|  |
| --- |
| \_\_\_\_  ***Project task 1***  *Pololu Zumo 32U4*  Doc.-Number: Pflichtenheft\_Draft3.docx  Doc.-Version: A4  299792458 |
| Customer:  NewTec  Development  Buchenweg 3  89284 Pfaffenhofen a. d. Roth  Germany |
|  |
| Author:  Team: ~~ o=o\  Hs Offenburg |

Change History

|  |  |  |
| --- | --- | --- |
| Doc.-Version | Description of Modification | Date |
| A1 | Initial revision |  |
| A2 | First version of use case diagrams, use case descriptions and requirements written | 20.03.2024 |
| A3 | Use case diagrams, use case descriptions and requirements modified | 27.03.2024 |
| A4 | Use case diagrams, use case descriptions and requirements modified | 30.03.2024 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Release

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Name | Responsibility | Date | Signature |
| Creation |  |  |  | i.A. |
| Verification |  |  |  | i.A. |
| Approval |  |  |  | i.A. |
| Release |  |  |  | i.A. |

Table of Contents

Change History 2

Release 2

Table of Contents 3

List of Tables 4

List of Figures 4

1 General 5

1.1 Abbreviations 5

1.2 Terminology 6

1.2.1 System scenarios 6

1.2.2 Interface scenarios 6

1.2.3 System states 7

1.2.4 Interface states 7

1.2.5 Hardware 8

1.2.6 Other 9

1.3 Referenced Documents 10

1.4 Applicable Standards 10

2 Introduction 11

2.1 System Overview 11

2.2 Interface Overview 12

2.3 Scenarios 13

2.3.1 System 13

2.3.2 Interface 18

3 Requirements 24

3.1 Functional Requirements 24

3.2 Non-Functional Requirements 25

List of Tables

[Table 1: Abbreviations 5](#_Toc162721078)

[Table 2: Terminology of the system scenarios 6](#_Toc162721079)

[Table 3: Terminology of the interface scenarios 6](#_Toc162721080)

[Table 4: Terminology of the system states 7](#_Toc162721081)

[Table 5: Terminology of the interface states 7](#_Toc162721082)

[Table 6: Terminology of the hardware 8](#_Toc162721083)

[Table 7: Other terminology 9](#_Toc162721084)

[Table 8: Referenced Documents 10](#_Toc162721085)

[Table 9: Applicable Standards 10](#_Toc162721086)

List of Figures

[Figure 1 System Overview 11](#_Toc162721087)

[Figure 2 Interface Overview 12](#_Toc162721088)

# General

## Abbreviations

|  |  |
| --- | --- |
| Abbreviation | Description |
| OLED | Organic light emitting diode |
| MCU | Microcontroller unit |

Table 1: Abbreviations

## Terminology

### System scenarios

|  |  |
| --- | --- |
| Term | Description |
| CalibrateLineSensors | The *LineSensors* are calibrated to the current light conditions |
| DisplayTeamName | The *TeamName* is shown on the *OledDisplay* |
| DriveLap | The *Robot* shows the *CountDown* on the *OledDisplay*, drives to the *StartFinishLine* and then drives one *FullLap* |
| SetParameters | Allows the *User* to choose between different *ParameterSets* |
| HandleError | Displays an error message on the *OledDisplay* until the *User* resets it |

Table 2: Terminology of the system scenarios

### Interface scenarios

|  |  |
| --- | --- |
| Term | Description |
| *InitializeMcu* | Required system resources and variables are initialized. The *TeamName* is shown on the *OledDisplay* |
| *CalibrateLineSensors* | The *LineSensors* are calibrated to the current light conditions |
| *GetReadyForLap* | The MCU starts with a *CountDown* from 3 to 0 and the Robot starts moving. |
| *DriveLap* | The *Robot* follows the *GuideLine* |
| *DriveOverGap* | Allows the robot to drive over an interruption in the *GuideLine* |
| *MeasureTime* | Measures the time of a lap. This use case runs parallel to other use cases. |
| *DisplayLapTime* | Displays the completed lap time |
| *HandleError* | Displays an error message on the *OledDisplay* |
| *SetParameters* | Allows parameters to be adjusted during runtime |

Table 3: Terminology of the interface scenarios

### System states

|  |  |
| --- | --- |
| Term | Description |
| LineSensorCalibrationDone | The calibration of the *LineSensors* is finished |
| Ready | The *Robot* is ready to retrieve a command from the *User* |
| Running | The *Robot* is driving on the *PlayField* |
| Setup | The *Robot* shows the parameters and allows them to be adjusted |
| Error | An error sets the *Robot* into an error *State*. This *State* must be reset by the *User* |

Table 4: Terminology of the system states

### Interface states

|  |  |
| --- | --- |
| Term | Description |
| InitializationDone | The initialization of the system is done |
| ReadyToDrive | The *Robot* is ready to drive |
| Drive1 | The *Robot* move to the *StartFinishLine* |
| Drive2 | The *Robot* drives a lap on the *PlayField* |
| DisplayTime | The *Robot* displays the lap time on the *OledDisplay* |
| Setup | The *Robot* shows the parameters and allows them to be adjusted |
| Error | An error sets the *Robot* into an error *State*. This *State* must be reset by the *User* |

Table 5: Terminology of the interface states

### Hardware

|  |  |
| --- | --- |
| Term | Description |
| Robot | Zumo32U4 |
| OledDisplay | The OLED display on the top of the Robot that allows visible feedback to the *User* |
| Buzzer | The buzzer on the *Robot* that allows audio feedback to the *User* |
| LineSensors | The sensor array at the front on the underside of the *Robot* that is able to recognize the lines of the *PlayField* |
| PowerSwitch | The switch at the back of the *Robot*. The *Robot* is only supplied with power when the switch is in the “ON” position |
| ResetButton | The rightmost button on the back of the *Robot* |
| StateButton | The leftmost button on the back of the *Robot* |
| ParamButton | The middle button on the back of the *Robot* |

Table 6: Terminology of the hardware

### Other

|  |  |
| --- | --- |
| Term | Description |
| AlarmSignal | Signal of frequency 440Hz, duration 200ms, repeated thrice in 200ms intervals, volume min. 60dB at 10cm distance |
| CountDown | A countdown that starts with 3 and counts down to 0. The number is always decremented after 1s and is shown on the *OledDisplay* in addition to the *TeamName* |
| DriveMotors | The two metal gearmotors that are connected to the wheels on the backside of the *Robot* |
| FullLap | One lap on the *PlayField* that starts with the *StartFinishLine* and ends with the same *StartFinishLine* |
| FullStop | The *Robot* is not moving because the *DriveMotors* do not receive any power |
| GuideLine | The line on the PlayField that marks the lap and shows the Robot where it must drive |
| ParameterSets | The *Robot* allows to choose between different sets of parameters which affect the *Robot*’s behavior, e.g. a more cautious driving style |
| PlayField | The flat surface the *Robot* drives on |
| PoweredOn | The *Robot* is supplied with power for at least 500ms so that all the software of the Mcu had enough time to finish all initialization steps |
| Redetect | Recognizing *GuideLine* after losing *GuideLine* while driving |
| ShortBeep | Signal of frequency 440Hz, duration 100ms, volume min. 60dB at 10cm distance |
| StartFinishLine | The line perpendicular to the *GuideLine* that marks the beginning and end of the lap |
| State | The state of the Robot. Only one state can be active at the same time |
| TeamName | The name that identifies the team that owns the Robot. The name is defined by the software of the Mcu and cannot be changed |
| User | The person that interacts with the *Robot* |

Table 7: Other terminology

## Referenced Documents

|  |  |  |
| --- | --- | --- |
| Reference | Document-Identification | Description |
| [1] | N/A | N/A |
|  | N/A | N/A |

Table 8: Referenced Documents

## Applicable Standards

|  |  |  |
| --- | --- | --- |
| Reference | Document-Identification | Description |
| [1] | N/A | N/A |
|  | N/A | N/A |

Table 9: Applicable Standards

# Introduction

## System Overview

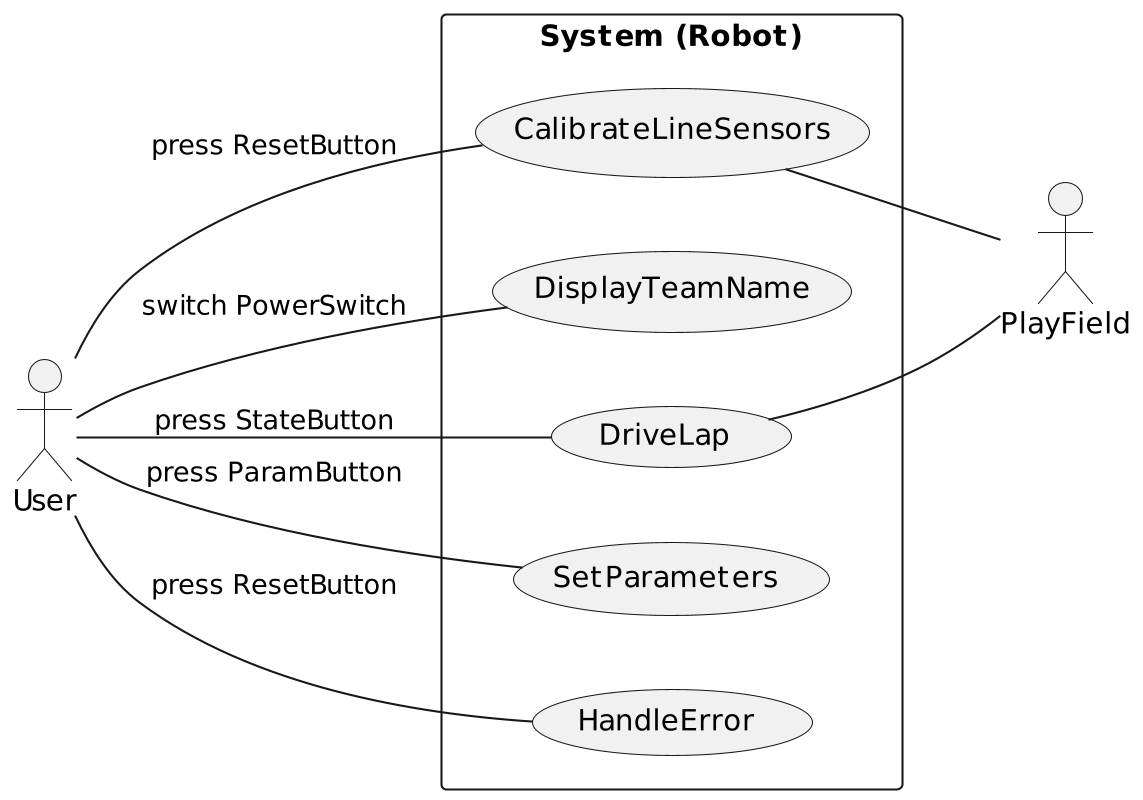
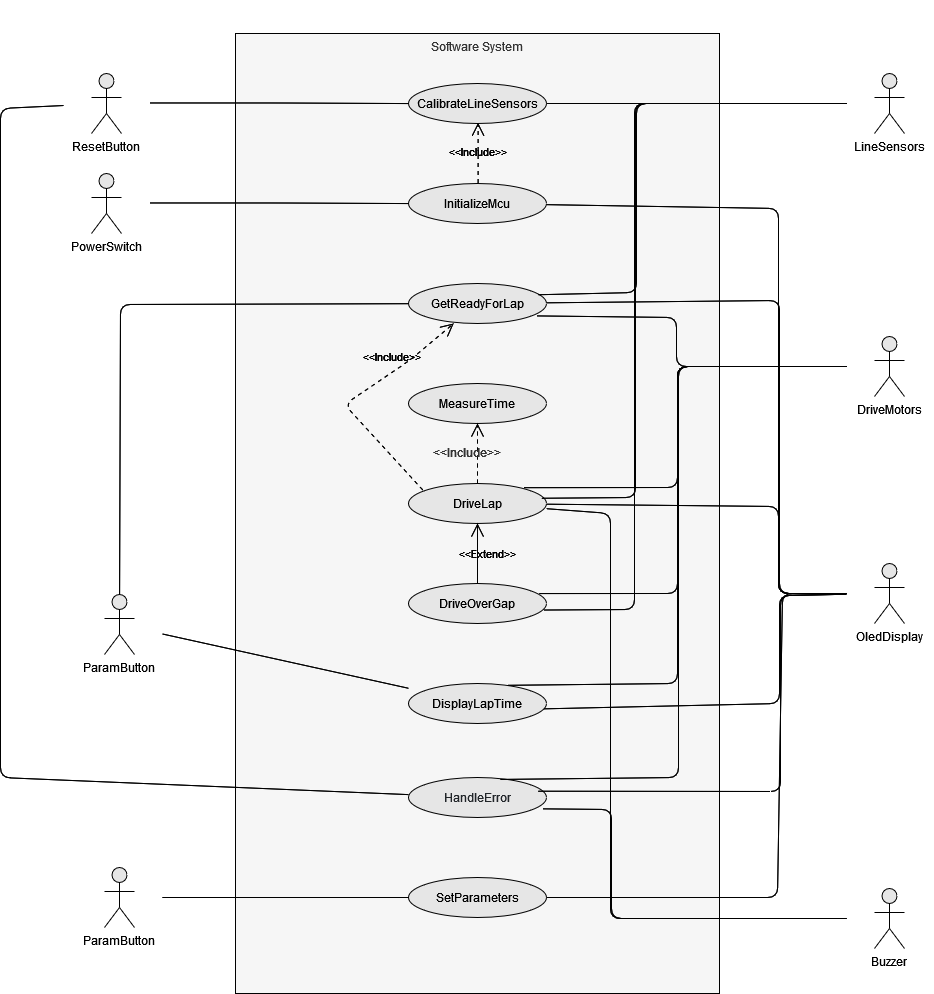


Figure 1 System Overview

## Interface Overview

Figure 2 Interface Overview



## Scenarios

### System

|  |  |
| --- | --- |
| Reference number | 2.3.1.1 |
| Name | *CalibrateLineSensors* |
| Short description | The *LineSensors* are calibrated to the current light conditions |
| Precondition | There are two different sets of preconditions in this use case. All the conditions of one set must be met. The condition sets are mutually exclusive and therefore cannot be both met at the same time.  Precondition set 1 (Robot is powered on)   * The *Robot* is placed on the *GuideLine* of the *PlayField* * The *PowerSwitch* is in the position “OFF”   Precondition set 2 (Calibration is triggered by the user)   * The *Robot* is placed on the *GuideLine* of the *PlayField* * The *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The *State Ready* is active |
| Postcondition | The *State LineSensorCalibrationDone* is active |
| Error case | none |
| Actors | *User, PlayField* |
| Trigger | There are two different triggers.  The first trigger is only effective if precondition set 1 met:   * The *PowerSwitch* is switched from “OFF” to “ON”   The second trigger is only effective if precondition set 2 met:   * The *User* presses the *ResetButton* |
| Standard sequence | 1. The *Robot* is placed on the *GuideLine* of the *PlayField* 2. The *PowerSwitch* is switched from “OFF” to “ON” 3. The *LineSensors* are calibrated to the current light conditions 4. The *State LineSensorCalibrationDone* *is activated* |
| Alternative sequences | 1. The *Robot* is placed on the *GuideLine* of the *PlayField* 2. The *State Ready* is active 3. The *User* presses the *ResetButton* 4. The *LineSensors* are calibrated to the current light conditions 5. The *State LineSensorCalibrationDone* *is activated* |

|  |  |
| --- | --- |
| Reference number | 2.3.1.2 |
| Name | *DisplayTeamName* |
| Short description | The *TeamName* is shown on the *OledDisplay* |
| Precondition | All these conditions must be met:   * The *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The *State LineSensorCalibrationDone* is active |
| Postcondition | All these conditions must be met:   * The *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The *State Ready* is active |
| Error case | none |
| Actors | *User* |
| Trigger | The *State LineSensorCalibrationDone* changes from not active to active |
| Standard sequence | 1. The *State LineSensorCalibrationDone* changes from not active to active 2. The *OledDisplay* is cleared 3. The *TeamName* is shown on the *OledDisplay* and not cleared 4. The *State Ready* is activated |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.1.3 |
| Name | *DriveLap* |
| Short description | The *Robot* shows the *CountDown* on the *OledDisplay*, drives to the *StartFinishLine* and then drives one *FullLap* |
| Precondition | All these conditions must be met:   * The *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The *State Ready* is active |
| Postcondition | All these conditions must be met:   * The *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The *State Ready* is active |
| Error case | There are two independent error cases:   * The first *StartFinishLine* is not recognized within 10s * The *FullLap* is not finished within 20s |
| Actors | *User, PlayField* |
| Trigger | The *User* presses the *StateButton* |
| Standard sequence | 1. The *User* presses the *StateButton* 2. The *State Running* is activated 3. The *Robot* shows the *CountDown* on the *OledDisplay* 4. The *Robot* starts driving, following the line on the *PlayField* 5. The *Robot* recognizes the *StartFinishLine* 6. The *ShortBeep* is played on the *Buzzer* 7. The *Robot* continues driving, following the line on the *PlayField* 8. The *Robot* recognizes the *StartFinishLine* 9. The *ShortBeep* is played on the *Buzzer* 10. The *Robot* goes to a *FullStop* 11. The *State Ready* is activated |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.1.4 |
| Name | *SetParameters* |
| Short description | Allows the *User* to choose between different *ParameterSets* |
| Precondition | All these conditions must be met:   * The *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The *State Ready* is active |
| Postcondition | All these conditions must be met:   * The *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The *State Ready* is active |
| Error case | none |
| Actors | *User* |
| Trigger | The *ParamButton* is pressed |
| Standard sequence | 1. The *ParamButton* is pressed 2. The *State Setup* is activated 3. The OledDisplay shows the *ParameterSets* and allows to select one 4. The *User* presses the *ResetButton* 5. The *State Ready* is activated |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.1.5 |
| Name | *HandleError* |
| Short description | Displays an error message on the *OledDisplay* until the *User* resets it |
| Precondition | All these conditions must be met:   * The *Robot* is *PoweredOn* * An error occurred |
| Postcondition | All these conditions must be met:   * The *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The use case *DisplayTeamName* (2.3.1.2) is activated |
| Error case | none |
| Actors | *User* |
| Trigger | An error occurs |
| Standard sequence | 1. An error occurs 2. The *State Error* is actived 3. The *Robot* goes to a *FullStop* 4. The *AlarmSignal* is played on the *Buzzer* 5. An error message is displayed on the *OledDisplay* 6. The User presses the *ResetButton* 7. The use case *DisplayTeamName* (2.3.1.2) is activated |
| Alternative sequences | none |

### Interface

|  |  |
| --- | --- |
| Reference number | 2.3.2.1 |
| Name | *InitializeMcu* |
| Short description | Required system resources and variables are initialized. The *TeamName* is shown on the *OledDisplay* |
| Precondition | All these conditions must be met:   * The MCU must be offline and have no power * The *PowerSwitch* is in the position “OFF” |
| Postcondition | The *State* *InitializationDone* is active |
| Error case | none |
| Actors | *PowerSwitch, OledDisplay* |
| Trigger | The *PowerSwitch* is switched from “OFF” to “ON” |
| Standard sequence | 1. All system resources are initialized 2. All system variables are initialized 3. The *TeamName* is displayed 4. The *State* *InitializationDone* is activated |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.2.2 |
| Name | *CalibrateLineSensors* |
| Short description | The *LineSensors* are calibrated to the current light conditions |
| Precondition | The *State* *Error* is not active |
| Postcondition | The *State ReadyToDrive* is active |
| Error case | none |
| Actors | *LineSensors, ResetButton* |
| Trigger | There are two different triggers. Both triggers execute the same standard sequence.  The first trigger is:   * The *State* *InitializationDone* is activated   The second trigger is:   * The *ResetButton* is pressed |
| Standard sequence | 1. The *LineSensors* are calibrated 2. The *State* *ReadyToDrive* is activated |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.2.3 |
| Name | *GetReadyForLap* |
| Short description | The MCU starts with a *CountDown* from 3 to 0 and the Robot starts moving. |
| Precondition | The *State* *ReadyToDrive is* active |
| Postcondition | The State *Drive1* is active |
| Error case | There are two independent error cases:   * The *StartFinishLine* was not crossed within 10 seconds |
| Actors | *OledDisplay, DriveMotors, LineSensors* |
| Trigger | The *StateButton* is pressed |
| Standard sequence | 1. The *CountDown* is displayed 2. As soon as 0 is displayed, the *DriveMotors* are supplied with power 3. The State *Drive1 is* activated 4. The robot follows the *GuideLine* |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.2.4 |
| Name | *DriveLap* |
| Short description | The *Robot* follows the *GuideLine* |
| Precondition | There are two different sets of preconditions in this use case. All the conditions of one set must be met. The condition sets are mutually exclusive and therefore cannot be both met at the same time.  Precondition set 1:   * The *State* *Drive1 is* active |
| Postcondition | The *State Drive2* is not active |
| Error case | The *FullLap* is not finished within 20s |
| Actors | *OledDisplay, DriveMotors, LineSensors* |
| Trigger | There are two independent triggers:   * The *StartFinishLine* has been recognized * The use case *DriveOverGap* (2.3.2.5) is finished |
| Standard sequence | 1. The *StartFinishLine* has been recognized 2. The *Buzzer* emits the *ShortBeep* 3. The *State* *Drive2 is* activated 4. The *Robot* follows the *GuideLine* 5. The *StartFinishLine* has been recognized again 6. The *Buzzer* emits the *ShortBeep* 7. The *State* *DisplayTime is* activated |
| Alternative sequences | 1. The use case *DriveOverGap* (2.3.2.5) is finished 2. The *State* *Drive2 is* activated 3. The *Robot* follows the *GuideLine* 4. The *StartFinishLine* has been recognized again 5. The *Buzzer* emits the *ShortBeep* 6. The *State* *DisplayTime is* activated |

|  |  |
| --- | --- |
| Reference number | 2.3.2.5 |
| Name | *DriveOverGap* |
| Short description | Allows the robot to drive over an interruption in the *GuideLine* |
| Precondition | All these conditions must be met:   * The *State* *Drive2* is active |
| Postcondition | The *State* *Drive2* is active |
| Error case | *GuideLine* is not recognized within 5s |
| Actors | *DriveMotors, LineSensors* |
| Trigger | The *GuideLine* is lost |
| Standard sequence | 1. The *Robot* searches for the *GuideLine* 2. The *GuideLine* has been recognized 3. The use case *DriveLap* (2.3.2.4) is activated |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.2.6 |
| Name | *MeasureTime* |
| Short description | Measures the time of a lap. This use case runs parallel to other use cases. |
| Precondition | All these conditions must be met:   * The *State* *Drive1* is active |
| Postcondition | The *State* *Drive2* is not active |
| Error case | none |
| Actors | none |
| Trigger | The *StartFinishLine* is recognized |
| Standard sequence | 1. The timer is started 2. The *StartFinishLine* is recognized again 3. The timer is stopped |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.2.7 |
| Name | *DisplayLapTime* |
| Short description | Displays the completed lap time |
| Precondition | The *State* *Drive2 is* active |
| Postcondition | The *State* *ReadyToDrive is* active |
| Error case | none |
| Actors | *OledDisplay, DriveMotors* |
| Trigger | The *State* *DisplayTime* changes from not active to active |
| Standard sequence | 1. Stops the power supply to the *DriveMotors* 2. Shows the completed lap time on the *OledDisplay* 3. The *ParamButton* is pressed 4. The *State* *ReadyToDrive* is activated |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.2.8 |
| Name | *HandleError* |
| Short description | Displays an error message on the *OledDisplay* |
| Precondition | An error occurred |
| Postcondition | The *State* *ReadyToDrive* isactive |
| Error case | none |
| Actors | *OledDisplay, Buzzer, DriveMotors, ResetButton* |
| Trigger | An error occurs |
| Standard sequence | 1. The State *Error is active* 2. Stops the power supply to the Drive*Motors* 3. The *AlarmSignal* is played on the *Buzzer* 4. An error message is displayed on the *OledDisplay* 5. The ResetButton is pressed 6. The State *ReadyToDrive* is actived |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.2.9 |
| Name | *SetParameters* |
| Short description | Allows parameters to be adjusted during runtime |
| Precondition | The *State* *Error* is not active |
| Postcondition | The *State* *ReadyToDrive is* active |
| Error case | none |
| Actors | *OLED-Display, ParamButton* |
| Trigger | The *ParamButton* button has been pressed |
| Standard sequence | 1. The *State Setup* is activated 2. The *OledDisplay* shows the *ParameterSets* and allows to select one 3. The *User* presses the *ResetButton* 4. The *State ReadyToDrive* is activated |
| Alternative sequences | none |

# Requirements

## Functional Requirements

3.1.1 The *Robot* shall start driving 3s after the operator presses the *StateButton*.

3.1.2 The *Robot* shall *CountDown* before the *Robot* starts driving

3.1.3 After pressing the *StateButton* the *Robot* shall detect the *StartFinishLine* in 10s or less

3.1.4 If the *Robot* does not detect the *StartFinishLine* in 10s or less the *Robot* shall detect an error

3.1.5 If the *Robot* detects the *StartFinishLine* the *Robot* shall do ALL of the following steps

* emit a *ShortBeep* via *Buzzer*
* start the time measurement for the lap

3.1.6 If the *Robot* detects the *StartFinishLine* again the *Robot* shall do ALL of the following steps

* stop any motion
* emit a *ShortBeep* via *Buzzer*
* end the time measurement for the lap
* display the measured time for the lap on the *OledDisplay*

3.1.7 The *Robot* shall complete one lap in 20s or less

3.1.8 If the *Robot* does not complete a lap in less than 20s the *Robot* shall detect an error

3.1.9 If the *Robot* leaves the *GuideLine*, the robot shall *Redetect* the track in 5s or less

3.1.10 If the *Robot* does not *Redetect* the *GuideLine* in less than 5s the Robot shall detect an error

3.1.11 If the *Robot* detects an error the *Robot* shall do the following steps in order

* stop any motion
* emit the *AlarmSignal* via *Buzzer*
* display the error reason on the *OledDisplay*

3.1.12 When the *ResetButton* is pressed while the *Robot* detects an error the *Robot* shall clear the error message from the *OledDisplay* AND displays the *TeamName*

3.1.13 When the *Robot* is *PoweredOn* the Robot shall display the *TeamName* on the *OledDisplay* for at least 2s

3.1.14 During the competition the only changes to the software shall be the selection of a predefined list of *ParameterSets* via the *ParamButton*

3.1.15 When the *Robot* is *PoweredOn* the *Robot* shall calibrate the *LineSensors*

3.1.16 When the *ResetButton* is pressed, while the *Robot* is not detecting an error the *Robot* shall calibrate the *LineSensors*

## Non-Functional Requirements

3.2.1 The software shall be hardware independent

3.2.2 The programmer shall write the software in the programming language C

3.2.3 The *Robot* shall run during daytime- or officelight conditions

3.2.4 The *Robot* shall start the competition on the *GuideLine*

3.2.5 The *Robot* shall be powered on when it is positioned on the *GuideLine*

3.2.6 The software shall at most use 80% of the available flash memory