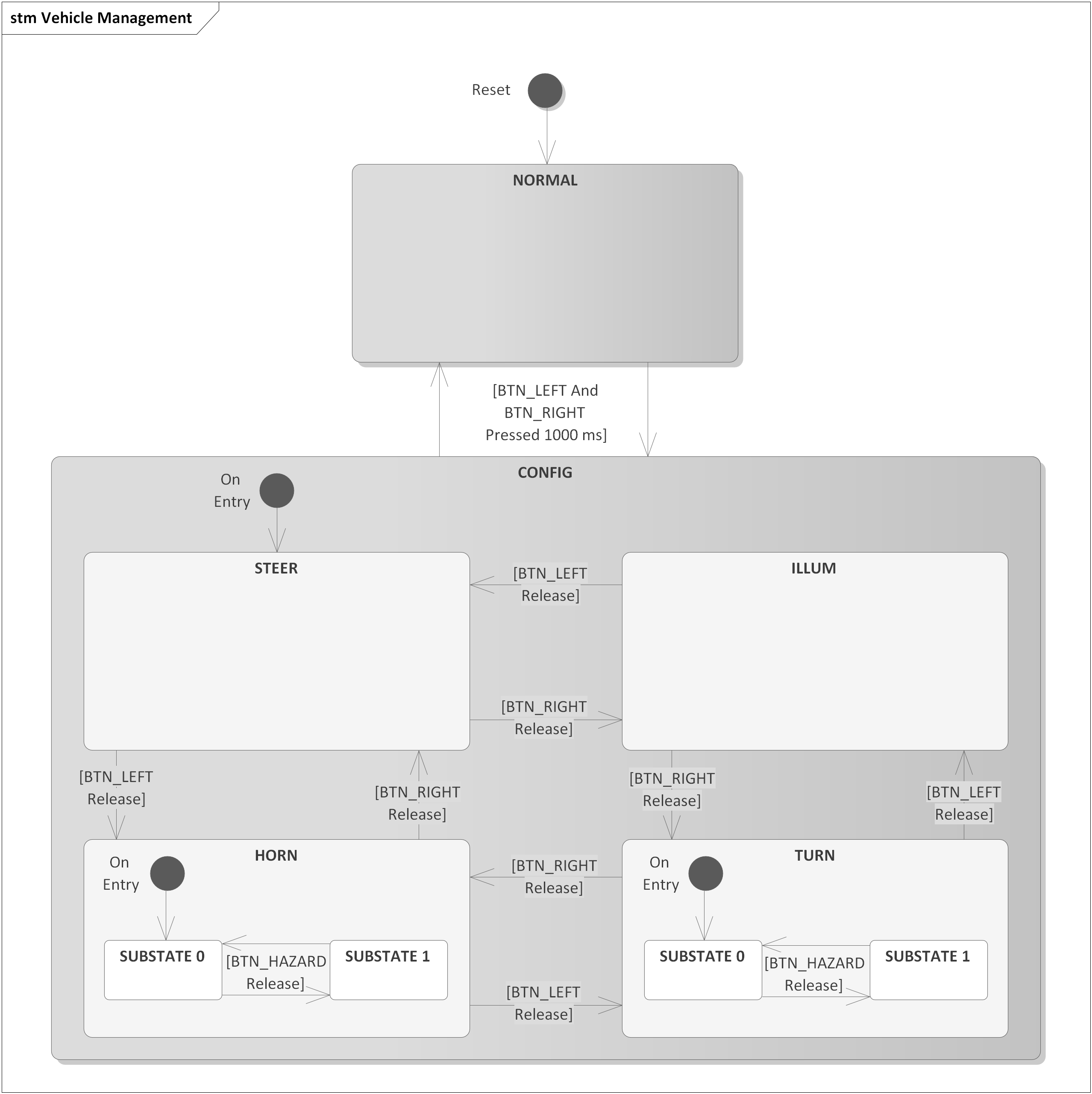


Figure 11 - Vehicle states and substates.

1. The default vehicle state after a reset shall be *NORMAL*.
2. The system shall set and react to the following runtime configuration parameters:

* *SteerConfig*: specifies the wheel steering outer angle calculation method – *STEER\_CONFIG\_PARALLEL* (0) or *STEER\_CONFIG\_ACKERMANN* (1).
* *IllumConfig*: specifies the period of the illumination fading – *[300, 1000] ms* – with a resolution of 10 ms.
* *HornConfig*:
  + *Mode*: specifies the horn activation mode – *HORN\_MODE\_CONTINUOUS* (0) or *HORN\_MODE\_BEEPING* (1).
  + *Period*: specifies the period of the horn signal while in the *HORN\_MODE\_BEEPING* mode – *[50, 150] ms* – with a resolution of 10 ms.
* *TurnConfig*:
  + *Period*: specifies the period of the indication signals – *[200, 800] ms* – with a resolution of 10 ms.
  + *NrOfPeriods*: specifies the number of periods to be used on a short turn indication – *[2, 5]* – with a resolution of 1.

1. The default values for the runtime configuration parameters after a reset shall be:

* *SteerConfig* = *STEER\_CONFIG\_PARALLEL*.
* *IllumConfig* = *500 ms*.
* *HornConfig*.*Mode* = *HORN\_MODE\_CONTINUOUS*.
* *HornConfig*.*Period* = *100 ms*.
* *TurnConfig*.*Period* = *500 ms*.
* *TurnConfig*.*NrOfPeriods* = *3*.

1. While in the *CONFIG* vehicle state, BUZZ\_HORN shall be continuously inactive.
2. While in the *CONFIG* vehicle state SERVO\_TURN\_LEFT and SERVO\_TURN\_RIGHT shall be set in opposing directions at an inner angle of 45 degrees (135 degrees for SERVO\_TURN\_LEFT and 45 degrees for SERVO\_TURN\_RIGHT in trigonometric sense).
3. While in the *CONFIG* vehicle state JOY\_STEER shall have no effect.
4. While in the *CONFIG* vehicle state, the correlation to be implemented between the states, substates, runtime configuration parameters to be saved, LEDs to be turned ON and values to be displayed on DISP7SEG is illustrated in **Table 2**.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **State** | **Substate** | **Configuration Parameter** | **Active LEDs** | **DISP7SEG Values** |
| *STEER* | *-* | *SteerConfig* | LED\_TURN\_FRONT\_LEFT | [0, 1] |
| *ILLUM* | *-* | *IllumConfig* | LED\_TURN\_FRONT\_RIGHT | [30, 100],  100 displayed as “1 ” |
| *HORN* | *SUBSTATE 0* | *HornConfig.*  *Mode* | LED\_TURN\_BACK\_RIGHT, LED\_ILLUM\_NIGHT (10%) | [0, 1] |
| *HORN* | *SUBSTATE 1* | *HornConfig.*  *Period* | LED\_TURN\_BACK\_RIGHT,  LED\_ILLUM\_DAY (10%) | [5, 15] |
| *TURN* | *SUBSTATE 0* | *TurnConfig.*  *Period* | LED\_TURN\_BACK\_LEFT,  LED\_ILLUM\_NIGHT (10%) | [20, 80] |
| *TURN* | *SUBSTATE 1* | *TurnConfig.*  *NrOfPeriods* | LED\_TURN\_BACK\_LEFT,  LED\_ILLUM\_DAY (10%) | [2, 5] |

Table 2 - Correlation between states, substates, configuration parameters, active LEDs and DISP7SEG values.

1. While in the *CONFIG* vehicle state and when BTN\_HORN is not pressed, the user shall select the value to be set for the current runtime configuration parameter, indicated on DISP7SEG, through POT\_LUMINANCE. DISP7SEG dot symbols shall be OFF.
2. While in the *CONFIG* vehicle state and when BTN\_HORN is pressed, the selected value shall be frozen and saved into the current runtime configuration parameter (additional changed via POT\_LUMINANCE shall have no effect on the displayed or saved value). DISP7SEG dot symbols shall be ON.

## Steering

1. The Steering feature shall be active only while in the *NORMAL* vehicle state.
2. When entering the *NORMAL* state, the servomotors shall point forwards and the buzzer shall be muted.
3. JOY\_STEER position shall control the inner angle as follows:

* Left turn: (50 ... 0] % – (0 ... 60] degrees.
* Right turn: (50 ... 100] % – (0 ... 60] degrees.

1. When *SteerConfig* = *STEER\_CONFIG\_PARALLEL,* the outer angle shall be equal to the inner angle.
2. When *SteerConfig* = *STEER\_CONFIG\_ACKERMANN,* the outer angle shall be calculated based on the inner angle through the following equations (for more details see Chapter 3.1):

* 𝛿o = 𝛿𝑖, when 𝛿𝑖 = 0;
* 𝛿o = 0.9654\*𝛿𝑖 + 0.019, when 𝛿𝑖 ∈ (0, 5];
* 𝛿o = 0.8212\* 𝛿𝑖 + 0.836, when 𝛿𝑖 ∈ (5, 25];
* 𝛿o = 0.658\*𝛿𝑖 + 4.3264, when 𝛿𝑖 ∈ (25, 45];
* 𝛿o = 0.6116\*𝛿𝑖 + 6.1664, when 𝛿𝑖 ∈ (45, 60];

1. When BTN\_HORN is pressed, the buzzer shall be activated as follows:

* When *HornConfig*.*Mode* = *HORN\_MODE\_CONTINUOUS*, the buzzer shall be continuously active.
* When *HornConfig*.*Mode* = *HORN\_MODE\_BEEPING*, the buzzer shall be controlled as illustrated in **Figure 12.**



Figure 12 - Horn beeping control signal.

1. When BTN\_HORN is not pressed, the buzzer shall be continuously inactive.

## Turning

1. The Turning feature shall be active only while in the *NORMAL* vehicle state.
2. The Turning feature shall implement the state machine from **Figure 13**.

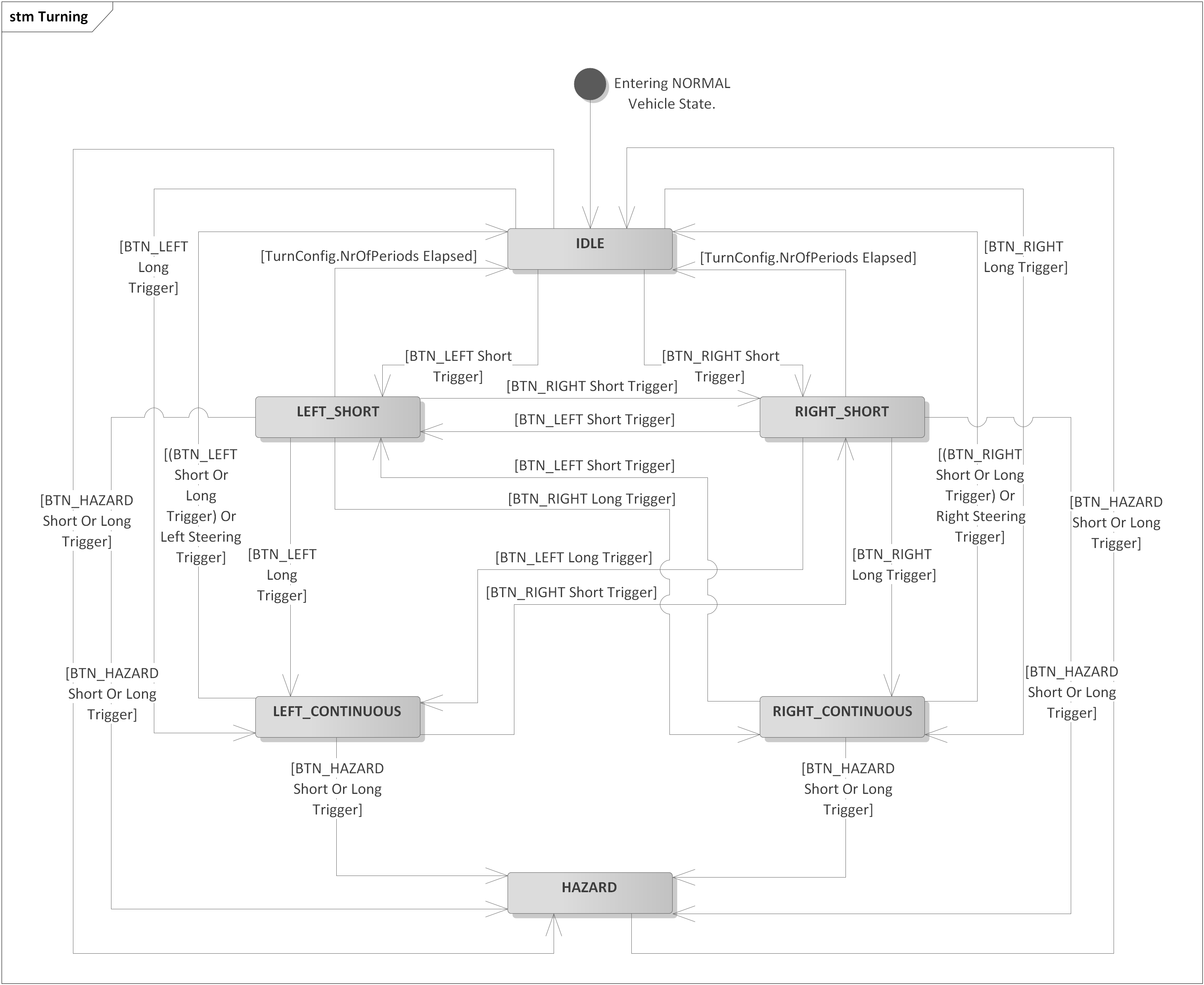


Figure 13 - Turning state machine.

1. While in the LEFT\_SHORT, RIGHT\_SHORT, LEFT\_CONTINUOUS, RIGHT\_CONTINUOUS, HAZARD states, the blinking LEDs shall be controlled as illustrated in **Figure 14**.



Figure 14 - LED blinking control signal.

1. While in the IDLE state, LED\_TURN\_FRONT\_LEFT, LED\_TURN\_FRONT\_RIGHT, LED\_TURN\_BACK\_LEFT and LED\_TURN\_BACK\_RIGHT, shall be turned OFF.
2. While in the LEFT\_SHORT state, LED\_TURN\_FRONT\_LEFT and LED\_TURN\_BACK\_LEFT shall blink a number of times equal to *TurnConfig.NrOfPeriods*.
3. While in the RIGHT\_SHORT state, LED\_TURN\_FRONT\_RIGHT and LED\_TURN\_BACK\_RIGHT shall blink a number of times equal to *TurnConfig.NrOfPeriods*.
4. While in the LEFT\_CONTINUOUS state, LED\_TURN\_FRONT\_LEFT and LED\_TURN\_BACK\_LEFT shall blink continuously.
5. While in the RIGHT\_CONTINUOUS state, LED\_TURN\_FRONT\_RIGHT and LED\_TURN\_BACK\_RIGHT shall blink continuously.
6. While in the HAZARD state, LED\_TURN\_FRONT\_LEFT, LED\_TURN\_FRONT\_RIGHT, LED\_TURN\_BACK\_LEFT and LED\_TURN\_BACK\_RIGHT shall blink continuously.
7. The short button trigger shall take place on the edge of a button being released after being continuously pressed for a period less than 500 ms.
8. The long button trigger shall take place when a button has been continuously pressed for a period of 500 ms (keeping the button pressed for more than 500 ms or releasing the button shall not create any other triggers).
9. The left steering trigger shall take place only in the LEFT\_CONTINUOUS state when the steering wheel has reached a position >= 45% after previously reaching a position <= 35%.
10. The right steering trigger shall take place only in the RIGHT\_CONTINUOUS state when the steering wheel has reached a position <= 55% after previously reaching a position >= 65%.

## Illumination

1. The Illumination feature shall be active only while in the *NORMAL* state.
2. When entering the *NORMAL* vehicle state, the illumination LEDs shall be OFF.
3. The illumination feature shall implement the state machine illustrated in **Figure 15**. The Illumination state machine depends on the following values and operations:

* *CurrentLuminance*: the current value of POT\_LUMINANCE.
* *AverageLuminance*: the average value of *CurrentLuminance*.
* *LastLuminance*: the last processed *AverageLuminance* value for which a fading operation has been triggered.
* *Duty(LuminanceValue)*: the target LED PWM duty cycle corresponding to the given *LuminanceValue*.
* *Fading(LedChannel, StartDuty, TargetDuty, TimeMs)*: fading operation for the *LedChannel* LED, starting from the *StartDuty* PWM duty cycle, finishing when the *TargetDuty* PWM duty cycle value is reached, over a period of *TimeMs* miliseconds.

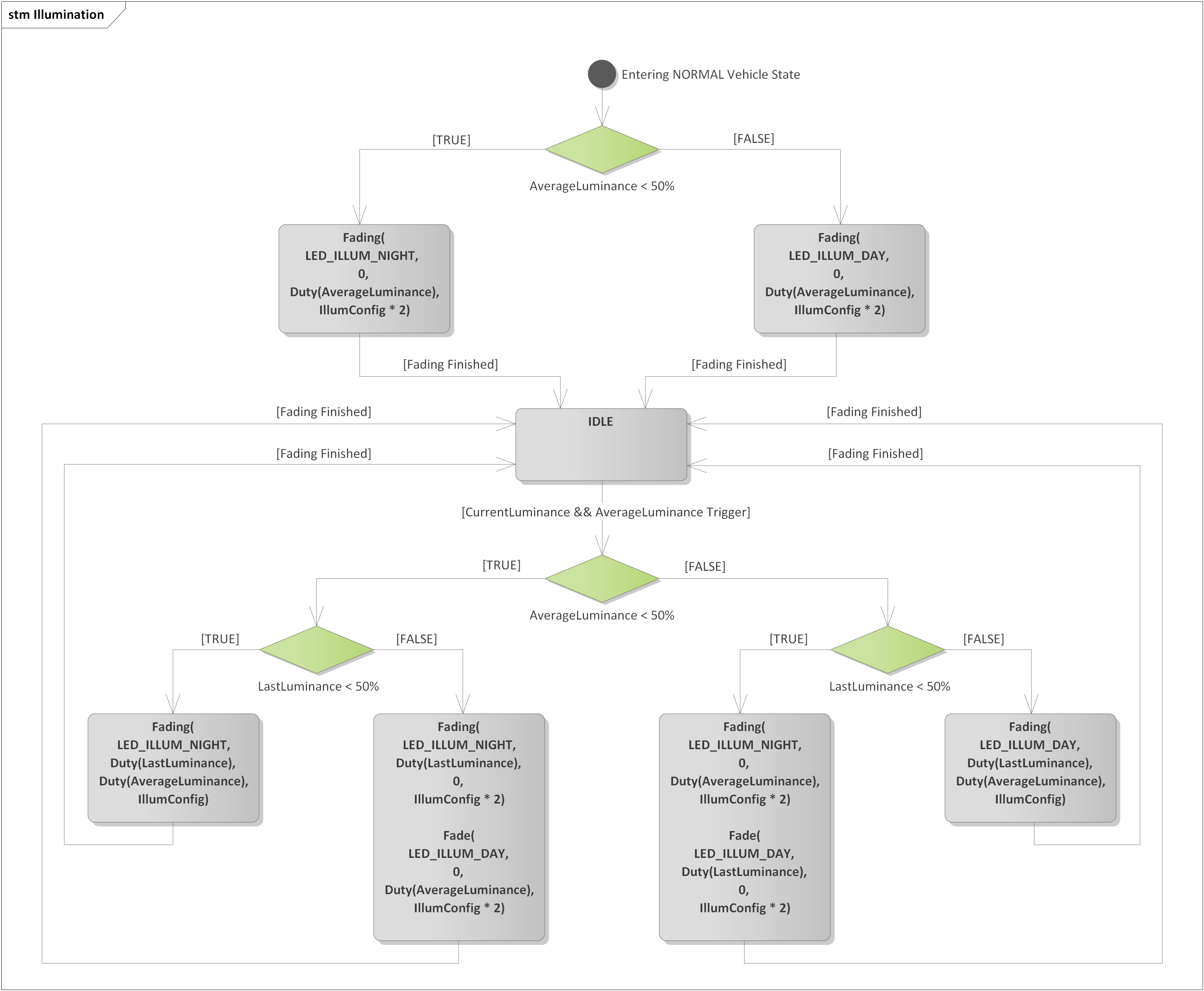


Figure 15 - Illumination state machine.

1. *AverageLuminance* shall be calculated with a moving average of size >= 25 over a period of 500 ms.
2. After entering the NORMAL vehicle state, the first decision point for *AverageLuminance* (< or >= 50%) shall be executed only after a full averaging of *CurrentLuminance* over a period of 500 ms has finished.
3. While in the IDLE state, the PWM duty cycles of both illumination LEDs shall remain constant.
4. Once a *Fading()* operation starts, no additional fading triggers shall take place until the ongoing operation finishes. *AverageLuminance* shall be continuously updated, even if a *Fading()* operation is ongoing.
5. The fading trigger from the IDLE state shall take place only when the following conditions are met. *DELTA* shall be 0.5%.

)

AND

)

AND

OR

1. *Duty(LuminanceValue)* shall be calculated with linear interpolation as follows:

* **Night mode:** *LuminanceValue =* [0, 50] -> *Duty(LuminanceValue) =* [0.5, 15]%;
* **Day mode:** *LuminanceValue =* [50, 100] *-> Duty(LuminanceValue) =* [20, 100]%.

1. The *Fading()* operation shall consist of a variation of PWM duty cycle levels following the following equation:

*FadingLevel(x)* takes values in the [0, 1]interval, defining PWM duty cycle percentages in the [0, 100]% interval (for more details see Chapter 3.4).

1. The *Fading()* operation shall perform a duty cycle level update once every 10 ms.

# Information about this Document

## Copyright

All rights, including translation rights, are reserved. Under no circumstances shall any fragment of this document be reproduced without written authorization from NTT DATA Romania S.A, including copying, photographing or replicating through other methods.

Copyright (c) NTT DATA Romania S.A.

## Version Index

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Chapter** | **Modification description** |
| 1.0 | 10.07.2021 | Nicolae-Bogdan Bacrău | All | Created. |

Table 3 - Version Index.