|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Project Number & Name | | | | | | Local Project Reference | Document state |
| Autonomous Parallel Parking System | | | | | | LIGA AC LABS | Registered |
| Document id | | Document title | | | | | Revision & Ver. |
| 001 | | Application Detailed Design | | | | | 000 v1 |
| File Name | | | | | Pages | | Type |
| **APPS\_Design.doc** | | | | | 11 | | **Specification** |
| **Author** | SW Designer | | | *ALV Document review policy* | Date of Release (dd-mm-yy) | | |
| Lucian Popa | | | No Review Document | **26-04-2016** | | |
|  | | |

|  |  |
| --- | --- |
|  | **Name(s)** |
|  | Lucian Popa |

**User Guide:**

This is a hidden text. This text will not be printed.

In the cartridge, fields in dark blue will be automatically updated by PLM and MS Word as following:

* Fields “File Name” and “Pages” are filled in by **MS Word** (Update command to be active),
* other fields in blue are **automatically updated at file download from PLM**:

So, in the cartridge **you have only to update** fields in red (then to put them in black):

* Author Job Position
* Formal reviewers with job positions and names

From second page, the header will be automatically updated at PLM download as well.

|  |  |
| --- | --- |
| **Purpose** | This document is the main document of the software design for Autonomous Parallel Parking System.Description of the purpose of the document and target for this new revision. Gives an overview of the document, and changes made in the specific revision |

Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| **Date**  dd-mmm-yy | **Author(s)** | **eAPDS Revision** | **Description/comment** |
| 26-Apr-16 | Lucian Popa | 000 v1 | First draft version |

Table of content

1. About this document 4

1.1. Terminology and definitions 4

1.2. Reference documents 4

1.2.1. External documents 4

2. Functional overview 5

3. Public interface 6

4. Data structures 7

5. Usage example 8

6. Internal functions 10

6.1. Functions 10

6.2. DEBUG COMMANDS 10

7. Resource usage 12

8. Requirements 13

# About this document

## Terminology and definitions

|  |  |
| --- | --- |
| **Terminology** | **Meaning** |
| ADC | Analog to digital converter |
| IR | Infrared |
| US | Ultrasonic |

## Reference documents

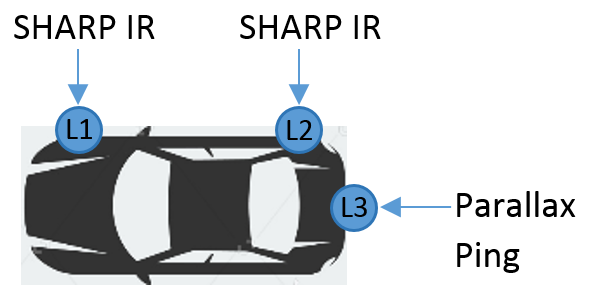
### External documents

|  |  |  |
| --- | --- | --- |
|  | **Title** | **Reference & revision**  **(If applicable)** |
|  | ATMEGA 2560 DATASHEET | <http://www.atmel.com/images/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf> |
|  | PING ULTRASONIC DISTANCE SENSOR DATASHEET | <https://www.parallax.com/sites/default/files/downloads/28015-PING-Sensor-Product-Guide-v2.0.pdf> |
|  | ULTRASONIC RANGING MODULE HC-SR04 DATASHEET | <http://www.micropik.com/PDF/HCSR04.pdf> |
|  | INFRARED DISTANCE MEASURING SENSOR UNIT DATASHEET | <https://www.pololu.com/file/0J845/GP2Y0A41SK0F.pdf.pdf> |
|  | RGB LEDs | <http://ro.farnell.com/kingbright/l-154a4sureqbfzgew/led-multicol-rgb-5mm-x-bright/dp/2290375> |

# Functional overview

The system consists of a RC-Car with two DC motors that are controlled by an Arduino development board (Arduino Mega). The chip according to data received from three sensors (2 infrared distance sensors, 1 ultrasonic distance sensor) makes the decisions.

We have configured a timer, which is used for reading the sensor values from the car and to set up a blinking LED when it is needed, at a certain period of time. Because we need an interrupt every 5 ms-s and the Arduino clock speed is 16 MHz we used a prescaler of 1024 (setting the CS50 and CS52 bits from TCCR5B register) to get a lower frequency therefore we get a higher period of time needed for the timers register to increment. The current value of the timer register’s is being compared to an outside register (OCR5A a 16 bit register with its max value being 65535) every time the timer’s register increments. We have set this register’s value as 78 in order to get an interrupt every 5 ms-s. To enable the clear timer on compare mode, we have set the WGM25 bit from the TCCR5B register. To enable interrupt on compare match we have set OCIE5A bit from the TIMSK5 register.



Three main state machines define the system’s whole functionality:

* Parking place detection state machine – we call this *Detection ALGO*;
* Parallel parking state machine – we call this *Parking ALGO*;
* LED state machine – we call this *LED State ALGO*.

1. Detection ALGO

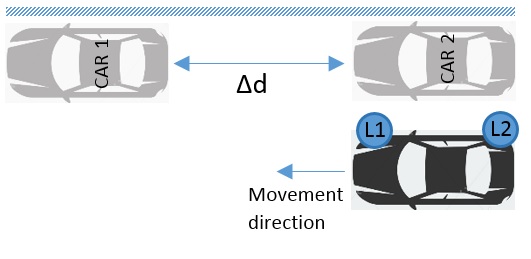
The car will process information to establish its state based on:

* The position of two parked cars;
* The distance between the parked cars – considered to be enough for the car to park

The position of the cars and the space between them is calculated using the two lateral sensors (L1 and L2 – SHARP IR Sensors). In this state machine no information from L3 – PARALLAX US Sensor– is used. In order to be able to fit inside the parking space, the distance between the cars should be 1.15 multiplied by the length of our car.

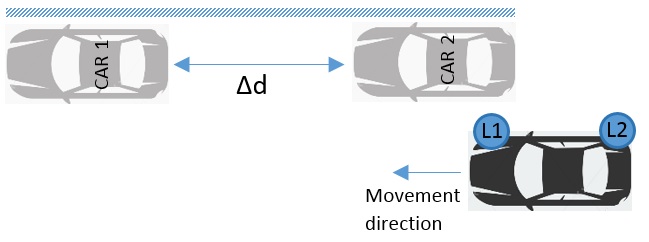
State 1:

* Previous state is ***ON***;
* L1 has detected an object;
* L2 has detected an object;



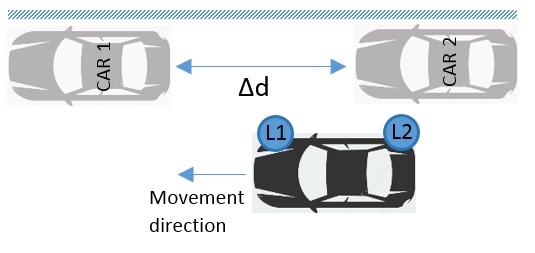
State 2:

* Previous state is ***ON***;
* L1 has detected an object;
* L2 has not detected an object;

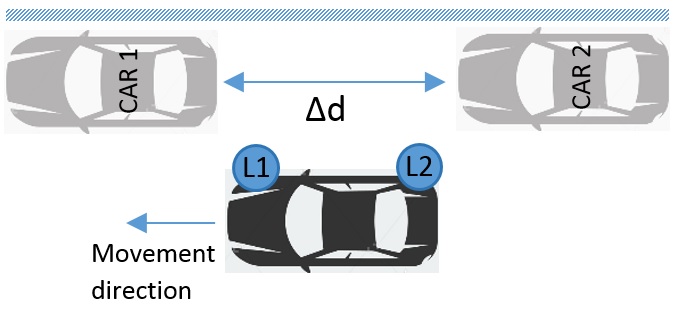


State 3:

* Previous state is ***ON***;
* L1 has not detected an object;
* L2 has detected an object;

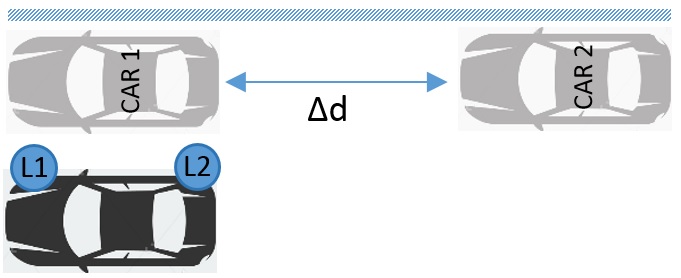


State 4:

* Previous state is ***State1****,* ***State2*** or ***State 3***;
* L1 has not detected an object;
* L2 has not detected an object;

State 5:

* Previous state is ***State4***
* L1 has detected an object
* L2 has detected an object
* Car will stop.



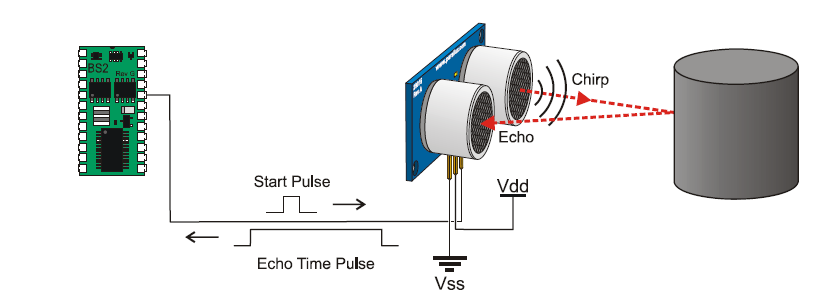
1. Parking ALGO [TBD]

The car will process information to establish its state based on:

* The position relative to the parked cars;
* The distance between the car and the curbstone.

1. LED State ALGO [TBD]

The car will process information to establish its state based on the former system states.

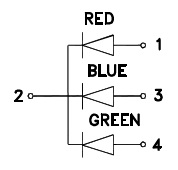
The ultrasonic sensor works by transmitting an ultrasonic burst and providing an output pulse that corresponds to the time required the burst echo to return to the sensor. By measuring the echo pulse width, the distance to target can be easily calculated.

The infrared sensor works by [**TBD**]

The RGB LED consists of three white diffused lens.

The L-154A4SUREQBFZGEW is a 5mm full-colour RGB LED Lamp with white diffused lens. The hyper red source device is made with AlGaInP on GaAs substrate. The blue source device is made with InGaN. The green source device is made with InGaN on sapphire. In our application we are using only the red and the green LEDs.





When the system is turned ON, the RGB LED will have the following states:

* 1. RED – moving forward, no parking place found - State 1, 2, 3 or 4
  2. Blinking GREEN – parking place found, parking started – State 5
  3. GREEN – parking done

The resistors for he LEDs where chosen as 220 ohm for the red one and 120 ohm for the green one for the forward current to be less then 20mA (14mA for the red LED and 16mA for the green LED).

# Public interface

This chapter contains the public interface.

|  |  |  |
| --- | --- | --- |
| void v\_setPinMode(void); | | |
|  | Description | Sets as INPUT or OUTPUT all the used pins. |
|  | Parameters | None |
|  | Return | None |

|  |  |  |
| --- | --- | --- |
| void v\_setupArdumoto (void); | | |
|  | Description | Intializes all the motor control pins as low |
|  | Parameters | None |
|  | Return | None |

|  |  |  |
| --- | --- | --- |
| void v\_driveArdumoto (uint8 u8\_direction, uint8 u8\_speed); | | |
|  | Description | Sends the direction (front/back) and the speed for the traction control motor. |
|  | Parameters | The direction (0/1) and the speed value. |
|  | Return | None |

|  |  |  |
| --- | --- | --- |
| void v\_stopArdumoto (uint8 u8\_motor); | | |
|  | Description | Stops a specified motor |
|  | Parameters | The motor that needs to be stopped (MOTOR\_A/MOTOR\_B) |
|  | Return | None |

|  |  |  |
| --- | --- | --- |
| void v\_MotorB\_left (void); | | |
|  | Description | Sets the direction motor at maximum left. |
|  | Parameters | None |
|  | Return | None |

|  |  |  |
| --- | --- | --- |
| void v\_MotorB\_right (void); | | |
|  | Description | Sets the direction motor at maximum right. |
|  | Parameters | None |
|  | Return | None |

|  |  |  |
| --- | --- | --- |
| float f\_getDistanceFromUltrasonic (void); | | |
|  | Description |  |
|  | Parameters |  |
|  | Return |  |

|  |  |  |
| --- | --- | --- |
| float f\_getDistanceFromInfrared (uint8 u8\_infraredSensor); | | |
|  | Description |  |
|  | Parameters |  |
|  | Return |  |

|  |  |  |
| --- | --- | --- |
| void v\_TimerConfig(); | | |
|  | Description | Configures the timer in respect to the functional overview. |
|  | Parameters | None |
|  | Return | None |

# Data structures

uint32 u32\_osSystemCounter – Operating time from power up in milliseconds

uint16 u16\_digital\_IR\_front – Digital value read from the 1st infrared sensor

uint16 u16\_digital\_IR\_rear – Digital value read from the 2nd infrared sensor

uint16 u16\_time\_ultrasonic - Time in microseconds used to calculate distance

uint8 MOTOR\_A – Instantiates the traction control motor

uint8 MOTOR\_B – Instantiates the direction motor

uint8 CW – Direction for the motors (clockwise/right)

uint8 CCW – Direction for the motors (counterclockwise/left)

# Defines

TBD

# Internal functions

This chapter contains the internal functions and commands.

## Functions

## Debug Commands

# Resource usage

# Requirements

|  |  |
| --- | --- |
| Requirement ID | Implemented [Y/N] |
|  |  |