El Tóxico

Generated by Doxygen 1.9.1

Chapter 1

TMR

Stable branch.

On this branch'll be all the fancy code to compile and run, so this branch will not be updated with the lasts algorithms, only with the stable ones.

Disclaimer.

Even this repo has the more stable and fancy scripts, algorithms and codes, these aren't the competition version, so the parameters may change, but i'll try to indicate which parameters you must change and pay special attention.

Hardware Requeriments.

The hardware where the robot car run, and is designed to run, has a large list of changes, but i'll put here the more actual hardware (stable only, obviously).
So, the current list is the next.

- Raspberry pi 4 Model B Rev 1.4 8GB.
- Roboclaw 2x30A.
- Servomotor Hexfly 60Kg.
- Redcat Racing Lighting Exp Drift (only the chasis).
- Personalized 3D printed parts.

Software Requeriments.

The software requeriments're more complicated to write correctly, because the current OS has a lot of installed packages, currently we haven't a lis from all this packages, configurations or something like that, tat's because I'll write only the main dependences, but you could open a request, discussion or something like that to help me to make more accurate this list.

- Ubuntu 22.04.3 LTS
- Ros2 Iron
- OpenCv 4.8.1

Relevant pkgs.

Robot history.

Along the project lifecycle, there are many authors, so I'll try to make a timeline alongside all the project, since the main team to the current team.

- 2020
  - The project start on the first months (I know the first commit has June version), the team was:
  - `Robles, Héctor`. (Team leader)
  - `Ruíz, Iñaki`. (Mechanic deisgner)
  - `Solano, Jorge`. (Team responsable)
- 2021
  - Ruíz, Iñaki lefted to team.
  - `Romero, Melania` (Programmer) joined to team.
  - `Negrete, Marco` (Technical guide) joined to team
- 2022
  - This's the first year which the robot's inscribed to the TMR, but on the remote simulated league, anyway, we won the `3rd` place.
  - Negrete, Marco helped us to join to `biorobotics lab` on the `Facultad de Ingeniería, UNAM`.
  - `Guerrero, Daniel` (Programmer) helped us and share his ideas with us.
  - `Rosario, Omar` (Mechanical designer) joined the team.
  - Delgado, Emilio (Programmer) joined to team.
  - Marín, Gustavo (Programmer) joined to team.
- 2023
  - This's the first year we could run on the phisical tournament, we won the `1st` place.
- 2024
  - Delgado, Emilio lefted to team.
  - Marín, Gustavo lefted to team.
  - Guerrero, Daniel lefted to team
  - `Vasquez, Jair` (Helper) Joined the team.

Current team.

The current team's pefrormed by.

- Negrte, Marco (Technical guide)
- Robles, Héctor (Team Leader)
- Solano, Jorge (Team responsable)

Sponsors.

Currently we don't have more sponsors than de UNAM, but if you want to become a sponsor to this project, you cand sendus an e-mail to the next emails.

- Offficial Mail (pumasautomodelcar@gmail.com)
- Negrete, Marco (marco.negrete@ingenieria.unam.edu)
- Robles, Hector (robletes062901@gmail.com)
- Solano, Jorge (jorge.solano@ingenieria.unam.edu)

Or anyother mail in our github's profiles. You can check our porpouse on the sponsor's presentation.

Team core.

The team core's performed by.

- Negrte, Marco (Technical guide)
- Robles, Héctor (Team Leader)
- Solano, Jorge (Team responsable)

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 blinkers_interface Namespace Reference

Classes

- class BlinkersInterface

Functions

- def main (args=None)

Variables

- gpio_pin = lgpio.gpiochip_open(0)

6.1.1 Function Documentation

6.1.1.1 main()

```
def blinkers_interface.main (
                args = None )
```

```
46 def main(args=None):
47     rclpy.init(args=args)
48     blinkersInterface = BlinkersInterface()
49     rclpy.spin(blinkersInterface)
50     blinkersInterface.destroy_node()
51     rclpy.shutdown()
52
```

6.1.2 Variable Documentation

6.1.2.1 gpio_pin

```
gpio_pin = lgpio.gpiochip_open(0)
```

6.2 controller Namespace Reference

Classes

- class ControlSubscriber

Functions

- def main (args=None)

Variables

- interface = lgpio.gpiochip_open(0)
- roboclaw = Roboclaw("/dev/ttyACM0", 115200)

6.2.1 Function Documentation

6.2.1.1 main()

```
def controller.main (
                args = None )
40 def main(args=None):
41     rclpy.init(args=args)
42     control_subscriber = ControlSubscriber()
43     rclpy.spin(control_subscriber)
44     control_subscriber.destroy_node()
45     rclpy.shutdown()
46
```

6.2.2 Variable Documentation

6.2.2.1 interface

```
interface = lgpio.gpiochip_open(0)
```

6.2.2.2 roboclaw

```
roboclaw = Roboclaw("/dev/ttyACM0", 115200)
```

6.3 motor_interface Namespace Reference

Classes

- class MotorInterface

Functions

- def main (args=None)

Variables

- roboclaw = Roboclaw("/dev/ttyACM0", 115200)

6.3.1 Function Documentation

6.3.1.1 main()

```
def motor_interface.main (
                args = None )
47 def main(args=None):
48     print(roboclaw.Open())
49     rclpy.init(args=args)
50     motorInterface = MotorInterface()
51     rclpy.spin(motorInterface)
52     motorInterface.destroy_node()
53     rclpy.shutdown()
54
```

6.3.2 Variable Documentation

6.3.2.1 roboclaw

```
roboclaw = Roboclaw("/dev/ttyACM0", 115200)
```

6.4 oled_interface Namespace Reference

Classes

- class OledInterface

Functions

- def main (args=None)

Variables

- RST = None
- int DC = 23
- int SPI_PORT = 0
- int SPI_DEVICE = 0
- disp = Adafruit_SSD1306.SSD1306_128_64(rst=RST)
- width = disp.width
- height = disp.height
- image = Image.new('1', (width, height))
- draw = ImageDraw.Draw(image)
- outline
- fill
- int padding = -2
- int top = padding
- int bottom = height-padding
- int x = 0
- font = ImageFont.load_default()

6.4.1 Function Documentation

6.4.1.1 main()

```
def oled_interface.main (
                args = None )
```

```
56 def main(args=None):
57     rclpy.init(args=args)
58     oledInterface = OledInterface()
59     rclpy.spin(oledInterface)
60     oledInterface.destroy_node()
61     rclpy.shutdown()
62
```

6.4.2 Variable Documentation

6.4.2.1 bottom

```
int bottom = height-padding
```

6.4.2.2 DC

```
int DC = 23
```

6.4.2.3 disp

```
disp = Adafruit_SSD1306.SSD1306_128_64(rst=RST)
```

6.4.2.4 draw

```
draw = ImageDraw.Draw(image)
```

6.4.2.5 fill

```
fill
```

6.4.2.6 font

```
font = ImageFont.load_default()
```

6.4.2.7 height

```
height = disp.height
```

6.4.2.8 image

```
image = Image.new('1', (width, height))
```

6.4.2.9 outline

```
outline
```

6.4.2.10 padding

```
int padding = -2
```

6.4.2.11 RST

```
RST = None
```

6.4.2.12 SPI_DEVICE

```
int SPI_DEVICE = 0
```

6.4.2.13 SPI_PORT

```
int SPI_PORT = 0
```

6.4.2.14 top

```
int top = padding
```

6.4.2.15 width

```
width = disp.width
```

6.4.2.16 x

```
int x = 0
```

6.5 roboclaw_3 Namespace Reference

Classes

- class Roboclaw

6.6 servo_interface Namespace Reference

Classes

- class SteeringInterface

Functions

- def main (args=None)

Variables

- servo_pin = lgpio.gpiochip_open(0)

6.6.1 Function Documentation

6.6.1.1 main()

```
def servo_interface.main (
                args = None )
```

```
33 def main(args=None):
34     rclpy.init(args=args)
35     steeringInterface = SteeringInterface()
36     rclpy.spin(steeringInterface)
37     steeringInterface.destroy_node()
38     rclpy.shutdown()
39
```

6.6.2 Variable Documentation

6.6.2.1 servo_pin

```
servo_pin = lgpio.gpiochip_open(0)
```

6.7 setup Namespace Reference

Variables

- string package_name = 'toxic_hardware'
- name
- version
- packages
- data_files
- install_requires
- zip_safe
- maintainer
- maintainer_email
- description
- license
- tests_require
- entry_points

6.7.1 Variable Documentation

6.7.1.1 data_files

```
data_files
```

6.7.1.2 description

```
description
```

6.7.1.3 entry_points

```
entry_points
```

6.7.1.4 install_requires

```
install_requires
```

6.7.1.5 license

```
license
```

6.7.1.6 maintainer

```
maintainer
```

6.7.1.7 maintainer_email

```
maintainer_email
```

6.7.1.8 name

```
name
```

6.7.1.9 package_name

```
string package_name = 'toxic_hardware'
```

6.7.1.10 packages

```
packages
```

6.7.1.11 tests_require

```
tests_require
```

6.7.1.12 version

```
version
```

6.7.1.13 zip_safe

```
zip_safe
```

6.8 webcam_pub Namespace Reference

Classes

- class ImagePublisher

Functions

- def main (args=None)

  This function create the publisher node, the image publisher object and starts to run the publisher.

6.8.1 Function Documentation

6.8.1.1 main()

```
def webcam_pub.main (
              args = None )
```

This function create the publisher node, the image publisher object and starts to run the publisher.

```
102 def main(args=None):
103
107     rclpy.init(args=args)
108     image_publisher = ImagePublisher()
109     rclpy.spin(image_publisher)
110     image_publisher.destroy_node()
111     rclpy.shutdown()
112
```

6.9 webcam_sub Namespace Reference

Classes

- class ImageSubscriber

Functions

- def main (args=None)

  This function create the subscriber and execute it.

6.9.1 Function Documentation

6.9.1.1 main()

```
def webcam_sub.main (
              args = None )
```

This function create the subscriber and execute it.

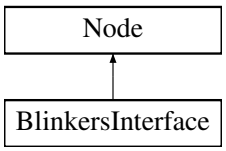
```
92 def main(args=None):
93
96
97     rclpy.init(args=args)
98     image_subscriber = ImageSubscriber()
99     rclpy.spin(image_subscriber)
100     image_subscriber.destroy_node()
101     rclpy.shutdown()
102
```

Chapter 7

Class Documentation

7.1 BlinkersInterface Class Reference

Inheritance diagram for BlinkersInterface:



Public Member Functions

- def __init__ (self)
- def blinkers_callback (self, data)

Public Attributes

- subscription

7.1.1 Constructor & Destructor Documentation

7.1.1.1 __init__()

```
def __init__ (
                self )
11      def __init__(self):
12          super().__init__('blinkers_interface')
13          self.subscription = self.create_subscription(
14                  Int8,
15                  '/blinkers',
16                  self.blinkers_callback,
17                  2
18                  )
19          self.subscription
20
```

7.1.2 Member Function Documentation

7.1.2.1 blinkers_callback()

```
def blinkers_callback (
                self,
                data )
```

```
21     def blinkers_callback(self, data):
22         global gpio_pin
23         recived = data.data
24         if recived == 0:
25             lgpio.gpio_write(gpio_pin, 17, 0)
26             lgpio.gpio_write(gpio_pin, 27, 0)
27         elif recived == 1:
28             lgpio.gpio_write(gpio_pin, 17, 1)
29             time.sleep(0.5)
30             lgpio.gpio_write(gpio_pin, 17, 0)
31             time.sleep(0.5)
32         elif recived == -1:
33             lgpio.gpio_write(gpio_pin, 27, 1)
34             time.sleep(0.5)
35             lgpio.gpio_write(gpio_pin, 27, 0)
36             time.sleep(0.5)
37         else:
38             lgpio.gpio_write(gpio_pin, 17, 1)
39             lgpio.gpio_write(gpio_pin, 27, 1)
40             time.sleep(0.5)
41             lgpio.gpio_write(gpio_pin, 17, 0)
42             lgpio.gpio_write(gpio_pin, 27, 0)
43             time.sleep(0.5)
44
45
```

7.1.3 Member Data Documentation

7.1.3.1 subscription

```
subscription
```

The documentation for this class was generated from the following file:

- main_ws/src/toxic_hardware/toxic_hardware/blinkers_interface.py

7.2 Roboclaw.Cmd Class Reference

Static Public Attributes

- int M1FORWARD = 0
- int M1BACKWARD = 1
- int SETMINMB = 2
- int SETMAXMB = 3
- int M2FORWARD = 4
- int M2BACKWARD = 5

- int M17BIT = 6
- int M27BIT = 7
- int MIXEDFORWARD = 8
- int MIXEDBACKWARD = 9
- int MIXEDRIGHT = 10
- int MIXEDLEFT = 11
- int MIXEDFB = 12
- int MIXEDLR = 13
- int GETM1ENC = 16
- int GETM2ENC = 17
- int GETM1SPEED = 18
- int GETM2SPEED = 19
- int RESETENC = 20
- int GETVERSION = 21
- int SETM1ENCCOUNT = 22
- int SETM2ENCCOUNT = 23
- int GETMBATT = 24
- int GETLBATT = 25
- int SETMINLB = 26
- int SETMAXLB = 27
- int SETM1PID = 28
- int SETM2PID = 29
- int GETM1ISPEED = 30
- int GETM2ISPEED = 31
- int M1DUTY = 32
- int M2DUTY = 33
- int MIXEDDUTY = 34
- int M1SPEED = 35
- int M2SPEED = 36
- int MIXEDSPEED = 37
- int M1SPEEDACCEL = 38
- int M2SPEEDACCEL = 39
- int MIXEDSPEEDACCEL = 40
- int M1SPEEDDIST = 41
- int M2SPEEDDIST = 42
- int MIXEDSPEEDDIST = 43
- int M1SPEEDACCELDIST = 44
- int M2SPEEDACCELDIST = 45
- int MIXEDSPEEDACCELDIST = 46
- int GETBUFFERS = 47
- int GETPWMS = 48
- int GETCURRENTS = 49
- int MIXEDSPEED2ACCEL = 50
- int MIXEDSPEED2ACCELDIST = 51
- int M1DUTYACCEL = 52
- int M2DUTYACCEL = 53
- int MIXEDDUTYACCEL = 54
- int READM1PID = 55
- int READM2PID = 56
- int SETMAINVOLTAGES = 57
- int SETLOGICVOLTAGES = 58
- int GETMINMAXMAINVOLTAGES = 59
- int GETMINMAXLOGICVOLTAGES = 60
- int SETM1POSPID = 61
- int SETM2POSPID = 62

- int READM1POSPID = 63
- int READM2POSPID = 64
- int M1SPEEDACCELDECCELPOS = 65
- int M2SPEEDACCELDECCELPOS = 66
- int MIXEDSPEEDACCELDECCELPOS = 67
- int SETM1DEFAULTACCEL = 68
- int SETM2DEFAULTACCEL = 69
- int SETPINFUNCTIONS = 74
- int GETPINFUNCTIONS = 75
- int SETDEADBAND = 76
- int GETDEADBAND = 77
- int RESTOREDEFAULTS = 80
- int GETTEMP = 82
- int GETTEMP2 = 83
- int GETERROR = 90
- int GETENCODERMODE = 91
- int SETM1ENCODERMODE = 92
- int SETM2ENCODERMODE = 93
- int WRITENVM = 94
- int READNVM = 95
- int SETCONFIG = 98
- int GETCONFIG = 99
- int SETM1MAXCURRENT = 133
- int SETM2MAXCURRENT = 134
- int GETM1MAXCURRENT = 135
- int GETM2MAXCURRENT = 136
- int SETPWMMODE = 148
- int GETPWMMODE = 149
- int READEEPROM = 252
- int WRITEEEPROM = 253
- int FLAGBOOTLOADER = 255

7.2.1 Member Data Documentation

7.2.1.1 FLAGBOOTLOADER

```
int FLAGBOOTLOADER = 255  [static]
```

7.2.1.2 GETBUFFERS

```
int GETBUFFERS = 47  [static]
```

7.2.1.3 GETCONFIG

```
int GETCONFIG = 99  [static]
```

7.2.1.4 GETCURRENTS

```
int GETCURRENTS = 49  [static]
```

7.2.1.5 GETDEADBAND

```
int GETDEADBAND = 77  [static]
```

7.2.1.6 GETENCODERMODE

```
int GETENCODERMODE = 91  [static]
```

7.2.1.7 GETERROR

```
int GETERROR = 90  [static]
```

7.2.1.8 GETLBATT

```
int GETLBATT = 25  [static]
```

7.2.1.9 GETM1ENC

```
int GETM1ENC = 16  [static]
```

7.2.1.10 GETM1ISPEED

```
int GETM1ISPEED = 30  [static]
```

7.2.1.11 GETM1MAXCURRENT

```
int GETM1MAXCURRENT = 135  [static]
```

7.2.1.12 GETM1SPEED

```
int GETM1SPEED = 18  [static]
```

7.2.1.13 GETM2ENC

```
int GETM2ENC = 17  [static]
```

7.2.1.14 GETM2ISPEED

```
int GETM2ISPEED = 31  [static]
```

7.2.1.15 GETM2MAXCURRENT

```
int GETM2MAXCURRENT = 136  [static]
```

7.2.1.16 GETM2SPEED

```
int GETM2SPEED = 19  [static]
```

7.2.1.17 GETMBATT

```
int GETMBATT = 24  [static]
```

7.2.1.18 GETMINMAXLOGICVOLTAGES

```
int GETMINMAXLOGICVOLTAGES = 60  [static]
```

7.2.1.19 GETMINMAXMAINVOLTAGES

```
int GETMINMAXMAINVOLTAGES = 59  [static]
```

7.2.1.20 GETPINFUNCTIONS

```
int GETPINFUNCTIONS = 75  [static]
```

7.2.1.21 GETPWMMODE

```
int GETPWMMODE = 149  [static]
```

7.2.1.22 GETPWMS

```
int GETPWMS = 48  [static]
```

7.2.1.23 GETTEMP

```
int GETTEMP = 82  [static]
```

7.2.1.24 GETTEMP2

```
int GETTEMP2 = 83  [static]
```

7.2.1.25 GETVERSION

```
int GETVERSION = 21  [static]
```

7.2.1.26 M17BIT

```
int M17BIT = 6  [static]
```

7.2.1.27 M1BACKWARD

```
int M1BACKWARD = 1  [static]
```

7.2.1.28 M1DUTY

```
int M1DUTY = 32  [static]
```

7.2.1.29 M1DUTYACCEL

```
int M1DUTYACCEL = 52  [static]
```

7.2.1.30 M1FORWARD

```
int M1FORWARD = 0  [static]
```

7.2.1.31 M1SPEED

```
int M1SPEED = 35  [static]
```

7.2.1.32 M1SPEEDACCEL

```
int M1SPEEDACCEL = 38  [static]
```

7.2.1.33 M1SPEEDACCELDECCELPOS

```
int M1SPEEDACCELDECCELPOS = 65  [static]
```

7.2.1.34 M1SPEEDACCELDIST

```
int M1SPEEDACCELDIST = 44  [static]
```

7.2.1.35 M1SPEEDDIST

```
int M1SPEEDDIST = 41  [static]
```

7.2.1.36 M27BIT

```
int M27BIT = 7  [static]
```

7.2.1.37 M2BACKWARD

```
int M2BACKWARD = 5  [static]
```

7.2.1.38 M2DUTY

```
int M2DUTY = 33  [static]
```

7.2.1.39 M2DUTYACCEL

```
int M2DUTYACCEL = 53  [static]
```

7.2.1.40 M2FORWARD

```
int M2FORWARD = 4  [static]
```

7.2.1.41 M2SPEED

```
int M2SPEED = 36  [static]
```

7.2.1.42 M2SPEEDACCEL

```
int M2SPEEDACCEL = 39  [static]
```

7.2.1.43 M2SPEEDACCELDECCELPOS

```
int M2SPEEDACCELDECCELPOS = 66  [static]
```

7.2.1.44 M2SPEEDACCELDIST

```
int M2SPEEDACCELDIST = 45  [static]
```

7.2.1.45 M2SPEEDDIST

```
int M2SPEEDDIST = 42  [static]
```

7.2.1.46 MIXEDBACKWARD

```
int MIXEDBACKWARD = 9  [static]
```

7.2.1.47 MIXEDDUTY

```
int MIXEDDUTY = 34  [static]
```

7.2.1.48 MIXEDDUTYACCEL

```
int MIXEDDUTYACCEL = 54  [static]
```

7.2.1.49 MIXEDFB

```
int MIXEDFB = 12  [static]
```

7.2.1.50 MIXEDFORWARD

```
int MIXEDFORWARD = 8  [static]
```

7.2.1.51 MIXEDLEFT

```
int MIXEDLEFT = 11  [static]
```

7.2.1.52 MIXEDLR

```
int MIXEDLR = 13  [static]
```

7.2.1.53 MIXEDRIGHT

```
int MIXEDRIGHT = 10  [static]
```

7.2.1.54 MIXEDSPEED

```
int MIXEDSPEED = 37  [static]
```

7.2.1.55 MIXEDSPEED2ACCEL

```
int MIXEDSPEED2ACCEL = 50  [static]
```

7.2.1.56 MIXEDSPEED2ACCELDIST

```
int MIXEDSPEED2ACCELDIST = 51  [static]
```

7.2.1.57 MIXEDSPEEDACCEL

```
int MIXEDSPEEDACCEL = 40  [static]
```

7.2.1.58 MIXEDSPEEDACCELDECCELPOS

```
int MIXEDSPEEDACCELDECCELPOS = 67  [static]
```

7.2.1.59 MIXEDSPEEDACCELDIST

```
int MIXEDSPEEDACCELDIST = 46  [static]
```

7.2.1.60 MIXEDSPEEDDIST

```
int MIXEDSPEEDDIST = 43  [static]
```

7.2.1.61 READEEPROM

```
int READEEPROM = 252  [static]
```

7.2.1.62 READM1PID

```
int READM1PID = 55  [static]
```

7.2.1.63 READM1POSPID

```
int READM1POSPID = 63  [static]
```

7.2.1.64 READM2PID

```
int READM2PID = 56  [static]
```

7.2.1.65 READM2POSPID

```
int READM2POSPID = 64  [static]
```

7.2.1.66 READNVM

```
int READNVM = 95  [static]
```

7.2.1.67 RESETENC

```
int RESETENC = 20  [static]
```

7.2.1.68 RESTOREDEFAULTS

```
int RESTOREDEFAULTS = 80  [static]
```

7.2.1.69 SETCONFIG

```
int SETCONFIG = 98  [static]
```

7.2.1.70 SETDEADBAND

```
int SETDEADBAND = 76  [static]
```

7.2.1.71 SETLOGICVOLTAGES

```
int SETLOGICVOLTAGES = 58  [static]
```

7.2.1.72 SETM1DEFAULTACCEL

```
int SETM1DEFAULTACCEL = 68  [static]
```

7.2.1.73 SETM1ENCCOUNT

```
int SETM1ENCCOUNT = 22  [static]
```

7.2.1.74 SETM1ENCODERMODE

```
int SETM1ENCODERMODE = 92  [static]
```

7.2.1.75 SETM1MAXCURRENT

```
int SETM1MAXCURRENT = 133  [static]
```

7.2.1.76 SETM1PID

```
int SETM1PID = 28  [static]
```

7.2.1.77 SETM1POSPID

```
int SETM1POSPID = 61  [static]
```

7.2.1.78 SETM2DEFAULTACCEL

```
int SETM2DEFAULTACCEL = 69  [static]
```

7.2.1.79 SETM2ENCCOUNT

```
int SETM2ENCCOUNT = 23  [static]
```

7.2.1.80 SETM2ENCODERMODE

```
int SETM2ENCODERMODE = 93  [static]
```

7.2.1.81 SETM2MAXCURRENT

```
int SETM2MAXCURRENT = 134  [static]
```

7.2.1.82 SETM2PID

```
int SETM2PID = 29  [static]
```

7.2.1.83 SETM2POSPID

```
int SETM2POSPID = 62  [static]
```

7.2.1.84 SETMAINVOLTAGES

```
int SETMAINVOLTAGES = 57  [static]
```

7.2.1.85 SETMAXLB

```
int SETMAXLB = 27  [static]
```

7.2.1.86 SETMAXMB

```
int SETMAXMB = 3  [static]
```

7.2.1.87 SETMINLB

```
int SETMINLB = 26  [static]
```

7.2.1.88 SETMINMB

```
int SETMINMB = 2  [static]
```

7.2.1.89 SETPINFUNCTIONS

```
int SETPINFUNCTIONS = 74  [static]
```

7.2.1.90 SETPWMMODE

```
int SETPWMMODE = 148  [static]
```

7.2.1.91 WRITEEEPROM

```
int WRITEEEPROM = 253  [static]
```

7.2.1.92 WRITENVM

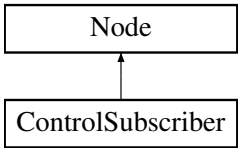
```
int WRITENVM = 94  [static]
```

The documentation for this class was generated from the following file:

- main_ws/src/toxic_hardware/toxic_hardware/roboclaw_3.py

7.3 ControlSubscriber Class Reference

Inheritance diagram for ControlSubscriber:



Public Member Functions

- def __init__ (self)
- def control_callback (self, data)

Public Attributes

- subscription
- speed_publisher
- steering_publisher

7.3.1 Constructor & Destructor Documentation

7.3.1.1 __init__()

```
def __init__ (
                self )
```

```
17     def __init__(self):
18         super().__init__('control_toxic_subscriber')
19         self.subscription = self.create_subscription(
20                 Joy,
21                 '/joy',
22                 self.control_callback,
23                 60
24                 )
25         self.speed_publisher = self.create_publisher(Float64, '/speed', 60)
26         self.steering_publisher = self.create_publisher(Float64, '/steering', 60)
27         self.subscription
28         self.steering_publisher
29         self.speed_publisher
30
```

7.3.2 Member Function Documentation

7.3.2.1 control_callback()

```
def control_callback (
                self,
                data )
```

```
31     def control_callback(self, data):
32         normalized_steering = data.axes[0]
33         normalized_fw_speed = data.axes[5]
34         normalized_bw_speed = data.axes[2]
35         print(normalized_steering)
36         print(normalized_fw_speed)
37         self.steering_publisher.publish(Float64(normalized_steering))
38         self.speed_publisher.publish(Float64(normalized_fw_speed + normalized_bw_speed))
39
```

7.3.3 Member Data Documentation

7.3.3.1 speed_publisher

```
speed_publisher
```

7.3.3.2 steering_publisher

```
steering_publisher
```

7.3.3.3 subscription

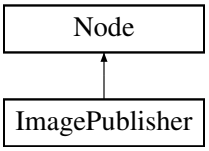
```
subscription
```

The documentation for this class was generated from the following file:

- main_ws/src/toxic_hardware/toxic_hardware/controller.py

7.4 ImagePublisher Class Reference

Inheritance diagram for ImagePublisher:



Public Member Functions

- def __init__ (self)

  ImagePublisher object to grab the live camera image and publish to a ROS2 Node as CvBridge message type.
- def timer_callback (self)

  Timer callback to publish the image, everytime the timer achieve the desired time, this callback'll run.

Public Attributes

- publisher_
- timer
- cap
- br

7.4.1 Constructor & Destructor Documentation

7.4.1.1 __init__()

```
def __init__ (
              self )
```

ImagePublisher object to grab the live camera image and publish to a ROS2 Node as CvBridge message type.

Parameters

Node	The ROS2 Node where the image publisher'll be able to read/write (this time, just write)

Returns

None, keeps running and alive while the camera's oppened.

ImagePublisher object init def (or constructor function, under a Java's OOP context).

Parameters

self	Self contained object like a 'this' reference, just to read, write, and generally access object's attributes.

Returns

ImagePublisherNewObject returns a new image publisher object.

@detail Creates the publisher node, set the timer, open the camera, set some needed params like image shape and framerate, open the bridge between OpenCv and ROS2.

```
60     def __init__(self):
61
74
75         super().__init__('image_publisher')
76         self.publisher_ = self.create_publisher(Image, '/raw_rgb', 1)
77         timer_period = 0.033
78         self.timer = self.create_timer(timer_period, self.timer_callback)
79         self.cap = cv2.VideoCapture(0)
80         self.cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
81         self.cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)
82         self.cap.set(cv2.CAP_PROP_FPS, 30)
83         self.br = CvBridge()
84
```

7.4.2 Member Function Documentation

7.4.2.1 timer_callback()

```
def timer_callback (
              self )
```

Timer callback to publish the image, everytime the timer achieve the desired time, this callback'll run.

Parameters

self	Self contained object like a 'this' reference, jus to read, write, and generally access object's attributes.

Returns

None, just publish the current frame.

@detail Not much, just validate if there's a new image and publish it

```
85     def timer_callback(self):
86
97
98         ret, frame = self.cap.read()
99         if ret:
100            self.publisher_.publish(self.br.cv2_to_imgmsg(frame))
101
```

7.4.3 Member Data Documentation

7.4.3.1 br

`br`

7.4.3.2 cap

`cap`

7.4.3.3 publisher_

`publisher_`

7.4.3.4 timer

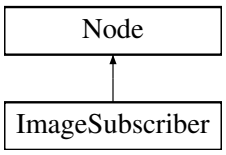
`timer`

The documentation for this class was generated from the following file:

- main_ws/src/toxic_vision/toxic_vision/webcam_pub.py

7.5 ImageSubscriber Class Reference

Inheritance diagram for ImageSubscriber:



Public Member Functions

- def __init__ (self)

  ImageSubscriber object to grab the currently published frame in a ROS2 Topic (this time, it's '/raw_rgb', but you can change it) and display in a GUI window.
- def listener_callback (self, data)

  Listener callback to grab the first buffer's image and process it.

Public Attributes

- subscription
- br

7.5.1 Constructor & Destructor Documentation

7.5.1.1 __init__()

```
def __init__ (
              self )
```

ImageSubscriber object to grab the currently published frame in a ROS2 Topic (this time, it's '/raw_rgb', but you can change it) and display in a GUI window.

Parameters

Node	The ROS2 Node where the image publisher'll be able to read/write (this time, just read)

Returns

None, keeps running and alive until it's user cancelled.

ImageSubscriber obtect init def (or constructor, under a Java´s OOP context)

Parameters

self,Self	contained object, like a 'this' reference just to read, write and genereally access object's attributes.

@detail creates the image subscriber and configure it to run.

```
55     def __init__(self):
56
65
66         super().__init__('image_subscriber')
67         self.subscription = self.create_subscription(
68                 Image,
69                 '/raw_rgb',
70                 self.listener_callback,
71                 25
72                 )
73         self.subscription
74         self.br = CvBridge()
75
```

7.5.2 Member Function Documentation

7.5.2.1 listener_callback()

```
def listener_callback (
              self,
              data )
```

Listener callback to grab the first buffer's image and process it.

Parameters

self	Self contained object like a 'this' reference, just to read, write and generally access to object's attributes.
data	Current message (first buffer's data)

Returns

None, properly just displays the image.

```
76     def listener_callback(self, data):
77
88         current_frame = self.br.imgmsg_to_cv2(data)
89         cv2.imshow("band_filter_sub", current_frame)
90         cv2.waitKey(1)
```

7.5.3 Member Data Documentation

7.5.3.1 br

```
br
```

7.5.3.2 subscription

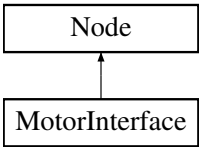
```
subscription
```

The documentation for this class was generated from the following file:

- main_ws/src/toxic_vision/toxic_vision/webcam_sub.py

7.6 MotorInterface Class Reference

Inheritance diagram for MotorInterface:



Public Member Functions

- def __init__ (self)
- def motor_callback (self, data)

Public Attributes

- subscription

7.6.1 Constructor & Destructor Documentation

7.6.1.1 __init__()

```
def __init__ (
               self )
```

```
21     def __init__(self):
22         super().__init__('motor_interface')
23         self.subscription = self.create_subscription(
24                 Float64,
25                 '/speed',
26                 self.motor_callback,
27                 60
28                 )
29         self.subscription
30
```

7.6.2 Member Function Documentation

7.6.2.1 motor_callback()

```
def motor_callback (
               self,
               data )
```

```
31     def motor_callback(self, data):
32         global roboclaw
33         recived = data.data
34         if recived > 1.0:
35             recived = 1.0
36         elif recived < -1.0:
37             recived = -1.0
38         if recived >= 0:
39             print("Forward@M1")
40             roboclaw.ForwardM1(0x80, int(32*recived))
41             print(recived)
42         elif recived < 0:
43             print("Backward@M1")
44             roboclaw.BackwardM1(0x80, int(32*-recived))
45             print(recived)
46
```

7.6.3 Member Data Documentation

7.6.3.1 subscription

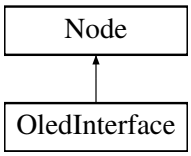
```
subscription
```

The documentation for this class was generated from the following file:

- main_ws/src/toxic_hardware/toxic_hardware/motor_interface.py

7.7 OledInterface Class Reference

Inheritance diagram for OledInterface:



Public Member Functions

- def __init__ (self)
- def oled_callback (self, data)

Public Attributes

- subscription

7.7.1 Constructor & Destructor Documentation

7.7.1.1 __init__()

```
def __init__ (
                self )
39     def __init__(self):
40         super().__init__('oled_interface')
41         self.subscription = self.create_subscription(
42                 String,
43                 '/message',
44                 self.oled_callback,
45                 60
46         )
47
```

7.7.2 Member Function Documentation

7.7.2.1 oled_callback()

```
def oled_callback (
                self,
                data )
```

```
48     def oled_callback(self, data):
49         global draw, image
50         draw.rectangle((0,0,width,height), outline=0, fill=0)
51         msg_content = data.data
52         draw.text((x, top), msg_content,  font=font, fill=255)
53         disp.image(image)
54         disp.display()
55
```

7.7.3 Member Data Documentation

7.7.3.1 subscription

```
subscription
```

The documentation for this class was generated from the following file:

- main_ws/src/toxic_hardware/toxic_hardware/oled_interface.py

7.8 Roboclaw Class Reference

Classes

- class Cmd

Public Member Functions

- def __init__ (self, comport, rate, timeout=0.01, retries=3)
- def crc_clear (self)
- def crc_update (self, data)
- def SendRandomData (self, cnt)
- def ForwardM1 (self, address, val)
- def BackwardM1 (self, address, val)
- def SetMinVoltageMainBattery (self, address, val)
- def SetMaxVoltageMainBattery (self, address, val)
- def ForwardM2 (self, address, val)
- def BackwardM2 (self, address, val)
- def ForwardBackwardM1 (self, address, val)
- def ForwardBackwardM2 (self, address, val)
- def ForwardMixed (self, address, val)
- def BackwardMixed (self, address, val)
- def TurnRightMixed (self, address, val)
- def TurnLeftMixed (self, address, val)
- def ForwardBackwardMixed (self, address, val)

- def LeftRightMixed (self, address, val)
- def ReadEncM1 (self, address)
- def ReadEncM2 (self, address)
- def ReadSpeedM1 (self, address)
- def ReadSpeedM2 (self, address)
- def ResetEncoders (self, address)
- def ReadVersion (self, address)
- def SetEncM1 (self, address, cnt)
- def SetEncM2 (self, address, cnt)
- def ReadMainBatteryVoltage (self, address)
- def ReadLogicBatteryVoltage (self, address)
- def SetMinVoltageLogicBattery (self, address, val)
- def SetMaxVoltageLogicBattery (self, address, val)
- def SetM1VelocityPID (self, address, p, i, d, qpps)
- def SetM2VelocityPID (self, address, p, i, d, qpps)
- def ReadISpeedM1 (self, address)
- def ReadISpeedM2 (self, address)
- def DutyM1 (self, address, val)
- def DutyM2 (self, address, val)
- def DutyM1M2 (self, address, m1, m2)
- def SpeedM1 (self, address, val)
- def SpeedM2 (self, address, val)
- def SpeedM1M2 (self, address, m1, m2)
- def SpeedAccelM1 (self, address, accel, speed)
- def SpeedAccelM2 (self, address, accel, speed)
- def SpeedAccelM1M2 (self, address, accel, speed1, speed2)
- def SpeedDistanceM1 (self, address, speed, distance, buffer)
- def SpeedDistanceM2 (self, address, speed, distance, buffer)
- def SpeedDistanceM1M2 (self, address, speed1, distance1, speed2, distance2, buffer)
- def SpeedAccelDistanceM1 (self, address, accel, speed, distance, buffer)
- def SpeedAccelDistanceM2 (self, address, accel, speed, distance, buffer)
- def SpeedAccelDistanceM1M2 (self, address, accel, speed1, distance1, speed2, distance2, buffer)
- def ReadBuffers (self, address)
- def ReadPWMs (self, address)
- def ReadCurrents (self, address)
- def SpeedAccelM1M2_2 (self, address, accel1, speed1, accel2, speed2)
- def SpeedAccelDistanceM1M2_2 (self, address, accel1, speed1, distance1, accel2, speed2, distance2, buffer)
- def DutyAccelM1 (self, address, accel, duty)
- def DutyAccelM2 (self, address, accel, duty)
- def DutyAccelM1M2 (self, address, accel1, duty1, accel2, duty2)
- def ReadM1VelocityPID (self, address)
- def ReadM2VelocityPID (self, address)
- def SetMainVoltages (self, address, min, max)
- def SetLogicVoltages (self, address, min, max)
- def ReadMinMaxMainVoltages (self, address)
- def ReadMinMaxLogicVoltages (self, address)
- def SetM1PositionPID (self, address, kp, ki, kd, kimax, deadzone, min, max)
- def SetM2PositionPID (self, address, kp, ki, kd, kimax, deadzone, min, max)
- def ReadM1PositionPID (self, address)
- def ReadM2PositionPID (self, address)
- def SpeedAccelDeccelPositionM1 (self, address, accel, speed, deccel, position, buffer)
- def SpeedAccelDeccelPositionM2 (self, address, accel, speed, deccel, position, buffer)
- def SpeedAccelDeccelPositionM1M2 (self, address, accel1, speed1, deccel1, position1, accel2, speed2, deccel2, position2, buffer)

- def SetM1DefaultAccel (self, address, accel)
- def SetM2DefaultAccel (self, address, accel)
- def SetPinFunctions (self, address, S3mode, S4mode, S5mode)
- def ReadPinFunctions (self, address)
- def SetDeadBand (self, address, min, max)
- def GetDeadBand (self, address)
- def RestoreDefaults (self, address)
- def ReadTemp (self, address)
- def ReadTemp2 (self, address)
- def ReadError (self, address)
- def ReadEncoderModes (self, address)
- def SetM1EncoderMode (self, address, mode)
- def SetM2EncoderMode (self, address, mode)
- def WriteNVM (self, address)
- def ReadNVM (self, address)
- def SetConfig (self, address, config)
- def GetConfig (self, address)
- def SetM1MaxCurrent (self, address, max)
- def SetM2MaxCurrent (self, address, max)
- def ReadM1MaxCurrent (self, address)
- def ReadM2MaxCurrent (self, address)
- def SetPWMMode (self, address, mode)
- def ReadPWMMode (self, address)
- def ReadEeprom (self, address, ee_address)
- def WriteEeprom (self, address, ee_address, ee_word)
- def Open (self)

Public Attributes

- comport
- rate
- timeout

Private Member Functions

- def _sendcommand (self, address, command)
- def _readchecksumword (self)
- def _readbyte (self)
- def _readword (self)
- def _readlong (self)
- def _readslong (self)
- def _writebyte (self, val)
- def _writesbyte (self, val)
- def _writeword (self, val)
- def _writesword (self, val)
- def _writelong (self, val)
- def _writeslong (self, val)
- def _read1 (self, address, cmd)
- def _read2 (self, address, cmd)
- def _read4 (self, address, cmd)
- def _read4_1 (self, address, cmd)
- def _read_n (self, address, cmd, args)
- def _writechecksum (self)

- def _write0 (self, address, cmd)
- def _write1 (self, address, cmd, val)
- def _write11 (self, address, cmd, val1, val2)
- def _write111 (self, address, cmd, val1, val2, val3)
- def _write2 (self, address, cmd, val)
- def _writeS2 (self, address, cmd, val)
- def _write22 (self, address, cmd, val1, val2)
- def _writeS22 (self, address, cmd, val1, val2)
- def _writeS2S2 (self, address, cmd, val1, val2)
- def _writeS24 (self, address, cmd, val1, val2)
- def _writeS24S24 (self, address, cmd, val1, val2, val3, val4)
- def _write4 (self, address, cmd, val)
- def _writeS4 (self, address, cmd, val)
- def _write44 (self, address, cmd, val1, val2)
- def _write4S4 (self, address, cmd, val1, val2)
- def _writeS4S4 (self, address, cmd, val1, val2)
- def _write441 (self, address, cmd, val1, val2, val3)
- def _writeS441 (self, address, cmd, val1, val2, val3)
- def _write4S4S4 (self, address, cmd, val1, val2, val3)
- def _write4S441 (self, address, cmd, val1, val2, val3, val4)
- def _write4444 (self, address, cmd, val1, val2, val3, val4)
- def _write4S44S4 (self, address, cmd, val1, val2, val3, val4)
- def _write44441 (self, address, cmd, val1, val2, val3, val4, val5)
- def _writeS44S441 (self, address, cmd, val1, val2, val3, val4, val5)
- def _write4S44S441 (self, address, cmd, val1, val2, val3, val4, val5, val6)
- def _write4S444S441 (self, address, cmd, val1, val2, val3, val4, val5, val6, val7)
- def _write4444444 (self, address, cmd, val1, val2, val3, val4, val5, val6, val7)
- def _write444444441 (self, address, cmd, val1, val2, val3, val4, val5, val6, val7, val8, val9)

Private Attributes

- _trystimeout
- _crc
- _port

7.8.1 Constructor & Destructor Documentation

7.8.1.1 __init__()

```
def __init__ (
               self,
               comport,
               rate,
               timeout = 0.01,
               retries = 3 )
```

```
9     def __init__(self, comport, rate, timeout=0.01, retries=3):
10         self.comport = comport
11         self.rate = rate
12         self.timeout = timeout;
13         self._trystimeout = retries
14         self._crc = 0;
15
```

7.8.2 Member Function Documentation

7.8.2.1 _read1()

```
def _read1 (
                self,
                address,
                cmd )  [private]
```

```
203     def _read1(self,address,cmd):
204         trys = self._trystimeout
205         while 1:
206             self._port.flushInput()
207             self._sendcommand(address,cmd)
208             val1 = self._readbyte()
209             if val1[0]:
210                 crc = self._readchecksumword()
211                 if crc[0]:
212                     if self._crc&0xFFFF!=crc[1]&0xFFFF:
213                         return (0,0)
214                     return (1,val1[1])
215             trys-=1
216             if trys==0:
217                 break
218         return (0,0)
219
```

7.8.2.2 _read2()

```
def _read2 (
                self,
                address,
                cmd )  [private]
```

```
220     def _read2(self,address,cmd):
221         trys = self._trystimeout
222         while 1:
223             self._port.flushInput()
224             self._sendcommand(address,cmd)
225             val1 = self._readword()
226             if val1[0]:
227                 crc = self._readchecksumword()
228                 if crc[0]:
229                     if self._crc&0xFFFF!=crc[1]&0xFFFF:
230                         return (0,0)
231                     return (1,val1[1])
232             trys-=1
233             if trys==0:
234                 break
235         return (0,0)
236
```

7.8.2.3 _read4()

```
def _read4 (
                self,
                address,
                cmd )  [private]
```

```
237     def _read4(self,address,cmd):
238         trys = self._trystimeout
239         while 1:
```

```
240             self._port.flushInput()
241             self._sendcommand(address,cmd)
242             val1 = self._readlong()
243             if val1[0]:
244                 crc = self._readchecksumword()
245                 if crc[0]:
246                     if self._crc&0xFFFF!=crc[1]&0xFFFF:
247                         return (0,0)
248                     return (1,val1[1])
249             trys-=1
250             if trys==0:
251                 break
252         return (0,0)
253
```

7.8.2.4 _read4_1()

```
def _read4_1 (
                self,
                address,
                cmd )  [private]
254     def _read4_1(self,address,cmd):
255         trys = self._trystimeout
256         while 1:
257             self._port.flushInput()
258             self._sendcommand(address,cmd)
259             val1 = self._readslong()
260             if val1[0]:
261                 val2 = self._readbyte()
262                 if val2[0]:
263                     crc = self._readchecksumword()
264                     if crc[0]:
265                         if self._crc&0xFFFF!=crc[1]&0xFFFF:
266                             return (0,0)
267                         return (1,val1[1],val2[1])
268             trys-=1
269             if trys==0:
270                 break
271         return (0,0)
272
```

7.8.2.5 _read_n()

```
def _read_n (
                self,
                address,
                cmd,
                args )  [private]
273     def _read_n(self,address,cmd,args):
274         trys = self._trystimeout
275         while 1:
276             self._port.flushInput()
277             trys-=1
278             if trys==0:
279                 break
280             failed=False
281             self._sendcommand(address,cmd)
282             data = [1,]
283             for i in range(0,args):
284                 val = self._readlong()
285                 if val[0]==0:
286                     failed=True
287                     break
288                 data.append(val[1])
289             if failed:
290                 continue
291             crc = self._readchecksumword()
292             if crc[0]:
293                 if self._crc&0xFFFF==crc[1]&0xFFFF:
294                     return (data);
295         return (0,0,0,0,0)
296
```

7.8.2.6 _readbyte()

```
def _readbyte (
                self ) [private]
```

```
143     def _readbyte(self):
144         data = self._port.read(1)
145         if len(data):
146             val = ord(data)
147             self.crc_update(val)
148             return (1,val)
149         return (0,0)
150
```

7.8.2.7 _readchecksumword()

```
def _readchecksumword (
                self ) [private]
```

```
135     def _readchecksumword(self):
136         data = self._port.read(2)
137         if len(data)==2:
138 #           crc = (ord(data[0])<<8) | ord(data[1])
139             crc = (data[0]<<8) | data[1]
140             return (1,crc)
141         return (0,0)
142
```

7.8.2.8 _readlong()

```
def _readlong (
                self ) [private]
```

```
159     def _readlong(self):
160         val1 = self._readbyte()
161         if val1[0]:
162             val2 = self._readbyte()
163             if val2[0]:
164                 val3 = self._readbyte()
165                 if val3[0]:
166                     val4 = self._readbyte()
167                     if val4[0]:
168                         return (1,val1[1]<<24|val2[1]<<16|val3[1]<<8|val4[1])
169         return (0,0)
170
```

7.8.2.9 _readslong()

```
def _readslong (
                self ) [private]
```

```
171     def _readslong(self):
172         val = self._readlong()
173         if val[0]:
174             if val[1]&0x80000000:
175                 return (val[0],val[1]-0x100000000)
176             return (val[0],val[1])
177         return (0,0)
178
```

7.8.2.10 _readword()

```
def _readword (
                self ) [private]
```

```
151     def _readword(self):
152         val1 = self._readbyte()
153         if val1[0]:
154             val2 = self._readbyte()
155             if val2[0]:
156                 return (1,val1[1]«8|val2[1])
157         return (0,0)
158
```

7.8.2.11 _sendcommand()

```
def _sendcommand (
                self,
                address,
                command ) [private]
```

```
125     def _sendcommand(self,address,command):
126         self.crc_clear()
127         self.crc_update(address)
128 #       self._port.write(chr(address))
129         self._port.write(address.to_bytes(1, 'big'))
130         self.crc_update(command)
131 #       self._port.write(chr(command))
132         self._port.write(command.to_bytes(1, 'big'))
133         return
134
```

7.8.2.12 _write0()

```
def _write0 (
                self,
                address,
                cmd ) [private]
```

```
305     def _write0(self,address,cmd):
306         trys=self._trystimeout
307         while trys:
308             self._sendcommand(address,cmd)
309             if self._writechecksum():
310                 return True
311             trys=trys-1
312         return False
313
```

7.8.2.13 _write1()

```
def _write1 (
                self,
                address,
                cmd,
                val ) [private]
```

```
314     def _write1(self,address,cmd,val):
315         trys=self._trystimeout
316         while trys:
317             self._sendcommand(address,cmd)
318             self._writebyte(val)
319             if self._writechecksum():
320                 return True
321             trys=trys-1
322         return False
323
```

7.8.2.14 _write11()

```
def _write11 (
                self,
                address,
                cmd,
                val1,
                val2 )  [private]
```

```
324     def _write11(self,address,cmd,val1,val2):
325         trys=self._trystimeout
326         while trys:
327             self._sendcommand(address,cmd)
328             self._writebyte(val1)
329             self._writebyte(val2)
330             if self._writechecksum():
331                 return True
332             trys=trys-1
333         return False
334
```

7.8.2.15 _write111()

```
def _write111 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3 )  [private]
```

```
335     def _write111(self,address,cmd,val1,val2,val3):
336         trys=self._trystimeout
337         while trys:
338             self._sendcommand(address,cmd)
339             self._writebyte(val1)
340             self._writebyte(val2)
341             self._writebyte(val3)
342             if self._writechecksum():
343                 return True
344             trys=trys-1
345         return False
346
```

7.8.2.16 _write2()

```
def _write2 (
                self,
                address,
                cmd,
                val )  [private]
```

```
347     def _write2(self,address,cmd,val):
348         trys=self._trystimeout
349         while trys:
350             self._sendcommand(address,cmd)
351             self._writeword(val)
352             if self._writechecksum():
353                 return True
354             trys=trys-1
355         return False
356
```

7.8.2.17 _write22()

```
def _write22 (
                self,
                address,
                cmd,
                val1,
                val2 )  [private]
```

```
367     def _write22(self,address,cmd,val1,val2):
368         trys=self._trystimeout
369         while trys:
370             self._sendcommand(address,cmd)
371             self._writeword(val1)
372             self._writeword(val2)
373             if self._writechecksum():
374                 return True
375             trys=trys-1
376         return False
377
```

7.8.2.18 _write4()

```
def _write4 (
                self,
                address,
                cmd,
                val )  [private]
```

```
424     def _write4(self,address,cmd,val):
425         trys=self._trystimeout
426         while trys:
427             self._sendcommand(address,cmd)
428             self._writelong(val)
429             if self._writechecksum():
430                 return True
431             trys=trys-1
432         return False
433
```

7.8.2.19 _write44()

```
def _write44 (
                self,
                address,
                cmd,
                val1,
                val2 )  [private]
```

```
444     def _write44(self,address,cmd,val1,val2):
445         trys=self._trystimeout
446         while trys:
447             self._sendcommand(address,cmd)
448             self._writelong(val1)
449             self._writelong(val2)
450             if self._writechecksum():
451                 return True
452             trys=trys-1
453         return False
454
```

7.8.2.20 _write441()

```
def _write441 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3 )  [private]
```

```
477     def _write441(self,address,cmd,val1,val2,val3):
478         trys=self._trystimeout
479         while trys:
480             self._sendcommand(address,cmd)
481             self._writelong(val1)
482             self._writelong(val2)
483             self._writebyte(val3)
484             if self._writechecksum():
485                 return True
486             trys=trys-1
487         return False
488
```

7.8.2.21 _write4444()

```
def _write4444 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3,
                val4 )  [private]
```

```
526     def _write4444(self,address,cmd,val1,val2,val3,val4):
527         trys=self._trystimeout
528         while trys:
529             self._sendcommand(address,cmd)
530             self._writelong(val1)
531             self._writelong(val2)
532             self._writelong(val3)
533             self._writelong(val4)
534             if self._writechecksum():
535                 return True
536             trys=trys-1
537         return False
538
```

7.8.2.22 _write44441()

```
def _write44441 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3,
                val4,
                val5 )  [private]
```

```
552     def _write44441(self,address,cmd,val1,val2,val3,val4,val5):
553         trys=self._trystimeout
554         while trys:
```

```
555             self._sendcommand(address,cmd)
556             self._writelong(val1)
557             self._writelong(val2)
558             self._writelong(val3)
559             self._writelong(val4)
560             self._writebyte(val5)
561             if self._writechecksum():
562                 return True
563             trys=trys-1
564         return False
565
```

7.8.2.23 _write4444444()

```
def _write4444444 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3,
                val4,
                val5,
                val6,
                val7 )  [private]
611     def _write4444444(self,address,cmd,val1,val2,val3,val4,val5,val6,val7):
612         trys=self._trystimeout
613         while trys:
614             self._sendcommand(address,cmd)
615             self._writelong(val1)
616             self._writelong(val2)
617             self._writelong(val3)
618             self._writelong(val4)
619             self._writelong(val5)
620             self._writelong(val6)
621             self._writelong(val7)
622             if self._writechecksum():
623                 return True
624             trys=trys-1
625         return False
626
```

7.8.2.24 _write444444441()

```
def _write444444441 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3,
                val4,
                val5,
                val6,
                val7,
                val8,
                val9 )  [private]
627     def _write444444441(self,address,cmd,val1,val2,val3,val4,val5,val6,val7,val8,val9):
628         trys=self._trystimeout
629         while trys:
630             self._sendcommand(address,cmd)
```

```
631             self._writelong(val1)
632             self._writelong(val2)
633             self._writelong(val3)
634             self._writelong(val4)
635             self._writelong(val5)
636             self._writelong(val6)
637             self._writelong(val7)
638             self._writelong(val8)
639             self._writebyte(val9)
640             if self._writechecksum():
641                 return True
642             trys=trys-1
643         return False
644
```

7.8.2.25 _write4S4()

```
def _write4S4 (
                self,
                address,
                cmd,
                val1,
                val2 )  [private]
455     def _write4S4(self,address,cmd,val1,val2):
456         trys=self._trystimeout
457         while trys:
458             self._sendcommand(address,cmd)
459             self._writelong(val1)
460             self._writeslong(val2)
461             if self._writechecksum():
462                 return True
463             trys=trys-1
464         return False
465
```

7.8.2.26 _write4S441()

```
def _write4S441 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3,
                val4 )  [private]
513     def _write4S441(self,address,cmd,val1,val2,val3,val4):
514         trys=self._trystimeout
515         while trys:
516             self._sendcommand(address,cmd)
517             self._writelong(val1)
518             self._writeslong(val2)
519             self._writelong(val3)
520             self._writebyte(val4)
521             if self._writechecksum():
522                 return True
523             trys=trys-1
524         return False
525
```

7.8.2.27 _write4S444S441()

```
def _write4S444S441 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3,
                val4,
                val5,
                val6,
                val7 )  [private]
```

```
595     def _write4S444S441(self,address,cmd,val1,val2,val3,val4,val5,val6,val7):
596         trys=self._trystimeout
597         while trys:
598             self._sendcommand(self,address,cmd)
599             self._writelong(val1)
600             self._writeslong(val2)
601             self._writelong(val3)
602             self._writelong(val4)
603             self._writeslong(val5)
604             self._writelong(val6)
605             self._writebyte(val7)
606             if self._writechecksum():
607                 return True
608             trys=trys-1
609         return False
610
```

7.8.2.28 _write4S44S4()

```
def _write4S44S4 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3,
                val4 )  [private]
```

```
539     def _write4S44S4(self,address,cmd,val1,val2,val3,val4):
540         trys=self._trystimeout
541         while trys:
542             self._sendcommand(address,cmd)
543             self._writelong(val1)
544             self._writeslong(val2)
545             self._writelong(val3)
546             self._writeslong(val4)
547             if self._writechecksum():
548                 return True
549             trys=trys-1
550         return False
551
```

7.8.2.29 _write4S44S441()

```
def _write4S44S441 (
                self,
                address,
                cmd,
```

```
                val1,
                val2,
                val3,
                val4,
                val5,
                val6 )  [private]
580     def _write4S44S441(self,address,cmd,val1,val2,val3,val4,val5,val6):
581         trys=self._trystimeout
582         while trys:
583             self._sendcommand(address,cmd)
584             self._writelong(val1)
585             self._writeslong(val2)
586             self._writelong(val3)
587             self._writeslong(val4)
588             self._writelong(val5)
589             self._writebyte(val6)
590             if self._writechecksum():
591                 return True
592             trys=trys-1
593         return False
594
```

7.8.2.30 _write4S4S4()

```
def _write4S4S4 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3 )  [private]
501     def _write4S4S4(self,address,cmd,val1,val2,val3):
502         trys=self._trystimeout
503         while trys:
504             self._sendcommand(address,cmd)
505             self._writelong(val1)
506             self._writeslong(val2)
507             self._writeslong(val3)
508             if self._writechecksum():
509                 return True
510             trys=trys-1
511         return False
512
```

7.8.2.31 _writebyte()

```
def _writebyte (
                self,
                val )  [private]
179     def _writebyte(self,val):
180         self.crc_update(val&0xFF)
181 #       self._port.write(chr(val&0xFF))
182         self._port.write(val.to_bytes(1, 'big'))
183
```

7.8.2.32 _writechecksum()

```
def _writechecksum (
                self ) [private]
```

```
297     def _writechecksum(self):
298         self._writeword(self._crc&0xFFFF)
299         val = self._readbyte()
300         if(len(val)>0):
301             if val[0]:
302                 return True
303         return False
304
```

7.8.2.33 _writelong()

```
def _writelong (
                self,
                val ) [private]
```

```
194     def _writelong(self,val):
195         self._writebyte((val»24)&0xFF)
196         self._writebyte((val»16)&0xFF)
197         self._writebyte((val»8)&0xFF)
198         self._writebyte(val&0xFF)
199
```

7.8.2.34 _writeS2()

```
def _writeS2 (
                self,
                address,
                cmd,
                val ) [private]
```

```
357     def _writeS2(self,address,cmd,val):
358         trys=self._trystimeout
359         while trys:
360             self._sendcommand(address,cmd)
361             self._writesword(val)
362             if self._writechecksum():
363                 return True
364             trys=trys-1
365         return False
366
```

7.8.2.35 _writeS22()

```
def _writeS22 (
                self,
                address,
                cmd,
                val1,
                val2 ) [private]
```

```
378     def _writeS22(self,address,cmd,val1,val2):
379         trys=self._trystimeout
380         while trys:
381             self._sendcommand(address,cmd)
382             self._writesword(val1)
383             self._writeword(val2)
384             if self._writechecksum():
385                 return True
386             trys=trys-1
387         return False
388
```

7.8.2.36 _writeS24()

```
def _writeS24 (
                self,
                address,
                cmd,
                val1,
                val2 )  [private]
```

```
400     def _writeS24(self,address,cmd,val1,val2):
401         trys=self._trystimeout
402         while trys:
403             self._sendcommand(address,cmd)
404             self._writesword(val1)
405             self._writelong(val2)
406             if self._writechecksum():
407                 return True
408             trys=trys-1
409         return False
410
```

7.8.2.37 _writeS24S24()

```
def _writeS24S24 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3,
                val4 )  [private]
```

```
411     def _writeS24S24(self,address,cmd,val1,val2,val3,val4):
412         trys=self._trystimeout
413         while trys:
414             self._sendcommand(address,cmd)
415             self._writesword(val1)
416             self._writelong(val2)
417             self._writesword(val3)
418             self._writelong(val4)
419             if self._writechecksum():
420                 return True
421             trys=trys-1
422         return False
423
```

7.8.2.38 _writeS2S2()

```
def _writeS2S2 (
                self,
                address,
                cmd,
                val1,
                val2 )  [private]
```

```
389     def _writeS2S2(self,address,cmd,val1,val2):
390         trys=self._trystimeout
391         while trys:
392             self._sendcommand(address,cmd)
393             self._writesword(val1)
394             self._writesword(val2)
395             if self._writechecksum():
396                 return True
397             trys=trys-1
398         return False
399
```

7.8.2.39 _writeS4()

```
def _writeS4 (
                self,
                address,
                cmd,
                val )  [private]
```

```
434     def _writeS4(self,address,cmd,val):
435         trys=self._trystimeout
436         while trys:
437             self._sendcommand(address,cmd)
438             self._writeslong(val)
439             if self._writechecksum():
440                 return True
441             trys=trys-1
442         return False
443
```

7.8.2.40 _writeS441()

```
def _writeS441 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3 )  [private]
```

```
489     def _writeS441(self,address,cmd,val1,val2,val3):
490         trys=self._trystimeout
491         while trys:
492             self._sendcommand(address,cmd)
493             self._writeslong(val1)
494             self._writelong(val2)
495             self._writebyte(val3)
496             if self._writechecksum():
497                 return True
498             trys=trys-1
499         return False
500
```

7.8.2.41 _writeS44S441()

```
def _writeS44S441 (
                self,
                address,
                cmd,
                val1,
                val2,
                val3,
                val4,
                val5 )  [private]
```

```
566     def _writeS44S441(self,address,cmd,val1,val2,val3,val4,val5):
567         trys=self._trystimeout
568         while trys:
569             self._sendcommand(address,cmd)
570             self._writeslong(val1)
571             self._writelong(val2)
572             self._writeslong(val3)
573             self._writelong(val4)
574             self._writebyte(val5)
575             if self._writechecksum():
576                 return True
577             trys=trys-1
578         return False
579
```

7.8.2.42 _writeS4S4()

```
def _writeS4S4 (
                self,
                address,
                cmd,
                val1,
                val2 )  [private]
```

```
466     def _writeS4S4(self,address,cmd,val1,val2):
467         trys=self._trystimeout
468         while trys:
469             self._sendcommand(address,cmd)
470             self._writeslong(val1)
471             self._writeslong(val2)
472             if self._writechecksum():
473                 return True
474             trys=trys-1
475         return False
476
```

7.8.2.43 _writesbyte()

```
def _writesbyte (
                self,
                val )  [private]
```

```
184     def _writesbyte(self,val):
185         self._writebyte(val)
186
```

7.8.2.44 _writeslong()

```
def _writeslong (
                self,
                val )  [private]
```

```
200     def _writeslong(self,val):
201         self._writelong(val)
202
```

7.8.2.45 _writesword()

```
def _writesword (
                self,
                val )  [private]
```

```
191     def _writesword(self,val):
192         self._writeword(val)
193
```

7.8.2.46 _writeword()

```
def _writeword (
              self,
              val )  [private]
187     def _writeword(self,val):
188         self._writebyte((val»8)&0xFF)
189         self._writebyte(val&0xFF)
190
```

7.8.2.47 BackwardM1()

```
def BackwardM1 (
              self,
              address,
              val )
656     def BackwardM1(self,address,val):
657         return self._write1(address,self.Cmd.M1BACKWARD,val)
658
```

7.8.2.48 BackwardM2()

```
def BackwardM2 (
              self,
              address,
              val )
668     def BackwardM2(self,address,val):
669         return self._write1(address,self.Cmd.M2BACKWARD,val)
670
```

7.8.2.49 BackwardMixed()

```
def BackwardMixed (
              self,
              address,
              val )
680     def BackwardMixed(self,address,val):
681         return self._write1(address,self.Cmd.MIXEDBACKWARD,val)
682
```

7.8.2.50 crc_clear()

```
def crc_clear (
              self )
112     def crc_clear(self):
113         self._crc = 0
114         return
115
```

7.8.2.51 crc_update()

```
def crc_update (
                self,
                data )
116     def crc_update(self,data):
117         self._crc = self._crc ^ (data « 8)
118         for bit in range(0, 8):
119             if (self._crc&0x8000)  == 0x8000:
120                 self._crc = ((self._crc « 1) ^ 0x1021)
121             else:
122                 self._crc = self._crc « 1
123         return
124
```

7.8.2.52 DutyAccelM1()

```
def DutyAccelM1 (
                self,
                address,
                accel,
                duty )
854     def DutyAccelM1(self,address,accel,duty):
855         return self._writeS24(address,self.Cmd.M1DUTYACCEL,duty,accel)
856
```

7.8.2.53 DutyAccelM1M2()

```
def DutyAccelM1M2 (
                self,
                address,
                accel1,
                duty1,
                accel2,
                duty2 )
860     def DutyAccelM1M2(self,address,accel1,duty1,accel2,duty2):
861         return self._writeS24S24(address,self.Cmd.MIXEDDUTYACCEL,duty1,accel1,duty2,accel2)
862
```

7.8.2.54 DutyAccelM2()

```
def DutyAccelM2 (
                self,
                address,
                accel,
                duty )
857     def DutyAccelM2(self,address,accel,duty):
858         return self._writeS24(address,self.Cmd.M2DUTYACCEL,duty,accel)
859
```

7.8.2.55 DutyM1()

```
def DutyM1 (
                self,
                address,
                val )
773     def DutyM1(self,address,val):
774         return self._writeS2(address,self.Cmd.M1DUTY,val)
775
```

7.8.2.56 DutyM1M2()

```
def DutyM1M2 (
                self,
                address,
                m1,
                m2 )
779     def DutyM1M2(self,address,m1,m2):
780         return self._writeS2S2(address,self.Cmd.MIXEDDUTY,m1,m2)
781
```

7.8.2.57 DutyM2()

```
def DutyM2 (
                self,
                address,
                val )
776     def DutyM2(self,address,val):
777         return self._writeS2(address,self.Cmd.M2DUTY,val)
778
```

7.8.2.58 ForwardBackwardM1()

```
def ForwardBackwardM1 (
                self,
                address,
                val )
671     def ForwardBackwardM1(self,address,val):
672         return self._write1(address,self.Cmd.M17BIT,val)
673
```

7.8.2.59 ForwardBackwardM2()

```
def ForwardBackwardM2 (
                self,
                address,
                val )
674     def ForwardBackwardM2(self,address,val):
675         return self._write1(address,self.Cmd.M27BIT,val)
676
```

7.8.2.60 ForwardBackwardMixed()

```
def ForwardBackwardMixed (
                self,
                address,
                val )
689     def ForwardBackwardMixed(self,address,val):
690         return self._write1(address,self.Cmd.MIXEDFB,val)
691
```

7.8.2.61 ForwardM1()

```
def ForwardM1 (
                self,
                address,
                val )
653     def ForwardM1(self,address,val):
654         return self._write1(address,self.Cmd.M1FORWARD,val)
655
```

7.8.2.62 ForwardM2()

```
def ForwardM2 (
                self,
                address,
                val )
665     def ForwardM2(self,address,val):
666         return self._write1(address,self.Cmd.M2FORWARD,val)
667
```

7.8.2.63 ForwardMixed()

```
def ForwardMixed (
                self,
                address,
                val )
677     def ForwardMixed(self,address,val):
678         return self._write1(address,self.Cmd.MIXEDFORWARD,val)
679
```

7.8.2.64 GetConfig()

```
def GetConfig (
                self,
                address )
1015     def GetConfig(self,address):
1016         return self._read2(address,self.Cmd.GETCONFIG)
1017
```

7.8.2.65 GetDeadBand()

```
def GetDeadBand (
               self,
               address )
970     def GetDeadBand(self,address):
971         val = self._read2(address,self.Cmd.GETDEADBAND)
972         if val[0]:
973             return (1,val[1]»8,val[1]&0xFF)
974         return (0,0,0)
975
```

7.8.2.66 LeftRightMixed()

```
def LeftRightMixed (
               self,
               address,
               val )
692     def LeftRightMixed(self,address,val):
693         return self._write1(address,self.Cmd.MIXEDLR,val)
694
```

7.8.2.67 Open()

```
def Open (
               self )
1076     def Open(self):
1077         try:
1078             self._port = serial.Serial(port=self.comport, baudrate=self.rate, timeout=1,
       interCharTimeout=self.timeout)
1079             #self._port = serial.Serial(port=self.comport, baudrate=self.rate, timeout=1,
       interCharTimeout=self.timeout)
1080         except:
1081             return 0
1082         return 1
1083
```

7.8.2.68 ReadBuffers()

```
def ReadBuffers (
               self,
               address )
818     def ReadBuffers(self,address):
819         val = self._read2(address,self.Cmd.GETBUFFERS)
820         if val[0]:
821             return (1,val[1]»8,val[1]&0xFF)
822         return (0,0,0)
823
```

7.8.2.69 ReadCurrents()

```
def ReadCurrents (
                self,
                address )
```

```
836     def ReadCurrents(self,address):
837         val = self._read4(address,self.Cmd.GETCURRENTS)
838         if val[0]:
839             cur1 = val[1]»16
840             cur2 = val[1]&0xFFFF
841             if cur1&0x8000:
842                 cur1-=0x10000
843             if cur2&0x8000:
844                 cur2-=0x10000
845             return (1,cur1,cur2)
846         return (0,0,0)
847
```

7.8.2.70 ReadEeprom()

```
def ReadEeprom (
                self,
                address,
                ee_address )
```

```
1042     def ReadEeprom(self,address,ee_address):
1043         trys = self._trystimeout
1044         while 1:
1045             self._port.flushInput()
1046             self._sendcommand(address,self.Cmd.READEEPROM)
1047             self.crc_update(ee_address)
1048             self._port.write(chr(ee_address))
1049             val1 = self._readword()
1050             if val1[0]:
1051                 crc = self._readchecksumword()
1052                 if crc[0]:
1053                     if self._crc&0xFFFF!=crc[1]&0xFFFF:
1054                         return (0,0)
1055                     return (1,val1[1])
1056             trys-=1
1057             if trys==0:
1058                 break
1059         return (0,0)
1060
```

7.8.2.71 ReadEncM1()

```
def ReadEncM1 (
                self,
                address )
```

```
695     def ReadEncM1(self,address):
696         return self._read4_1(address,self.Cmd.GETM1ENC)
697
```

7.8.2.72 ReadEncM2()

```
def ReadEncM2 (
                self,
                address )
```

```
698     def ReadEncM2(self,address):
699         return self._read4_1(address,self.Cmd.GETM2ENC)
700
```

7.8.2.73 ReadEncoderModes()

```
def ReadEncoderModes (
                self,
                address )
```

```
989     def ReadEncoderModes(self,address):
990         val = self._read2(address,self.Cmd.GETENCODERMODE)
991         if val[0]:
992             return (1,val[1]»8,val[1]&0xFF)
993         return (0,0,0)
994
```

7.8.2.74 ReadError()

```
def ReadError (
                self,
                address )
```

```
986     def ReadError(self,address):
987         return self._read4(address,self.Cmd.GETERROR)
988
```

7.8.2.75 ReadISpeedM1()

```
def ReadISpeedM1 (
                self,
                address )
```

```
767     def ReadISpeedM1(self,address):
768         return self._read4_1(address,self.Cmd.GETM1ISPEED)
769
```

7.8.2.76 ReadISpeedM2()

```
def ReadISpeedM2 (
                self,
                address )
```

```
770     def ReadISpeedM2(self,address):
771         return self._read4_1(address,self.Cmd.GETM2ISPEED)
772
```

7.8.2.77 ReadLogicBatteryVoltage()

```
def ReadLogicBatteryVoltage (
                self,
                address )
```

```
750     def ReadLogicBatteryVoltage(self,address,):
751         return self._read2(address,self.Cmd.GETLBATT)
752
```

7.8.2.78 ReadM1MaxCurrent()

```
def ReadM1MaxCurrent (
                self,
                address )
1024     def ReadM1MaxCurrent(self,address):
1025         data = self._read_n(address,self.Cmd.GETM1MAXCURRENT,2)
1026         if data[0]:
1027             return (1,data[1])
1028         return (0,0)
1029
```

7.8.2.79 ReadM1PositionPID()

```
def ReadM1PositionPID (
                self,
                address )
911     def ReadM1PositionPID(self,address):
912         data = self._read_n(address,self.Cmd.READM1POSPID,7)
913         if data[0]:
914             data[1]/=1024.0
915             data[2]/=1024.0
916             data[3]/=1024.0
917             return data
918         return (0,0,0,0,0,0,0,0)
919
```

7.8.2.80 ReadM1VelocityPID()

```
def ReadM1VelocityPID (
                self,
                address )
863     def ReadM1VelocityPID(self,address):
864         data = self._read_n(address,self.Cmd.READM1PID,4)
865         if data[0]:
866             data[1]/=65536.0
867             data[2]/=65536.0
868             data[3]/=65536.0
869             return data
870         return (0,0,0,0,0)
871
```

7.8.2.81 ReadM2MaxCurrent()

```
def ReadM2MaxCurrent (
                self,
                address )
1030     def ReadM2MaxCurrent(self,address):
1031         data = self._read_n(address,self.Cmd.GETM2MAXCURRENT,2)
1032         if data[0]:
1033             return (1,data[1])
1034         return (0,0)
1035
```

7.8.2.82 ReadM2PositionPID()

```
def ReadM2PositionPID (
                self,
                address )
```

```
920     def ReadM2PositionPID(self,address):
921         data = self._read_n(address,self.Cmd.READM2POSPID,7)
922         if data[0]:
923             data[1]/=1024.0
924             data[2]/=1024.0
925             data[3]/=1024.0
926             return data
927         return (0,0,0,0,0,0,0,0)
928
```

7.8.2.83 ReadM2VelocityPID()

```
def ReadM2VelocityPID (
                self,
                address )
```

```
872     def ReadM2VelocityPID(self,address):
873         data = self._read_n(address,self.Cmd.READM2PID,4)
874         if data[0]:
875             data[1]/=65536.0
876             data[2]/=65536.0
877             data[3]/=65536.0
878             return data
879         return (0,0,0,0,0)
880
```

7.8.2.84 ReadMainBatteryVoltage()

```
def ReadMainBatteryVoltage (
                self,
                address )
```

```
747     def ReadMainBatteryVoltage(self,address):
748         return self._read2(address,self.Cmd.GETMBATT)
749
```

7.8.2.85 ReadMinMaxLogicVoltages()

```
def ReadMinMaxLogicVoltages (
                self,
                address )
```

```
895     def ReadMinMaxLogicVoltages(self,address):
896         val = self._read4(address,self.Cmd.GETMINMAXLOGICVOLTAGES)
897         if val[0]:
898             min = val[1]»16
899             max = val[1]&0xFFFF
900             return (1,min,max)
901         return (0,0,0)
902
```

7.8.2.86 ReadMinMaxMainVoltages()

```
def ReadMinMaxMainVoltages (
                self,
                address )
887     def ReadMinMaxMainVoltages(self,address):
888         val = self._read4(address,self.Cmd.GETMINMAXMAINVOLTAGES)
889         if val[0]:
890             min = val[1]»16
891             max = val[1]&0xFFFF
892             return (1,min,max)
893         return (0,0,0)
894
```

7.8.2.87 ReadNVM()

```
def ReadNVM (
                self,
                address )
1007     def ReadNVM(self,address):
1008         return self._write0(address,self.Cmd.READNVM)
1009
```

7.8.2.88 ReadPinFunctions()

```
def ReadPinFunctions (
                self,
                address )
947     def ReadPinFunctions(self,address):
948         trys = self._trystimeout
949         while 1:
950             self._sendcommand(address,self.Cmd.GETPINFUNCTIONS)
951             val1 = self._readbyte()
952             if val1[0]:
953                 val2 = self._readbyte()
954                 if val1[0]:
955                     val3 = self._readbyte()
956                     if val1[0]:
957                         crc = self._readchecksumword()
958                         if crc[0]:
959                             if self._crc&0xFFFF!=crc[1]&0xFFFF:
960                                 return (0,0)
961                             return (1,val1[1],val2[1],val3[1])
962             trys-=1
963             if trys==0:
964                 break
965         return (0,0)
966
```

7.8.2.89 ReadPWMMode()

```
def ReadPWMMode (
                self,
                address )
1039     def ReadPWMMode(self,address):
1040         return self._read1(address,self.Cmd.GETPWMMODE)
1041
```

7.8.2.90 ReadPWMs()

```
def ReadPWMs (
               self,
               address )
824     def ReadPWMs(self,address):
825         val = self._read4(address,self.Cmd.GETPWMS)
826         if val[0]:
827             pwm1 = val[1]»16
828             pwm2 = val[1]&0xFFFF
829             if pwm1&0x8000:
830                 pwm1-=0x10000
831             if pwm2&0x8000:
832                 pwm2-=0x10000
833             return (1,pwm1,pwm2)
834         return (0,0,0)
835
```

7.8.2.91 ReadSpeedM1()

```
def ReadSpeedM1 (
               self,
               address )
701     def ReadSpeedM1(self,address):
702         return self._read4_1(address,self.Cmd.GETM1SPEED)
703
```

7.8.2.92 ReadSpeedM2()

```
def ReadSpeedM2 (
               self,
               address )
704     def ReadSpeedM2(self,address):
705         return self._read4_1(address,self.Cmd.GETM2SPEED)
706
```

7.8.2.93 ReadTemp()

```
def ReadTemp (
               self,
               address )
980     def ReadTemp(self,address):
981         return self._read2(address,self.Cmd.GETTEMP)
982
```

7.8.2.94 ReadTemp2()

```
def ReadTemp2 (
               self,
               address )
983     def ReadTemp2(self,address):
984         return self._read2(address,self.Cmd.GETTEMP2)
985
```

7.8.2.95 ReadVersion()

```
def ReadVersion (
              self,
              address )
```

```
710     def ReadVersion(self,address):
711         trys=self._trystimeout
712         while 1:
713             self._port.flushInput()
714             self._sendcommand(address,self.Cmd.GETVERSION)
715             str = ""
716             passed = True
717             for i in range(0,48):
718                 data = self._port.read(1)
719                 if len(data):
720                     val = ord(data)
721                     self.crc_update(val)
722                     if(val==0):
723                         break
724 #                   str+=data[0]
725                     str+=chr(data[0])
726                 else:
727                     passed = False
728                     break
729             if passed:
730                 crc = self._readchecksumword()
731                 if crc[0]:
732                     if self._crc&0xFFFF==crc[1]&0xFFFF:
733                         return (1,str)
734                     else:
735                         time.sleep(0.01)
736             trys-=1
737             if trys==0:
738                 break
739         return (0,0)
740
```

7.8.2.96 ResetEncoders()

```
def ResetEncoders (
              self,
              address )
```

```
707     def ResetEncoders(self,address):
708         return self._write0(address,self.Cmd.RESETENC)
709
```

7.8.2.97 RestoreDefaults()

```
def RestoreDefaults (
              self,
              address )
```

```
977     def RestoreDefaults(self,address):
978         return self._write0(address,self.Cmd.RESTOREDEFAULTS)
979
```

7.8.2.98 SendRandomData()

```
def SendRandomData (
                self,
                cnt )
646     def SendRandomData(self,cnt):
647         for i in range(0,cnt):
648             byte = random.getrandbits(8)
649 #           self._port.write(chr(byte))
650             self._port.write(byte.to_bytes(1, 'big'))
651         return
652
```

7.8.2.99 SetConfig()

```
def SetConfig (
                self,
                address,
                config )
1012     def SetConfig(self,address,config):
1013         return self._write2(address,self.Cmd.SETCONFIG,config)
1014
```

7.8.2.100 SetDeadBand()

```
def SetDeadBand (
                self,
                address,
                min,
                max )
967     def SetDeadBand(self,address,min,max):
968         return self._write11(address,self.Cmd.SETDEADBAND,min,max)
969
```

7.8.2.101 SetEncM1()

```
def SetEncM1 (
                self,
                address,
                cnt )
741     def SetEncM1(self,address,cnt):
742         return self._write4(address,self.Cmd.SETM1ENCCOUNT,cnt)
743
```

7.8.2.102 SetEncM2()

```
def SetEncM2 (
              self,
              address,
              cnt )
744     def SetEncM2(self,address,cnt):
745         return self._write4(address,self.Cmd.SETM2ENCCOUNT,cnt)
746
```

7.8.2.103 SetLogicVoltages()

```
def SetLogicVoltages (
              self,
              address,
              min,
              max )
884     def SetLogicVoltages(self,address,min, max):
885         return self._write22(address,self.Cmd.SETLOGICVOLTAGES,min,max)
886
```

7.8.2.104 SetM1DefaultAccel()

```
def SetM1DefaultAccel (
              self,
              address,
              accel )
938     def SetM1DefaultAccel(self,address,accel):
939         return self._write4(address,self.Cmd.SETM1DEFAULTACCEL,accel)
940
```

7.8.2.105 SetM1EncoderMode()

```
def SetM1EncoderMode (
              self,
              address,
              mode )
995     def SetM1EncoderMode(self,address,mode):
996         return self._write1(address,self.Cmd.SETM1ENCODERMODE,mode)
997
```

7.8.2.106 SetM1MaxCurrent()

```
def SetM1MaxCurrent (
              self,
              address,
              max )
1018     def SetM1MaxCurrent(self,address,max):
1019         return self._write44(address,self.Cmd.SETM1MAXCURRENT,max,0)
1020
```

7.8.2.107 SetM1PositionPID()

```
def SetM1PositionPID (
              self,
              address,
              kp,
              ki,
              kd,
              kimax,
              deadzone,
              min,
              max )
```

```
903     def SetM1PositionPID(self,address,kp,ki,kd,kimax,deadzone,min,max):
904 #       return
        self._write4444444(address,self.Cmd.SETM1POSPID,long(kd*1024),long(kp*1024),long(ki*1024),kimax,deadzone,min,max)
905         return
        self._write4444444(address,self.Cmd.SETM1POSPID,kd*1024,kp*1024,ki*1024,kimax,deadzone,min,max)
906
```

7.8.2.108 SetM1VelocityPID()

```
def SetM1VelocityPID (
              self,
              address,
              p,
              i,
              d,
              qpps )
```

```
759     def SetM1VelocityPID(self,address,p,i,d,qpps):
760 #       return self._write4444(address,self.Cmd.SETM1PID,long(d*65536),long(p*65536),long(i*65536),qpps)
761         return self._write4444(address,self.Cmd.SETM1PID,d*65536,p*65536,i*65536,qpps)
762
```

7.8.2.109 SetM2DefaultAccel()

```
def SetM2DefaultAccel (
              self,
              address,
              accel )
```

```
941     def SetM2DefaultAccel(self,address,accel):
942         return self._write4(address,self.Cmd.SETM2DEFAULTACCEL,accel)
943
```

7.8.2.110 SetM2EncoderMode()

```
def SetM2EncoderMode (
              self,
              address,
              mode )
```

```
998     def SetM2EncoderMode(self,address,mode):
999         return self._write1(address,self.Cmd.SETM2ENCODERMODE,mode)
1000
```

7.8.2.111 SetM2MaxCurrent()

```
def SetM2MaxCurrent (
              self,
              address,
              max )
1021     def SetM2MaxCurrent(self,address,max):
1022         return self._write44(address,self.Cmd.SETM2MAXCURRENT,max,0)
1023
```

7.8.2.112 SetM2PositionPID()

```
def SetM2PositionPID (
              self,
              address,
              kp,
              ki,
              kd,
              kimax,
              deadzone,
              min,
              max )
907     def SetM2PositionPID(self,address,kp,ki,kd,kimax,deadzone,min,max):
908 #       return
      self._write4444444(address,self.Cmd.SETM2POSPID,long(kd*1024),long(kp*1024),long(ki*1024),kimax,deadzone,min,max)
909         return
      self._write4444444(address,self.Cmd.SETM2POSPID,kd*1024,kp*1024,ki*1024,kimax,deadzone,min,max)
910
```

7.8.2.113 SetM2VelocityPID()

```
def SetM2VelocityPID (
              self,
              address,
              p,
              i,
              d,
              qpps )
763     def SetM2VelocityPID(self,address,p,i,d,qpps):
764 #       return self._write4444(address,self.Cmd.SETM2PID,long(d*65536),long(p*65536),long(i*65536),qpps)
765         return self._write4444(address,self.Cmd.SETM2PID,d*65536,p*65536,i*65536,qpps)
766
```

7.8.2.114 SetMainVoltages()

```
def SetMainVoltages (
              self,
              address,
              min,
              max )
881     def SetMainVoltages(self,address,min, max):
882         return self._write22(address,self.Cmd.SETMAINVOLTAGES,min,max)
883
```

7.8.2.115 SetMaxVoltageLogicBattery()

```
def SetMaxVoltageLogicBattery (
              self,
              address,
              val )
```

```
756     def SetMaxVoltageLogicBattery(self,address,val):
757         return self._write1(address,self.Cmd.SETMAXLB,val)
758
```

7.8.2.116 SetMaxVoltageMainBattery()

```
def SetMaxVoltageMainBattery (
              self,
              address,
              val )
```

```
662     def SetMaxVoltageMainBattery(self,address,val):
663         return self._write1(address,self.Cmd.SETMAXMB,val)
664
```

7.8.2.117 SetMinVoltageLogicBattery()

```
def SetMinVoltageLogicBattery (
              self,
              address,
              val )
```

```
753     def SetMinVoltageLogicBattery(self,address,val):
754         return self._write1(address,self.Cmd.SETMINLB,val)
755
```

7.8.2.118 SetMinVoltageMainBattery()

```
def SetMinVoltageMainBattery (
              self,
              address,
              val )
```

```
659     def SetMinVoltageMainBattery(self,address,val):
660         return self._write1(address,self.Cmd.SETMINMB,val)
661
```

7.8.2.119 SetPinFunctions()

```
def SetPinFunctions (
              self,
              address,
              S3mode,
              S4mode,
              S5mode )
```

```
944     def SetPinFunctions(self,address,S3mode,S4mode,S5mode):
945         return self._write111(address,self.Cmd.SETPINFUNCTIONS,S3mode,S4mode,S5mode)
946
```

7.8.2.120 SetPWMMode()

```
def SetPWMMode (
                self,
                address,
                mode )
1036     def SetPWMMode(self,address,mode):
1037         return self._write1(address,self.Cmd.SETPWMMODE,mode)
1038
```

7.8.2.121 SpeedAccelDeccelPositionM1()

```
def SpeedAccelDeccelPositionM1 (
                self,
                address,
                accel,
                speed,
                deccel,
                position,
                buffer )
929     def SpeedAccelDeccelPositionM1(self,address,accel,speed,deccel,position,buffer):
930         return
        self._write44441(address,self.Cmd.M1SPEEDACCELDECCELPOS,accel,speed,deccel,position,buffer)
931
```

7.8.2.122 SpeedAccelDeccelPositionM1M2()

```
def SpeedAccelDeccelPositionM1M2 (
                self,
                address,
                accel1,
                speed1,
                deccel1,
                position1,
                accel2,
                speed2,
                deccel2,
                position2,
                buffer )
935     def
        SpeedAccelDeccelPositionM1M2(self,address,accel1,speed1,deccel1,position1,accel2,speed2,deccel2,position2,buffer):
936         return
        self._write444444441(address,self.Cmd.MIXEDSPEEDACCELDECCELPOS,accel1,speed1,deccel1,position1,accel2,speed2,deccel2,po
937
```

7.8.2.123 SpeedAccelDeccelPositionM2()

```
def SpeedAccelDeccelPositionM2 (
                self,
                address,
                accel,
                speed,
                deccel,
                position,
                buffer )
932     def SpeedAccelDeccelPositionM2(self,address,accel,speed,deccel,position,buffer):
933         return
       self._write44441(address,self.Cmd.M2SPEEDACCELDECCELPOS,accel,speed,deccel,position,buffer)
934
```

7.8.2.124 SpeedAccelDistanceM1()

```
def SpeedAccelDistanceM1 (
                self,
                address,
                accel,
                speed,
                distance,
                buffer )
809     def SpeedAccelDistanceM1(self,address,accel,speed,distance,buffer):
810         return self._write4S441(address,self.Cmd.M1SPEEDACCELDIST,accel,speed,distance,buffer)
811
```

7.8.2.125 SpeedAccelDistanceM1M2()

```
def SpeedAccelDistanceM1M2 (
                self,
                address,
                accel,
                speed1,
                distance1,
                speed2,
                distance2,
                buffer )
815     def SpeedAccelDistanceM1M2(self,address,accel,speed1,distance1,speed2,distance2,buffer):
816         return
       self._write4S44S441(address,self.Cmd.MIXEDSPEEDACCELDIST,accel,speed1,distance1,speed2,distance2,buffer)
817
```

7.8.2.126 SpeedAccelDistanceM1M2_2()

```
def SpeedAccelDistanceM1M2_2 (
                self,
                address,
                accel1,
                speed1,
                distance1,
                accel2,
                speed2,
                distance2,
                buffer )
851     def SpeedAccelDistanceM1M2_2(self,address,accel1,speed1,distance1,accel2,speed2,distance2,buffer):
852         return
       self._write4S444S441(address,self.Cmd.MIXEDSPEED2ACCELDIST,accel1,speed1,distance1,accel2,speed2,distance2,buffer)
853
```

7.8.2.127 SpeedAccelDistanceM2()

```
def SpeedAccelDistanceM2 (
                self,
                address,
                accel,
                speed,
                distance,
                buffer )
812     def SpeedAccelDistanceM2(self,address,accel,speed,distance,buffer):
813         return self._write4S441(address,self.Cmd.M2SPEEDACCELDIST,accel,speed,distance,buffer)
814
```

7.8.2.128 SpeedAccelM1()

```
def SpeedAccelM1 (
                self,
                address,
                accel,
                speed )
791     def SpeedAccelM1(self,address,accel,speed):
792         return self._write4S4(address,self.Cmd.M1SPEEDACCEL,accel,speed)
793
```

7.8.2.129 SpeedAccelM1M2()

```
def SpeedAccelM1M2 (
                self,
                address,
                accel,
                speed1,
                speed2 )
797     def SpeedAccelM1M2(self,address,accel,speed1,speed2):
798         return self._write4S4S4(address,self.Cmd.MIXEDSPEEDACCEL,accel,speed1,speed2)
799
```

7.8.2.130 SpeedAccelM1M2_2()

```
def SpeedAccelM1M2_2 (
                self,
                address,
                accel1,
                speed1,
                accel2,
                speed2 )
848     def SpeedAccelM1M2_2(self,address,accel1,speed1,accel2,speed2):
849         return self._write4S44S4(address,self.Cmd.MIXEDSPEED2ACCEL,accel,speed1,accel2,speed2)
850
```

7.8.2.131 SpeedAccelM2()

```
def SpeedAccelM2 (
                self,
                address,
                accel,
                speed )
794     def SpeedAccelM2(self,address,accel,speed):
795         return self._write4S4(address,self.Cmd.M2SPEEDACCEL,accel,speed)
796
```

7.8.2.132 SpeedDistanceM1()

```
def SpeedDistanceM1 (
                self,
                address,
                speed,
                distance,
                buffer )
800     def SpeedDistanceM1(self,address,speed,distance,buffer):
801         return self._writeS441(address,self.Cmd.M1SPEEDDIST,speed,distance,buffer)
802
```

7.8.2.133 SpeedDistanceM1M2()

```
def SpeedDistanceM1M2 (
                self,
                address,
                speed1,
                distance1,
                speed2,
                distance2,
                buffer )
806     def SpeedDistanceM1M2(self,address,speed1,distance1,speed2,distance2,buffer):
807         return
       self._writeS44S441(address,self.Cmd.MIXEDSPEEDDIST,speed1,distance1,speed2,distance2,buffer)
808
```

7.8.2.134 SpeedDistanceM2()

```
def SpeedDistanceM2 (
                self,
                address,
                speed,
                distance,
                buffer )
803     def SpeedDistanceM2(self,address,speed,distance,buffer):
804         return self._writeS441(address,self.Cmd.M2SPEEDDIST,speed,distance,buffer)
805
```

7.8.2.135 SpeedM1()

```
def SpeedM1 (
                self,
                address,
                val )
782     def SpeedM1(self,address,val):
783         return self._writeS4(address,self.Cmd.M1SPEED,val)
784
```

7.8.2.136 SpeedM1M2()

```
def SpeedM1M2 (
                self,
                address,
                m1,
                m2 )
788     def SpeedM1M2(self,address,m1,m2):
789         return self._writeS4S4(address,self.Cmd.MIXEDSPEED,m1,m2)
790
```

7.8.2.137 SpeedM2()

```
def SpeedM2 (
                self,
                address,
                val )
785     def SpeedM2(self,address,val):
786         return self._writeS4(address,self.Cmd.M2SPEED,val)
787
```

7.8.2.138 TurnLeftMixed()

```
def TurnLeftMixed (
                self,
                address,
                val )
```

```
686     def TurnLeftMixed(self,address,val):
687         return self._write1(address,self.Cmd.MIXEDLEFT,val)
688
```

7.8.2.139 TurnRightMixed()

```
def TurnRightMixed (
                self,
                address,
                val )
```

```
683     def TurnRightMixed(self,address,val):
684         return self._write1(address,self.Cmd.MIXEDRIGHT,val)
685
```

7.8.2.140 WriteEeprom()

```
def WriteEeprom (
                self,
                address,
                ee_address,
                ee_word )
```

```
1061     def WriteEeprom(self,address,ee_address,ee_word):
1062         retval = self._write111(address,self.Cmd.WRITEEEPROM,ee_address,ee_word»8,ee_word&0xFF)
1063         if retval==True:
1064             trys = self._trystimeout
1065             while 1:
1066                 self._port.flushInput()
1067                 val1 = self._readbyte()
1068                 if val1[0]:
1069                     if val1[1]==0xaa:
1070                         return True
1071                 trys-=1
1072                 if trys==0:
1073                     break
1074         return False
1075
```

7.8.2.141 WriteNVM()

```
def WriteNVM (
                self,
                address )
```

```
1002     def WriteNVM(self,address):
1003         return self._write4(address,self.Cmd.WRITENVM,0xE22EAB7A)
1004
```

7.8.3 Member Data Documentation

7.8.3.1 _crc

`_crc [private]`

7.8.3.2 _port

`_port [private]`

7.8.3.3 _trystimeout

`_trystimeout [private]`

7.8.3.4 comport

`comport`

7.8.3.5 rate

`rate`

7.8.3.6 timeout

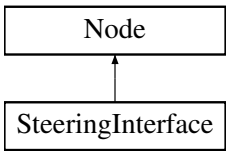
`timeout`

The documentation for this class was generated from the following file:

- main_ws/src/toxic_hardware/toxic_hardware/roboclaw_3.py

7.9 SteeringInterface Class Reference

Inheritance diagram for SteeringInterface:



Public Member Functions

- def __init__ (self)
- def steering_callback (self, data)

Public Attributes

- subscription

7.9.1 Constructor & Destructor Documentation

7.9.1.1 __init__()

```
def __init__ (
                self )
9     def __init__(self):
10         super().__init__('steering_interface')
11         self.subscription = self.create_subscription(
12                 Float64,
13                 '/steering',
14                 self.steering_callback,
15                 60
16                 )
17         self.subscription
18
```

7.9.2 Member Function Documentation

7.9.2.1 steering_callback()

```
def steering_callback (
                self,
                data )
19     def steering_callback(self, data):
20         global servo_pin
21         recived = data.data
22         if recived > 1:
23             recived = 1
24         elif recived < -1:
25             recived = -1
26         lgpio.tx_pwm(
27                 servo_pin,
28                 18,
29                 50,
30                 (7.2 + (recived*((10-5)/2)))
31                 )
32
```

7.9.3 Member Data Documentation

7.9.3.1 subscription

`subscription`

The documentation for this class was generated from the following file:

- main_ws/src/toxic_hardware/toxic_hardware/servo_interface.py

Chapter 8

File Documentation

8.1 main_ws/src/toxic_hardware/setup.py File Reference

Namespaces

- setup

Variables

- string package_name = 'toxic_hardware'
- name
- version
- packages
- data_files
- install_requires
- zip_safe
- maintainer
- maintainer_email
- description
- license
- tests_require
- entry_points

8.2 main_ws/src/toxic_vision/setup.py File Reference

Namespaces

- setup

8.3 main_ws/src/toxic_hardware/toxic_hardware/blinkers_interface.py File Reference

Classes

- class BlinkersInterface

Namespaces

- blinkers_interface

Functions

- def main (args=None)

Variables

- gpio_pin = lgpio.gpiochip_open(0)

8.4 main_ws/src/toxic_hardware/toxic_hardware/controller.py File Reference

Classes

- class ControlSubscriber

Namespaces

- controller

Functions

- def main (args=None)

Variables

- interface = lgpio.gpiochip_open(0)
- roboclaw = Roboclaw("/dev/ttyACM0", 115200)

8.5 main_ws/src/toxic_hardware/toxic_hardware/motor_interface.py File Reference

This script creates a susbcriber node to parse a std_msgs.msg.Float64 ROS's message type to a Roboclaw's serial output.

Classes

- class MotorInterface

Namespaces

- motor_interface

Functions

- def main (args=None)

Variables

- roboclaw = Roboclaw("/dev/ttyACM0", 115200)

8.5.1 Detailed Description

This script creates a susbcriber node to parse a std_msgs.msg.Float64 ROS's message type to a Roboclaw's serial output.

8.6 main_ws/src/toxic_hardware/toxic_hardware/oled_interface.py File Reference

Classes

- class OledInterface

Namespaces

- oled_interface

Functions

- def main (args=None)

Variables

- RST = None
- int DC = 23
- int SPI_PORT = 0
- int SPI_DEVICE = 0
- disp = Adafruit_SSD1306.SSD1306_128_64(rst=RST)
- width = disp.width
- height = disp.height
- image = Image.new('1', (width, height))
- draw = ImageDraw.Draw(image)
- outline
- fill
- int padding = -2
- int top = padding
- int bottom = height-padding
- int x = 0
- font = ImageFont.load_default()

8.7 main_ws/src/toxic_hardware/toxic_hardware/roboclaw_3.py File Reference

Classes

- class Roboclaw
- class Roboclaw.Cmd

Namespaces

- roboclaw_3

8.8 main_ws/src/toxic_hardware/toxic_hardware/servo_interface.py File Reference

Classes

- class SteeringInterface

Namespaces

- servo_interface

Functions

- def main (args=None)

Variables

- servo_pin = lgpio.gpiochip_open(0)

8.9 main_ws/src/toxic_vision/toxic_vision/webcam_pub.py File Reference

Main image publisher as /raw_rgb node.

Classes

- class ImagePublisher

Namespaces

- webcam_pub

Functions

- def main (args=None)

  This function create the publisher node, the image publisher object and starts to run the publisher.

8.9.1 Detailed Description

Main image publisher as /raw_rgb node.

8.9.2 Detail

This script create the image publisher as "/raw_rgb", open the camera, read the camera and publish frames as Ros2 OpenCv messages, to be able to all nodes to subscribe to te current frames to process it.

8.9.3 Important parameters to change.

On this code, you'll change the camera int ID to select the camera you'll want opencv to open, read ad publish on the line:

```
self.cap = cv2.VideoCapture(0)
```

8.9.4 Dependences.

- rclpy
  - Used to node manipulation.
- rclpy.node
  - Used to node manipulation. (really idk why we don't import only this or the above library)
- sensor_msgs.msg.Image
  - Used to be able to use the Image message type (very self-explained, I think)
- cvBridge.cvBridge
  - Used to parse cv2 frame (like a numpy's multidimensional array) to ROS2's Image type
- cv2
  - In this case it's just used to open the camera, grab the image and read it from the buffer

8.9.5 Copyright

This script conforms part from 'El tóxico' and it's licenced unser GPL 3V0.

8.9.6 Code Spinnet.

8.10 main_ws/src/toxic_vision/toxic_vision/webcam_sub.py File Reference

Very simple image subscriber to /raw_rgb node.

Classes

- class ImageSubscriber

Namespaces

- webcam_sub

Functions

- def main (args=None)

  This function create the subscriber and execute it.

8.10.1 Detailed Description

Very simple image subscriber to /raw_rgb node.

8.10.2 Detail

This script create an image subscriber to "/raw_rgb" and draw the recived frame, this node's used just as test to verify the image publisher and nodes communication performance.

8.10.3 Important parameters to change.

On this code, you don'r need to change anything, except if you've changed the name of the image publisher node, or just wanna check another node.

8.10.4 Dependences.

- rclpy
  - Used to node manipulation.
- rclpy.node
  - Used to node manipulation. (really idk why we don't import only this or the above library)
- sensor_msgs.msg.Image
  - Used to be able to use the Image message type (very self-explained, I think)
- cvBridge.cvBridge
  - Used to parse a ROS2's Image type to OpenCv Frame type
- cv2
  - In this case it's just used to draw the image in a frame.

8.10.5 Copyright

This script conforms part from 'El tóxico' and it's licenced unser GPL 3V0.

8.10.6 Code Spinnet.

8.11 README.md File Reference

Index