|  |
| --- |
| \_\_\_\_  ***Project task 1***  *Pololu Zumo 32U4*  Doc.-Number: Pflichtenheft\_A6.docx  Doc.-Version: A6  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 |
| A5 | Use case diagrams, use case descriptions and requirements modified | 03.04.2024 |
| A6 | Use case diagrams, use case descriptions and requirements modified | 11.04.2024 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

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 19

3 Requirements 25

3.1 Functional Requirements 25

3.1.1 Before start 25

3.1.2 During run 25

3.1.3 After run 25

3.1.4 Others 26

3.2 Non-Functional Requirements 26

List of Tables

[Table 1: Abbreviations 5](#_Toc163732042)

[Table 2: Terminology of the system scenarios 6](#_Toc163732043)

[Table 3: Terminology of the interface scenarios 6](#_Toc163732044)

[Table 4: Terminology of the system states 7](#_Toc163732045)

[Table 5: Terminology of the interface states 7](#_Toc163732046)

[Table 6: Terminology of the hardware 8](#_Toc163732047)

[Table 7: Other terminology 9](#_Toc163732048)

[Table 8: Referenced Documents 10](#_Toc163732049)

[Table 9: Applicable Standards 10](#_Toc163732050)

List of Figures

[Figure 1 System Overview 11](#_Toc163732051)

[Figure 2 Interface Overview 12](#_Toc163732052)

# 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 to choose between different *ParameterSets* |

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* allows to choose between different sets of parameters which affect the *Robot*’s behavior, e.g. a more cautious driving style |
| 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 |
| DriveToStart | The *Robot* move to the *StartFinishLine* |
| DriveToFinish | The *Robot* drives a lap on the *PlayField* |
| StateDriveOverGap | The Robot drives over a gap in the *GuideLine* |
| 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 as described in PololuDataSheet[2] |
| OledDisplay | The OLED display on the top of the Robot that allows visible feedback to the *User* as described in PololuDataSheet[2] |
| Buzzer | The buzzer on the *Robot* that allows audio feedback to the *User* as described in PololuDataSheet[2] |
| 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 as described in PololuDataSheet[2] |
| StartButton | The leftmost user button on the back of the *Robot* as described in PololuDataSheet[2] |
| ParamButton | The middle user button on the back of the *Robot* as described in PololuDataSheet[2] |
| CalibrateButton | The rightmost user button on the back of the *Robot* as described in PololuDataSheet[2] |
| ResetButton | The reset button on the back of the *Robot* with the label “reset” as described in PololuDataSheet[2] |

Table 6: Terminology of the hardware

### Other

|  |  |
| --- | --- |
| Term | Description |
| AlarmSignal | The *AlarmSignal* is a signal of frequency 440Hz which is played for 333ms and after a pause of 333ms played again for 333ms. The volume is a minimum of 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 |
| LapTime | The time the Robot needed to finish a *FullLap* |
| 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 | Searching and recognizing the *GuideLine* after losing *GuideLine* while driving |
| ShortBeep | The ShortBeep is a signal of frequency 440Hz with a duration of 1000ms and a minimum volume of 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] | SpecificationSheet | The document „11001\_0099\_0088\_RD-Product-Specification.pdf“ which describes the specifications oft he project |
| [2] | PololuDataSheet | The datasheet oft he *Robot* with the name „zumo\_32u4\_oled\_robot.pdf“ |

Table 8: Referenced Documents

## Applicable Standards

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

Table 9: Applicable Standards

# Introduction

## System Overview

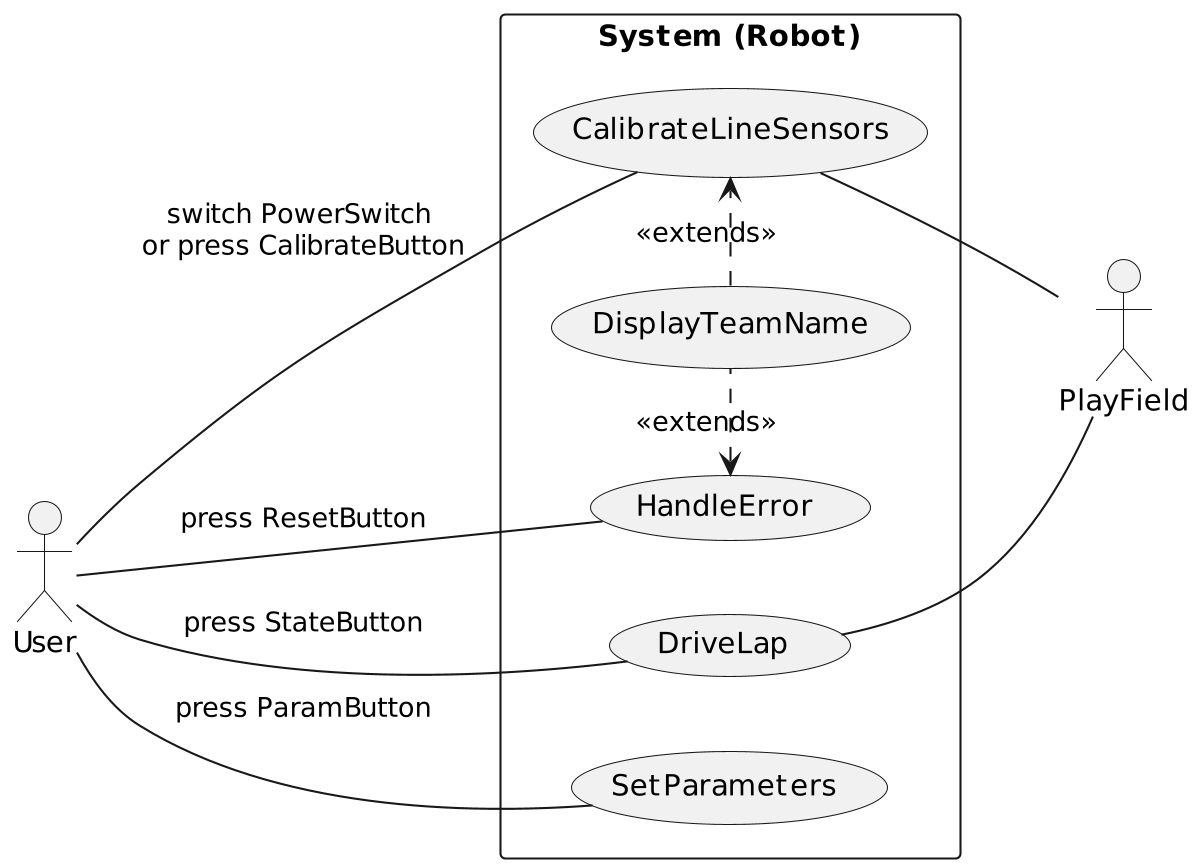


Figure 1 System Overview

## Interface Overview

StartButton

Figure 2 – Interface Overview

Figure 3 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* releases the *CalibrateButton* |
| Standard sequence | 1. The *Robot* is placed on the *GuideLine* of the *PlayField*  2a. 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 | 2b.1. The *State Ready* is active  2. The *User* releases the *CalibrateButton*  3. Go to standard sequence 3. |

|  |  |
| --- | --- |
| Reference number | 2.3.1.2 |
| Name | *DisplayTeamName* |
| Short description | The *TeamName* is shown on the *OledDisplay* |
| 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 *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The *State LineSensorCalibrationDone* is active   Precondition set 2:   * The *Robot* is *PoweredOn* * The *Robot* is in *FullStop* * The *State Error* 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 | none |
| Trigger | There are two different triggers.  The first trigger is only effective if precondition set 1 met:   * The *State LineSensorCalibrationDone* changes from not active to active   The second trigger is only effective if precondition set 2 met:   * The use case *HandleError* is finished |
| 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 three independent error cases:   * The first *StartFinishLine* is not recognized within 10s * The *GuideLine* is lost and not *Redetected* within 5s * The *FullLap* is not finished within 20s |
| Actors | *User, PlayField* |
| Trigger | The *User* releases the *StartButton* |

(This scenario is continued on the next page)

(Continuation of the scenario *DriveLap* (2.3.1.3))

|  |  |
| --- | --- |
| Standard sequence | 1. The *User* releases the *StartButton*  2. The *State Running* is activated  3. The *Robot* shows the *CountDown* on the *OledDisplay*  4. The *Robot* starts the start timer  5. The *Robot* starts driving, following the *GuideLine* on the *PlayField*  6a. The *Robot* recognizes the *StartFinishLine*  7. The *Robot* starts the lap timer  8. The *ShortBeep* is played on the *Buzzer*  9a. The *Robot* continues driving, following the *GuideLine* on the *PlayField*  10a. The *Robot* recognizes the *StartFinishLine*  11. The *ShortBeep* is played on the *Buzzer*  12. The *Robot* goes to a *FullStop*  13. The *LapTime* is shown on the *OledDisplay*  14. The *User* releases the *StartButton*  15. The *State Ready* is activated |
| Alternative sequences | This sequence describes the error case when the first *StartFinishLine* is not recognized within 10s:  6b.1. The *StartFinishLine* is not recognized within 10s after the start of  the start timer  2. The use case *HandleError* (2.3.1.5) is activated |
| This sequence extends the standard sequence to be able to *Redetect* the *GuideLine* e.g. after a gap:  9b.1. The *Robot* does not recognize the *GuideLine*  2. The Robot starts the gap timer  3a. The Robot *Redetects* the *GuideLine*  4. Go to standard sequence 9a. |
| This sequence describes the error case when the *GuideLine* is lost and not *Redetected* within 5s:  9b.3b.1 The *Robot* does not *Redetect* the *GuideLine* within 5s after the start  of the gap timer  2. The use case *HandleError* (2.3.1.5) is activated |
| This sequence describes the error case when the *FullLap* is not finished within 20s:  10b.1. The *Robot* does not recognize the *StartFinishLine* within 20s after  the start of the lap timer  2. The use case *HandleError* (2.3.1.5) is activated |

|  |  |
| --- | --- |
| 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 released |
| Standard sequence | 1. The *ParamButton* is released 2. The *State Setup* is activated 3. Cycle to the next *ParameterSet* 4. The *OledDisplay* shows the number of the active *ParameterSet* 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 *CalibrateLineSensors* (2.3.1.1) is activated |
| Error case | none |
| Actors | *User* |
| Trigger | An error occurs |
| Standard sequence | 1. An error occurs 2. The *State Error* is activated 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* places the *Robot* on the *GuideLine* 7. The *User* presses the *ResetButton*   (This triggers a hard reset with the same effect as if the *PowerSwitch* is switched from “OFF” to “ON”. The following steps are part of the use case *CalibrateLineSensors* (2.3.1.1)) |
| 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. The *PowerSwitch* is switched from “OFF” to “ON” 2. All system resources are initialized 3. All system variables are initialized 4. The *TeamName* is displayed on the *OledDisplay* 5. 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, CalibrateButton* |
| 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 *CalibrateButton* is released |
| Standard sequence | 1. The *State* *InitializationDone* is activated 2. The *LineSensors* are calibrated 3. The *State* *ReadyToDrive* is activated |
| Alternative sequences | 1. The *CalibrateButton* is released 2. The *LineSensors* are calibrated 3. The *State* *ReadyToDrive* is activated |

|  |  |
| --- | --- |
| 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 *DriveToStart* is not active |
| Error case | The *DriveToStart* is active for more than 10s |
| Actors | *OledDisplay, DriveMotors, LineSensors* |
| Trigger | The *StartButton* is released |
| Standard sequence | 1. The *StartButton* is released 2. The *CountDown* is displayed 3. As soon as 0 is displayed, the *DriveMotors* are supplied with power 4. The State *DriveToStart* isactivated 5. The *Robot* follows the *GuideLine* 6. The *StartFinishLine* is recognized 7. The *State* *DriveToFinish* is activated |
| Alternative sequences | 1. The State DriveToStart is active for more than 10 s 2. The use case HandleError (2.3.2.8) is entered |

|  |  |
| --- | --- |
| Reference number | 2.3.2.4 |
| Name | *DriveLap* |
| Short description | The *Robot* follows the *GuideLine* |
| Precondition | The *State* *DriveToFinish* isactive |
| Postcondition | The *State DriveToFinish* is not active |
| Error case | The *FullLap* is not finished within 20s |
| Actors | *OledDisplay, DriveMotors, LineSensors* |
| Trigger | There are two independent triggers:   * The *State* *DriveToFinish* changes from not active to active * The use case *DriveOverGap* (2.3.2.5) is finished |
| Standard sequence | 1a. The State DriveToFinish changes from not active to active  2. The *Buzzer* emits the *ShortBeep*  3a. The *Robot* follows the *GuideLine*  4a. The *StartFinishLine* is again  5. The *Buzzer* emits the *ShortBeep*  6. The *State* *DisplayTime is* activated |
| Alternative sequences | 1b.1. The use case *DriveOverGap* (2.3.2.5) is finished  2. The *State* *DriveToFinish is* activated  3. Go to standard sequence 3a. |
|  | 3b.1. The Robot encounters a gap  2. The use case *DriveOverGap* (2.3.2.5) is entered |
|  | 4b.1. The *StartFinishLine* is not detected after 20s  2. The use case *HandleError* (2.3.2.8) is entered |

|  |  |
| --- | --- |
| Reference number | 2.3.2.5 |
| Name | *DriveOverGap* |
| Short description | Allows the *Robot* to drive over an interruption in the *GuideLine* |
| Precondition | The *State* *DriveToFinish* is active |
| Postcondition | The *State* *DriveToFinish* is active |
| Error case | *GuideLine* is not recognized within 5s |
| Actors | *DriveMotors, LineSensors* |
| Trigger | The *GuideLine* is lost |
| Standard sequence | 1. The *State* *StateDriveOverGap* is activated  2. The *Robot* searches for the *GuideLine*  3a. The *GuideLine* is recognized  4. The use case *DriveLap* (2.3.2.4) is activated |
| Alternative sequences | 3b.1. The *GuideLine* is not recognized within 5s  2. The use case *HandleError* (2.3.2.8) is entered |

|  |  |
| --- | --- |
| Reference number | 2.3.2.6 |
| Name | *MeasureTime* |
| Short description | Measures the time of a *FullLap*. This use case runs parallel to other use cases. |
| Precondition | The *State* *DriveToStart* is active |
| Postcondition | The *State* *DriveToFinish* is not active |
| Error case | none |
| Actors | *LineSensors* |
| Trigger | The *StartFinishLine* is recognized |
| Standard sequence | 1. The *StartFinishLine* is recognized 2. The timer is started 3. The *StartFinishLine* is recognized 4. The timer is stopped |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.2.7 |
| Name | *DisplayLapTime* |
| Short description | Displays the completed *LapTime* |
| Precondition | The *State* *DriveToFinish 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. The *State* *DisplayTime* changes from not active to active 2. Stops the power supply to the *DriveMotors* 3. Shows the completed lap time on the *OledDisplay* 4. The *StartButton* is released 5. 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* *Error* isnot active |
| Error case | none |
| Actors | *OledDisplay, Buzzer, DriveMotors, ResetButton* |
| Trigger | An error occurs |
| Standard sequence | 1. An error occurs 2. The *State* *Error is activated* 3. Stops the power supply to the Drive*Motors* 4. The *AlarmSignal* is played on the *Buzzer* 5. An error message is displayed on the *OledDisplay* 6. The *ResetButton* is released 7. The use case *InitializeMcu* (2.3.2.1)is activated |
| Alternative sequences | none |

|  |  |
| --- | --- |
| Reference number | 2.3.2.9 |
| Name | *SetParameters* |
| Short description | Allows to choose between different *ParameterSets* |
| 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 released |
| Standard sequence | 1. The *ParamButton* is released 2. The *State Setup* is activated 3. Cycle to the next *ParameterSet* 4. The *OledDisplay* shows the number of the active *ParameterSet* 5. The *State ReadyToDrive* is activated |
| Alternative sequences | none |

# Requirements

## Functional Requirements

### Before start

3.1.1.1 If the *Robot* is *PoweredOn* the *Robot* shall do ALL of the following steps

* display the *TeamName* on the *OledDisplay* for at least 2s
* calibrate the *LineSensors*

3.1.1.2 Upon releasing the *StartButton* the *Robot* shall do ALL of the following steps in the given order

* *CountDown*
* *start the time measurement for timeout*
* *start following the GuideLine*

3.1.1.3 If the *Robot* does not detect the *StartFinishLine* in 10s or less the *Robot* shall go into the *Error* *State*

*3.1.1.4 The Robot shall enter the DriveToFinish state in standard sequence if ALL of the following conditions are met*

* *is in DriveToStart state*
* *detects the StartFinishLine*

3.1.1.6 If the *Robot* enters the DriveToFinish state via standard sequence the Robot shall do ALL of the following steps

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

### During run

3.1.2.1 If the *Robot* does not complete a *FullLap* in less than 20s the *Robot* shall go into the *Error* *State*

3.1.2.2 If the *Robot* leaves the *GuideLine*, the *Robot* shall *Redetect* the *GuideLine* in 5s or less

3.1.2.3 If the *Robot* does not *Redetect* the *GuideLine* in less than 5s the *Robot* shall go into *Error* *State*

*3.1.2.4 If the Robot Redetects the GuideLine the Robot shall follow the GuideLine*

### After run

3.1.3.1 The Robot shall enter the DisplayLapTime state when ALL of the following conditions are met

* is in state DriveToFinish
* detects the StatrtFinishLine

3.1.3.2 If the *Robot* *enter the DisplayLapTime state* the *Robot* shall do ALL of the following steps in the given order

* end the time measurement for the *FullLap*
* go to *FullStop*
* emit a *ShortBeep* via *Buzzer*
* display the measured time for the *FullLap* on the *OledDisplay*

### Others

3.1.4.1 If the *Robot* goes into the *Error* *State* the *Robot* shall do the following steps in order

* Go to *FullStop*
* emit an *AlarmSignal* via *Buzzer*
* display the error reason on the *OledDisplay*

3..1.4.2 Upon releasing the *ResetButton* the *Robot* shall enter the PoweredOn state

3.1.4.3 Upon releasing the *CalibrateButton* the *Robot* shall do ALL of the following steps

* calibrate the *LineSensors*
* display the *TeamName* on the *OledDisplay* for at least 2s

3.1.4.4 Upon releasing the *ParamButton* the Robot shall do ALL of the following

* activate the next ParameterSet
* display the active ParameterSet on the OledDisplay
* enter the ReadyToDriveState

## 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

3.2.7 The dimensions of the *GuideLine* AND the *StartFinishLine* are defined in the SpecificationSheet[1]

3.2.8 The *Robot* shall be placed on the *GuideLine* of the *PlayField* with a minimum distance of 1cm in from the *StartFinishLine*