Drive Car

1

Generated by Doxygen 1.8.13

# **Contents**

1	Drive	e Car	1
	1.1	Introduction	1
	1.2	Build	1
2	Drive	∍-Car	3
3	Data	Structure Index	5
	3.1	Data Structures	5
4	File I	Index	7
	4.1	File List	7
5	Data	Structure Documentation	9
	5.1	axisState Struct Reference	9
	5.2	CANFrame Struct Reference	9
		5.2.1 Detailed Description	10
	5.3	CARState Struct Reference	10
	5.4	Joystick Struct Reference	10
	5.5	Panda Struct Reference	11
		5.5.1 Detailed Description	11
	5.6	Params Struct Reference	11

ii CONTENTS

6	File	Docume	entation		13
	6.1	panda.	h File Refe	erence	13
		6.1.1	Detailed	Description	14
		6.1.2	Macro De	efinition Documentation	14
			6.1.2.1	REQUEST_IN	14
			6.1.2.2	REQUEST_OUT	15
		6.1.3	Function	Documentation	15
			6.1.3.1	panda_can_clear()	15
			6.1.3.2	panda_can_recv()	15
			6.1.3.3	panda_can_send()	16
			6.1.3.4	panda_can_send_many()	16
			6.1.3.5	panda_close()	17
			6.1.3.6	panda_connect()	17
			6.1.3.7	panda_get_version()	17
			6.1.3.8	panda_set_can_speed()	18
			6.1.3.9	panda_set_safety_mode()	18
			6.1.3.10	panda_setup()	19
			6.1.3.11	print_many()	19
Inc	lex				21

# **Drive Car**

# 1.1 Introduction

This project aims to drive a car from a Linux pc using a game controller. For communication with the car interfaces, a comma.ai Panda is used. The comma.ai Panda is talked to via USB and the libusb.

# 1.2 Build

To build this project, you can just run  ${\tt make}$ 

To clean all the build files and the compiled software, run  ${\tt make\ clean}$ 

2 Drive Car

# **Drive-Car**

This is some code to control a Toyota Rav4 Hybrid using a Linux PC.

To control the car a Panda is used. This panda is being communicated with using the libusb library.

4 Drive-Car

# **Data Structure Index**

# 3.1 Data Structures

Here are the data structures with brief descriptions:

axisState	. 9
CANFrame	
Defines a standard CAN frame	. 9
CARState	. 10
Joystick	. 10
Panda	
Defines the interface for a specific connected Panda	. 11
Params	. 11

6 Data Structure Index

# File Index

# 4.1 File List

Here is a list of all documented files with brief descriptions:

joystick.	.h	??
panda.h		
	File containing all panda specific function declarations	13

8 File Index

# **Data Structure Documentation**

# 5.1 axisState Struct Reference

## **Data Fields**

- int16\_t x
- int16\_t **y**

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

· joystick.h

# 5.2 CANFrame Struct Reference

Defines a standard CAN frame.

```
#include <panda.h>
```

## **Data Fields**

uint16\_t ID

The CAN frame ID.

• uint8\_t data [8]

The Data sent with the frame, max. 8 Bytes.

• uint8 t bus

Which bus to send the data on. For using multiple CAN busses.

uint8\_t length

The number of bytes te be sent.

uint8\_t freq

How frequent to send the frame.

## 5.2.1 Detailed Description

Defines a standard CAN frame.

This struct defines a standard CAN frame, so that the software can be used with different CAN devices with different drivers.

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

· panda.h

# 5.3 CARState Struct Reference

#### **Data Fields**

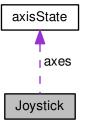
- float speed
- · float angle

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

· main.c

# 5.4 Joystick Struct Reference

Collaboration diagram for Joystick:



#### **Data Fields**

- const char \* name
- int fd
- axisState axes [3]
- uint8\_t **buttons** [12]
- uint8\_t numberOfAxes
- uint8\_t numberOfButtons

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

· joystick.h

5.5 Panda Struct Reference

## 5.5 Panda Struct Reference

Defines the interface for a specific connected Panda.

```
#include <panda.h>
```

#### **Data Fields**

• libusb\_device\_handle \* handle The LibUSB handle.

• struct libusb\_device\_descriptor desc

The LibUSB file descriptor.

## 5.5.1 Detailed Description

Defines the interface for a specific connected Panda.

This struct contains the USB handle and file descriptor, so it can be passed to all functions.

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

• panda.h

# 5.6 Params Struct Reference

#### **Data Fields**

- char \* **js**
- uint8\_t enableDsu
- uint8\_t enableCam

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

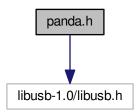
• main.c

# **File Documentation**

# 6.1 panda.h File Reference

File containing all panda specific function declarations.

#include <libusb-1.0/libusb.h>
Include dependency graph for panda.h:



#### **Data Structures**

• struct Panda

Defines the interface for a specific connected Panda.

• struct CANFrame

Defines a standard CAN frame.

# **Macros**

Constant to define what type of request you want to make.

• #define REQUEST\_OUT (LIBUSB\_ENDPOINT\_OUT | LIBUSB\_REQUEST\_TYPE\_VENDOR | LIBUSB\_← RECIPIENT\_DEVICE)

Constant to define what type of request you want to make.

14 File Documentation

#### **Functions**

int panda\_setup (Panda \*p)

Setup and connect to the Panda.

int panda\_connect (Panda \*p)

Connect to the Panda (Called from setup)

int panda\_close (Panda \*p)

Close the USB handle of the Panda.

int panda\_get\_version (Panda \*p)

Retrieve and print the current version of the Panda firmware.

• int panda\_set\_safety\_mode (Panda \*p, uint16\_t mode)

Set the safety mode of the Panda, to allow sending on the CAN busses.

int panda\_set\_can\_speed (Panda \*p, int bus, int speed)

Set the speed of a specific CAN bus of the Panda.

int panda\_can\_send\_many (Panda \*p, CANFrame frames[], int length)

Send many CAN frames to the Panda.

int panda\_can\_send (Panda \*p, CANFrame frame)

Send one CAN frame to the Panda.

• int panda\_can\_recv (Panda \*p, unsigned char \*data, int length)

Request received CAN frames from the Panda.

int panda\_can\_clear (Panda \*p, int bus)

Clear an internal buffer of the Panda.

void print\_many (CANFrame frames[], int length)

Debug the frames that would be sent.

#### 6.1.1 Detailed Description

File containing all panda specific function declarations.

Author

Laurens Wuyts

Date

10 May 2018 This file contains all the function declarations for using the panda, as well as the definition of the Panda struct.

#### 6.1.2 Macro Definition Documentation

#### 6.1.2.1 REQUEST\_IN

#define REQUEST\_IN (LIBUSB\_ENDPOINT\_IN | LIBUSB\_REQUEST\_TYPE\_VENDOR | LIBUSB\_RECIPIENT\_DEVICE)

Constant to define what type of request you want to make.

These defines are constants to define what type of USB request is made.

#### 6.1.2.2 REQUEST\_OUT

```
#define REQUEST_OUT (LIBUSB_ENDPOINT_OUT | LIBUSB_REQUEST_TYPE_VENDOR | LIBUSB_RECIPIENT_DEVI\leftarrow CE)
```

Constant to define what type of request you want to make.

These defines are constants to define what type of USB request is made.

## 6.1.3 Function Documentation

## 6.1.3.1 panda\_can\_clear()

```
int panda_can_clear (
          Panda * p,
          int bus )
```

Clear an internal buffer of the Panda.

#### **Parameters**

р	Pointer to Panda struct.
bus	The bus to clear the buffer of.

#### **Returns**

```
0 Success
< 0 Fail
```

# 6.1.3.2 panda\_can\_recv()

```
int panda_can_recv (
          Panda * p,
          unsigned char * data,
          int length )
```

Request received CAN frames from the Panda.

#### **Parameters**

р	Pointer to Panda struct.
data	The received data from the Panda.
length	The maximum quantity of data to request.

16 File Documentation

#### Returns

```
0 Success
< 0 Fail
```

## 6.1.3.3 panda\_can\_send()

```
int panda_can_send (  {\tt Panda} \, * \, p \text{,}   {\tt CANFrame} \ \textit{frame} \ )
```

Send one CAN frame to the Panda.

#### **Parameters**

p	Pointer to Panda struct.
frame	The frame to send.

#### Returns

```
0 Success
< 0 Fail
```

## 6.1.3.4 panda\_can\_send\_many()

```
int panda_can_send_many (
          Panda * p,
           CANFrame frames[],
          int length )
```

Send many CAN frames to the Panda.

## **Parameters**

р	Pointer to Panda struct.
frames	The CAN frames to send to the Panda.
length	The number of CAN frames to send.

#### Returns

0 Success < 0 Fail

## 6.1.3.5 panda\_close()

Close the USB handle of the Panda.

#### **Parameters**

```
p Pointer to Panda struct.
```

#### Returns

```
0 Success
< 0 Fail
```

## 6.1.3.6 panda\_connect()

Connect to the Panda (Called from setup)

#### **Parameters**

```
p Pointer to Panda struct.
```

# Returns

```
0 Success
< 0 Fail
```

#### 6.1.3.7 panda\_get\_version()

```
int panda_get_version ( {\tt Panda} \ * \ p \ )
```

Retrieve and print the current version of the Panda firmware.

#### **Parameters**

```
p Pointer to Panda struct.
```

18 File Documentation

#### Returns

```
0 Success
< 0 Fail
```

#### 6.1.3.8 panda\_set\_can\_speed()

Set the speed of a specific CAN bus of the Panda.

#### **Parameters**

p	Pointer to Panda struct.
bus	Which bus to change
speed	The speed to set in kbps

## Returns

```
0 Success
< 0 Fail
```

## 6.1.3.9 panda\_set\_safety\_mode()

```
int panda_set_safety_mode (
          Panda * p,
           uint16_t mode )
```

Set the safety mode of the Panda, to allow sending on the CAN busses.

#### **Parameters**

р	Pointer to Panda struct.
mode	Mode to set the Panda to. (0 = listen only, 0x1337 = Write all)

## Returns

```
0 Success
< 0 Fail
```

## 6.1.3.10 panda\_setup()

Setup and connect to the Panda.

#### **Parameters**

```
p Pointer to Panda struct.
```

## Returns

```
0 Success
< 0 Fail
```

# 6.1.3.11 print\_many()

Debug the frames that would be sent.

#### **Parameters**

frames	The frames to print.
length	The number of frames to print.

## Returns

0 Success < 0 Problem 20 File Documentation

# Index

```
axisState, 9
CANFrame, 9
CARState, 10
Joystick, 10
Panda, 11
panda.h, 13
    panda_can_clear, 15
    panda_can_recv, 15
    panda_can_send, 16
    panda_can_send_many, 16
    panda_close, 16
    panda_connect, 17
    panda_get_version, 17
    panda_set_can_speed, 18
    panda_set_safety_mode, 18
    panda_setup, 18
    print_many, 19
    REQUEST_IN, 14
    REQUEST_OUT, 14
panda_can_clear
    panda.h, 15
panda_can_recv
    panda.h, 15
panda can send
    panda.h, 16
panda_can_send_many
    panda.h, 16
panda_close
    panda.h, 16
panda_connect
    panda.h, 17
panda_get_version
    panda.h, 17
panda_set_can_speed
    panda.h, 18
panda_set_safety_mode
    panda.h, 18
panda_setup
    panda.h, 18
Params, 11
print_many
    panda.h, 19
REQUEST_IN
    panda.h, 14
REQUEST_OUT
```

panda.h, 14