Vehicle Control Unit (VCU)

Generated by Doxygen 1.8.14

## **Contents**

1	Fast	tCharge	SAE VCU	firmware									1
2	CAN	l Netwo	rks										3
3	Mod	lule Ind	ex										5
	3.1	Module	es					 	 	 	 		5
4	File	Index											7
	4.1	File Lis	st					 	 	 	 		7
5	Mod	lule Dod	cumentati	on									9
	5.1	CAN_I	module_gr	oup				 	 	 	 		9
		5.1.1	Detailed	Description				 	 	 	 		9
		5.1.2	Function	Documentati	on			 	 	 	 		9
			5.1.2.1	can_init() .				 	 	 	 	 -	9
	5.2	CAN_f	unzionale	_group				 	 	 	 	 -	11
		5.2.1	Detailed	Description				 	 	 	 		12
		5.2.2	Function	Documentati	on			 	 	 	 		12
			5.2.2.1	CAN_FUNZ	_BOOTU	IP_CB()		 	 	 	 		12
			5.2.2.2	CAN_FUNZ	GENER	AL_CB	()	 	 	 	 		12
			5.2.2.3	CAN_FUNZ	_VENDO	R_ID_C	B()	 	 	 	 		13
			5.2.2.4	can_funzion	nale_init()			 	 	 	 		13
			5.2.2.5	can_funzion	nale_initial	lized()		 	 	 	 		14
			5.2.2.6	can_funzion	nale_onlin	e()		 	 	 	 		14
			5.2.2.7	can_funzion	nale_send	l sync()		 	 	 	 		15

ii CONTENTS

		5.2.2.8	get_torque_actual_value()	15
		5.2.2.9	inverter_regen_request()	15
		5.2.2.10	inverter_torque_request()	16
5.3	CAN_s	servizi_gro	up	17
	5.3.1	Detailed	Description	18
	5.3.2	Function	Documentation	18
		5.3.2.1	CAN_SERV_BOOTUP_CB()	18
		5.3.2.2	CAN_SERV_GENERAL_CB()	19
		5.3.2.3	can_servizi_go_operational()	19
		5.3.2.4	can_servizi_init()	20
		5.3.2.5	can_servizi_initialized()	20
		5.3.2.6	can_servizi_online()	21
		5.3.2.7	get_servizi_apps_plausibility()	21
		5.3.2.8	get_servizi_brake()	21
		5.3.2.9	get_servizi_brake_plausibility()	22
		5.3.2.10	get_servizi_tps1()	22
		5.3.2.11	get_servizi_tps2()	23
		5.3.2.12	get_tcs_torque_coefficient()	23
		5.3.2.13	tcs_online()	23
		5.3.2.14	timeout()	24
	5.3.3	Variable	Documentation	24
		5.3.3.1	brake_percentage	24
		5.3.3.2	tps1_percentage	24
		5.3.3.3	tps2_percentage	24
5.4	Board_	_model_gr	oup	25
	5.4.1	Detailed	Description	27
	5.4.2	Function	Documentation	27
		5.4.2.1	get_apps_plausibility()	27
		5.4.2.2	get_brake_percentage()	28
		5.4.2.3	get_brake_plausibility()	28
		5.4.2.4	get_SC_value()	28
		5.4.2.5	get_tps1_percentage()	29
		5.4.2.6	get_tps2_percentage()	29
		5.4.2.7	model_init()	29

CONTENTS

6	File I	Documentation	31
	6.1	can_funzionale.cpp File Reference	31
		6.1.1 Detailed Description	32
	6.2	can_funzionale.h File Reference	32
		6.2.1 Detailed Description	33
	6.3	CAN_ID.h File Reference	33
		6.3.1 Detailed Description	33
	6.4	can_servizi.cpp File Reference	33
		6.4.1 Detailed Description	35
	6.5	can_servizi.h File Reference	35
		6.5.1 Detailed Description	36
	6.6	CO_can.cpp File Reference	36
		6.6.1 Detailed Description	37
	6.7	CO_can.h File Reference	37
		6.7.1 Detailed Description	37
	6.8	common.h File Reference	37
		6.8.1 Detailed Description	38
	6.9	model.cpp File Reference	38
		6.9.1 Detailed Description	40
	6.10	model.h File Reference	41
		6.10.1 Detailed Description	41
	6.11	VCU.ino File Reference	41
		6.11.1 Detailed Description	42
		6.11.2 Function Documentation	42
		6.11.2.1 loop()	42
		6.11.2.2 setup()	42
Inc	dex		43

## **Chapter 1**

# FastChargeSAE VCU firmware

Vehicle Control Unit (VCU) firmware is based upon 4 states:

- STAND: stato 0, accensione della vettura, si ritorna qui ogni volta che casca l'SC
- HVON: stato 1, alta tensione attiva si accede solo da STAND tramite AIRbutton e SC>3V
- DRIVE: stato 2, lo stato di guida sicura, accedibile tramite procedura RTD ma anche con lo scatto delle plausibilità tramite procedura di rientro

## Chapter 2

## **CAN Networks**

Two CAN networks have been designed to be inserted into the vehicle: a first CAN network between the VCU and the inverter (CAN funzionale) and a second CAN network between the VCU, TCU and SCUs (CAN servizi).

4 CAN Networks

# **Chapter 3**

# **Module Index**

## 3.1 Modules

Here is a list of all modules:

CAN_module_group	. 9
CAN_funzionale_group	11
CAN_servizi_group	17
Board_model_group	. 25

6 Module Index

## **Chapter 4**

## File Index

## 4.1 File List

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

oard_pinout.h	??
an_funzionale.cpp	
CAN funzionale module implementation	31
an_funzionale.h	
CAN funzionale module header	32
CAN_ID.h	
CAN nodes ID definitions module	33
an_servizi.cpp	
CAN servizi module implementation	33
an_servizi.h	
CAN servizi module header	35
CO_can.cpp	
CAN setup module implementation	36
CO_can.h	
CAN setup header module	37
ommon.h	
Common macro definitions module	37
ef.h	
lter.cpp	
lter.h	??
nodel.cpp	
Board model implementation file	38
nodel.h	
Board model header file	
tds.cpp	
tds.h	
tates.cpp	
tates.h	??
CU.ino Main module file	/11

8 File Index

## **Chapter 5**

## **Module Documentation**

## 5.1 CAN\_module\_group

## **Modules**

- CAN\_funzionale\_group
- CAN\_servizi\_group

#### **Macros**

#define VCU\_NODE\_ID 2
 VCU Node ID.

## **Functions**

• bool can\_init ()

This function initializes both CAN funzionale and CAN servizi networks.

- 5.1.1 Detailed Description
- 5.1.2 Function Documentation

```
5.1.2.1 can_init()
bool can_init ( )
```

This function initializes both CAN funzionale and CAN servizi networks.

## Author

```
Arella Matteo (mail: arella.1646983@studenti.uniroma1.it)
```

## Return values

true	CAN networks initialized successfully
false	CAN networks initialization failed

Definition at line 26 of file CO\_can.cpp.

## 5.2 CAN\_funzionale\_group

#### **Macros**

• #define INVERTER NODE ID 1

Inverter Node ID.

#### **Functions**

· volatile bool can funzionale initialized ()

This function returns CAN funzionale initialization status.

void can\_funzionale\_send\_sync ()

This function sends a periodic CANOpen sync message to inverter slave node.

void CAN FUNZ BOOTUP CB (CAN FRAME \*frame)

This function manage boot-up message sent over CAN funzionale network by inverter slave node. Upon boot-up message reception the VCU send a SDO client request for check inverter vendor ID; then inverter is considered online over CAN funzionale network.

• void CAN\_FUNZ\_VENDOR\_ID\_CB (CAN\_FRAME \*frame)

This function manage SDO server response with inverter Vendor ID. VCU sends NMT operational and PDOs to enable PWM; then inverter is considered correctly configured and a timer is started for sending periodic sync messages.

• void CAN\_FUNZ\_GENERAL\_CB (CAN\_FRAME \*frame)

This function manage TPDO from inverter and deserializes data:

bool can funzionale init ()

This function initialize CAN funzionale hardware port with baudrate CAN\_FUNZ\_BAUDRATE. Mailbox 0 is configured for receiving boot-up messages from inverter slave node (filter = 0x00000700 + INVERTER\_NODE\_ID, mask = 0x1FFFFFFF); mailbox 1 is configured for receiving vendorID SDO response from inverter (filter = 0x00000580 + INVERTER\_NODE\_ID, mask = 0x1FFFFFFFF); remaining mailboxes are configured for receiving TPDOs from inverter slave node (filter = 0x00000080, mask = 0x1FFFFCFF).

• volatile bool can\_funzionale\_online ()

This function returns if inverter is online and active over CAN funzionale.

void inverter\_torque\_request (uint16\_t torque)

This function send torque request to inverter. If inverter is active over CAN funzionale network then the request is done via RPDO1 viceversa it's done via analog signal.

void inverter regen request (uint16 t regen)

This function send regen request to inverter.

volatile uint16\_t get\_torque\_actual\_value ()

This function return the torque value requested by inverter to motor retrieved from TPDO1 from inverter over CAN funzionale network.

## **Variables**

volatile bool can\_funz\_initialized = false

CAN funzionale initialization status flag (true if initialized)

volatile bool inverter\_online = false

Inverter online status flag (true if online)

volatile bool inverter configured = false

Inverter configured status flag (true if configured)

volatile uint16\_t torque\_actual\_value = 0

Torque requested by inverter to motor.

## 5.2.1 Detailed Description

## 5.2.2 Function Documentation

## 5.2.2.1 CAN\_FUNZ\_BOOTUP\_CB()

This function manage boot-up message sent over CAN funzionale network by inverter slave node. Upon boot-up message reception the VCU send a SDO client request for check inverter vendor ID; then inverter is considered online over CAN funzionale network.

#### Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### **Parameters**

in	frame	CAN frame received from CAN funzionale port	
----	-------	---	--

Definition at line 77 of file can\_funzionale.cpp.

#### 5.2.2.2 CAN\_FUNZ\_GENERAL\_CB()

This function manage TPDO from inverter and deserializes data:

TPDO num	NODE-ID	Data
1	INVERTER_NODE_ID	Torque Actual Val

#### Author

#### Arella Mattec

```
(mail: arella.1646983@studenti.uniroma1.it)
```

## **Parameters**

in	frame	CAN frame received from CAN servizi port
----	-------	--

Definition at line 170 of file can\_funzionale.cpp.

#### 5.2.2.3 CAN\_FUNZ\_VENDOR\_ID\_CB()

This function manage SDO server response with inverter Vendor ID. VCU sends NMT operational and PDOs to enable PWM; then inverter is considered correctly configured and a timer is started for sending periodic sync messages.

#### **Author**

## Arella Matteo (mail: arella.1646983@studenti.uniromal.it)

#### **Parameters**

in	frame	CAN frame received from CAN servizi port	7
----	-------	--	---

Definition at line 108 of file can\_funzionale.cpp.

## 5.2.2.4 can\_funzionale\_init()

```
bool can_funzionale_init ( )
```

This function initialize CAN funzionale hardware port with baudrate CAN\_FUNZ\_BAUDRATE. Mailbox 0 is configured for receiving boot-up messages from inverter slave node (filter = 0x00000700 + INVERTER\_NODE\_ID, mask = 0x1FFFFFFF); mailbox 1 is configured for receiving vendorID SDO response from inverter (filter = 0x00000580 + INVERTER\_NODE\_ID, mask = 0x1FFFFFFF); remaining mailboxes are configured for receiving TPDOs from inverter slave node (filter = 0x00000080, mask = 0x1FFFFCFF).

This function initializes CAN funzionale network with inverter.

## **Author**

## Arella Matteo (mail: arella.1646983@studenti.uniromal.it)

### Return values

true	CAN servizi initialized
false	CAN servizi not initialized

#### Author

## Arella Matteo

```
(mail: arella.1646983@studenti.uniromal.it)
```

#### Return values

true	CAN funzionale network initialized successfully
false	CAN funzionale network initialization failed

Definition at line 192 of file can\_funzionale.cpp.

## 5.2.2.5 can\_funzionale\_initialized()

```
volatile bool can_funzionale_initialized ( )
```

This function returns CAN funzionale initialization status.

#### Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniromal.it)
```

## Return values

	true	CAN funzionale network initialized	
Ī	false	CAN funzionale network not initialized	

Definition at line 44 of file can\_funzionale.cpp.

## 5.2.2.6 can\_funzionale\_online()

```
volatile bool can_funzionale_online ( )
```

This function returns if inverter is online and active over CAN funzionale.

This function returns if CAN funzionale network is online.

## Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

## Return values

true	CAN funzionale initialized, inverter online and inverter configured successfully
false	CAN funzionale not initialized or inverter not online or configured.

#### Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### **Return values**

true	CAN funzionale network online	
false	CAN funzionale network offline	

Definition at line 226 of file can funzionale.cpp.

```
5.2.2.7 can_funzionale_send_sync()
```

```
void can_funzionale_send_sync ( )
```

This function sends a periodic CANOpen sync message to inverter slave node.

#### **Author**

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

Definition at line 55 of file can\_funzionale.cpp.

#### 5.2.2.8 get\_torque\_actual\_value()

```
volatile uint16_t get_torque_actual_value ( )
```

This function return the torque value requested by inverter to motor retrieved from TPDO1 from inverter over CAN funzionale network.

This function return the torque value requested by inverter to motor.

## Author

## Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### Returns

Torque requested by inverter to motor

Definition at line 276 of file can\_funzionale.cpp.

## 5.2.2.9 inverter\_regen\_request()

This function send regen request to inverter.

### Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

## **Parameters**

in	regen	Regen request value

Definition at line 258 of file can\_funzionale.cpp.

## 5.2.2.10 inverter\_torque\_request()

This function send torque request to inverter. If inverter is active over CAN funzionale network then the request is done via RPDO1 viceversa it's done via analog signal.

This function send torque request to inverter.

## Author

## Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

## **Parameters**

in	torque	Torque value in percentage multiplied per tcs torque limiter coefficient

Definition at line 241 of file can\_funzionale.cpp.

5.3 CAN\_servizi\_group 17

## 5.3 CAN\_servizi\_group

#### **Macros**

• #define SCU\_FRONTAL\_NODE\_ID 1

Frontal SCU Node ID.

• #define TCU\_NODE\_ID 4

TCU Node ID.

#### **Functions**

· void timeout ()

This function is executed periodically after CAN servizi 'go Operational' NMT request is sent. When timeout occurs if next\_pedals\_seq\_num is greater than curr\_pedals\_seq\_num then frontal SCU is considered active, viceversa it is considered offline.

• volatile bool can\_servizi\_initialized ()

This function returns CAN servizi initialization status.

void CAN\_SERV\_BOOTUP\_CB (CAN\_FRAME \*frame)

This function manage boot-up messages sent over CAN servizi network by slave nodes.

void CAN SERV GENERAL CB (CAN FRAME \*frame)

This function manage PDOs received over CAN servizi network and deserializes data:

bool can\_servizi\_init ()

This function initialize CAN servizi hardware port with baudrate CAN\_SERV\_BAUDRATE. Mailbox 0 is configured for receiving boot-up messages from CAN servizi slave nodes (filter = 0x00000700, mask = 0x1FFFF80); remaining mailboxes are configured for receiving TPDOs from CAN servizi slave nodes (filter = 0x00000080, mask = 0x1FFF← FC80).

• void can\_servizi\_go\_operational ()

This function send a CANOpen master NMT message for request 'go to Operational' state to CAN servizi slave nodes (SCUs and TCU).

volatile bool can\_servizi\_online ()

This function returns if CAN servizi network is online.

volatile bool tcs\_online ()

This function returns if TCU node is active and online on the CAN servizi network.

volatile uint8\_t get\_servizi\_tps1 ()

This function returns the value of the first APPS in percentage, retrieved by frontal SCU node over CAN servizi network.

· volatile uint8 t get servizi tps2 ()

This function returns the value of the second APPS in percentage, retrieved by frontal SCU node over CAN servizi network.

volatile uint8\_t get\_servizi\_brake ()

This function returns the value of brake pedal position sensor in percentage, retrieved by frontal SCU node over CAN servizi network.

• volatile bool get\_servizi\_apps\_plausibility ()

This function returns the value of APPS plausibility retrieved by frontal SCU node over CAN servizi network.

volatile bool get\_servizi\_brake\_plausibility ()

This function returns the value of brake plausibility retrieved by frontal SCU node over CAN servizi network.

volatile uint8\_t get\_tcs\_torque\_coefficient ()

This function returns the value of torque limiter percentage retrieved by TCU node over CAN servizi network.

#### **Variables**

volatile bool can\_serv\_initialized = false

CAN servizi initialization status flag (true if initialized)

• volatile bool SCU\_F\_online = false

Frontal SCU online status flag (true if online)

• volatile bool TCS\_online = false

TCS online status flag (true if online)

• volatile uint32\_t curr\_pedals\_seq\_num = 0

Frontal SCU PDOtx1 current sequence number.

volatile uint32\_t next\_pedals\_seq\_num = 0

Frontal SCU PDOtx1 next sequence number.

• volatile uint8\_t tps1\_percentage = 0

First APPS percentage value retrieved by frontal SCU node.

• volatile uint8\_t tps2\_percentage = 0

Second APPS percentage value retrieved by frontal SCU node.

volatile uint8\_t brake\_percentage = 0

Brake pedal position sensor percentage value retrieved by frontal SCU node.

volatile bool apps plausibility = true

APPS plausibility status retrieved by frontal SCU node.

volatile bool brake\_plausibility = true

Brake plausibility status retrieved by frontal SCU node.

• volatile uint8\_t tcs\_coefficient = 0

torque limiter percentage retrieved by TCU node

## 5.3.1 Detailed Description

#### 5.3.2 Function Documentation

#### 5.3.2.1 CAN\_SERV\_BOOTUP\_CB()

This function manage boot-up messages sent over CAN servizi network by slave nodes.

#### **Author**

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniromal.it)
```

#### **Parameters**

i	n	frame	CAN frame received from CAN servizi port
---	---	-------	--

Definition at line 115 of file can\_servizi.cpp.

5.3 CAN\_servizi\_group 19

### 5.3.2.2 CAN\_SERV\_GENERAL\_CB()

This function manage PDOs received over CAN servizi network and deserializes data:

TPDO num	NODE-ID	Length	Data
			APPS1 percentage
			APPS2 percentage
1	SCU_FRONTAL_NODE_ID	4	Brake percentage
			APPS plausibility
			BRAKE plausibility
1	TCU_NODE_ID	1	TCU torque limiter

When PDOtx1 message is received from frontal SCU node then <a href="next\_pedals\_seq\_num">next\_pedals\_seq\_num</a> is incremented for keep track of last pedals message received.

#### **Author**

### Arella Matteo

(mail: arella.1646983@studenti.uniroma1.it)

#### **Parameters**

in	frame	CAN frame received from CAN servizi port
----	-------	--

Definition at line 149 of file can\_servizi.cpp.

## 5.3.2.3 can\_servizi\_go\_operational()

```
void can_servizi_go_operational ( )
```

This function send a CANOpen master NMT message for request 'go to Operational' state to CAN servizi slave nodes (SCUs and TCU).

## **Author**

### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

Definition at line 206 of file can\_servizi.cpp.

#### 5.3.2.4 can\_servizi\_init()

```
bool can_servizi_init ( )
```

This function initialize CAN servizi hardware port with baudrate CAN\_SERV\_BAUDRATE. Mailbox 0 is configured for receiving boot-up messages from CAN servizi slave nodes (filter = 0x00000700, mask = 0x1FFFF80); remaining mailboxes are configured for receiving TPDOs from CAN servizi slave nodes (filter = 0x00000080, mask = 0x1FFFFC80).

This function initializes CAN servizi network.

#### Author

### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### Return values

true	CAN servizi initialized	
false	CAN servizi not initialized	

#### **Author**

#### Arella Matteo

(mail: arella.1646983@studenti.uniromal.it)

#### Return values

true	CAN servizi network initialized successfully	
false	CAN servizi network initialization failed	

Definition at line 182 of file can\_servizi.cpp.

## 5.3.2.5 can\_servizi\_initialized()

```
volatile bool can_servizi_initialized ( )
```

This function returns CAN servizi initialization status.

#### Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

### Return values

true	CAN servizi network initialized
false	CAN servizi network not initialized

5.3 CAN\_servizi\_group 21

Definition at line 102 of file can\_servizi.cpp.

#### 5.3.2.6 can\_servizi\_online()

```
volatile bool can_servizi_online ( )
```

This function returns if CAN servizi network is online.

#### **Author**

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### Return values

true	CAN servizi network online
false	CAN servizi network offline

Definition at line 220 of file can\_servizi.cpp.

## 5.3.2.7 get\_servizi\_apps\_plausibility()

```
volatile bool get_servizi_apps_plausibility ( )
```

This function returns the value of APPS plausibility retrieved by frontal SCU node over CAN servizi network.

### Author

## Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### Return values

true	APPS plausibility
false	APPS implausibility

Definition at line 245 of file can\_servizi.cpp.

## 5.3.2.8 get\_servizi\_brake()

```
volatile uint8_t get_servizi_brake ( )
```

This function returns the value of brake pedal position sensor in percentage, retrieved by frontal SCU node over CAN servizi network.

#### Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### Returns

Brake pedal position percentage value

Definition at line 240 of file can\_servizi.cpp.

#### 5.3.2.9 get\_servizi\_brake\_plausibility()

```
volatile bool get_servizi_brake_plausibility ( )
```

This function returns the value of brake plausibility retrieved by frontal SCU node over CAN servizi network.

## **Author**

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### Return values

true	Brake plausibility
false	Brake implausibility

Definition at line 250 of file can\_servizi.cpp.

## 5.3.2.10 get\_servizi\_tps1()

```
volatile uint8_t get_servizi_tps1 ( )
```

This function returns the value of the first APPS in percentage, retrieved by frontal SCU node over CAN servizi network.

### **Author**

### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### Returns

First APPS percentage value

Definition at line 230 of file can\_servizi.cpp.

5.3 CAN\_servizi\_group 23

```
5.3.2.11 get_servizi_tps2()
```

```
volatile uint8_t get_servizi_tps2 ( )
```

This function returns the value of the second APPS in percentage, retrieved by frontal SCU node over CAN servizi network.

**Author** 

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

### Returns

Second APPS percentage value

Definition at line 235 of file can\_servizi.cpp.

```
5.3.2.12 get_tcs_torque_coefficient()
```

```
volatile uint8_t get_tcs_torque_coefficient ( )
```

This function returns the value of torque limiter percentage retrieved by TCU node over CAN servizi network.

Author

### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

## Returns

Torque limiter coefficient in percentage

Definition at line 255 of file can\_servizi.cpp.

#### 5.3.2.13 tcs\_online()

```
volatile bool tcs_online ( )
```

This function returns if TCU node is active and online on the CAN servizi network.

**Author** 

## Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### Return values

true	TCS is online and active
false	TCS is offline

Definition at line 225 of file can\_servizi.cpp.

#### 5.3.2.14 timeout()

```
void timeout ( )
```

This function is executed periodically after CAN servizi 'go Operational' NMT request is sent. When timeout occurs if next\_pedals\_seq\_num is greater than curr\_pedals\_seq\_num then frontal SCU is considered active, viceversa it is considered offline.

Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniromal.it)
```

Definition at line 93 of file can\_servizi.cpp.

#### 5.3.3 Variable Documentation

## 5.3.3.1 brake\_percentage

```
volatile uint8_t brake_percentage = 0
```

Brake pedal position sensor percentage value retrieved by frontal SCU node.

Brake pedal position sensor percentage value retrieved by analog brake signal (BRAKE\_PIN)

Definition at line 64 of file can\_servizi.cpp.

#### 5.3.3.2 tps1\_percentage

```
volatile uint8_t tps1_percentage = 0
```

First APPS percentage value retrieved by frontal SCU node.

First APPS percentage value retrieved by analog tps1 signal (TPS1\_PIN)

Definition at line 52 of file can\_servizi.cpp.

## 5.3.3.3 tps2\_percentage

```
volatile uint8_t tps2_percentage = 0
```

Second APPS percentage value retrieved by frontal SCU node.

Second APPS percentage value retrieved by analog tps2 signal (TPS2 PIN)

Definition at line 58 of file can\_servizi.cpp.

## 5.4 Board\_model\_group

#### **Macros**

- #define CAN FUNZIONALE Can1
- #define CAN\_SERVIZI Can0
- #define AIRcc 48
- #define AIRGnd 49
- #define PRE 47
- #define BUZZER 52
- #define AIRB 14
- #define RTDB 15
- #define FAN 9
- #define INVERTER\_ANALOG\_PIN DAC1
- #define BRAKE REGEN PIN DAC0
- #define TPS1 PIN A0

Pin on board dedicated to first APPS (tps1)

#define TPS1\_ADC\_CHAN\_NUM ADC\_CHER\_CH7

GPIO pin on the Atmel SAM3X8E processor corresponding to tps1 signal (AD7)

#define TPS2\_PIN A1

Pin on board dedicated to second APPS (tps2)

#define TPS2\_ADC\_CHAN\_NUM ADC\_CHER\_CH6

GPIO pin on the Atmel SAM3X8E processor corresponding to tps2 signal (AD6)

#define BRAKE\_PIN A2

Pin on board dedicated to brake pedal position sensor.

#define BRAKE ADC CHAN NUM ADC CHER CH5

GPIO pin on the Atmel SAM3X8E processor corresponding to brake signal (AD5)

#define SC\_PIN A3

Pin on board dedicated to SC read signal.

· #define SC ADC CHAN NUM ADC CHER CH4

GPIO pin on the Atmel SAM3X8E processor corresponding to SC signal (AD4)

#define ADC\_BUFFER\_SIZE 128

Size (bytes) of buffer for store each ADC channel data.

- #define BUFFERS 4
- #define ADC\_MIN 0

ADC lower bound value.

#define ADC\_MAX 4095

ADC upper bound value.

• #define APPS\_PLAUS\_RANGE 10

Size (bytes) of each ADC buffer.

#define ADC\_CHANNELS\_LIST TPS1\_ADC\_CHAN\_NUM | TPS2\_ADC\_CHAN\_NUM | BRAKE\_ADC\_CHAN\_NUM | SC ADC CHAN NUM

List of ADC channels dedicated to each board pinout.

• #define ADC\_CHANNELS 4

Number of ADC channels.

• #define TPS1\_ADC\_OFFSET 0

Offset from DMA buffer.

• #define TPS2\_ADC\_OFFSET 1

Offset from DMA buffer.

#define BRAKE\_ADC\_OFFSET 2

Offset from DMA buffer.

• #define SC\_ADC\_OFFSET 3

Offset from DMA buffer.

• #define BUFFER LENGTH ADC BUFFER SIZE \* ADC CHANNELS

Length, in bytes, of each DMA buffer.

#define TPS1\_UPPER\_BOUND 3723

First APPS max output voltage (3V)

• #define TPS1 LOWER BOUND 993

First APPS min output voltage (0.8V)

#define TPS2\_UPPER\_BOUND 1861

Second APPS max output voltage (1.5V)

• #define TPS2 LOWER BOUND 496

Second APPS min output voltage (0.4V)

• #define BRAKE UPPER BOUND 0

Brake sensor max output voltage (TODO: check Voutmax)

· #define BRAKE LOWER BOUND ADC MAX

Brake sensor min output voltage (TODO: check Voutmin)

#### **Functions**

- · void ADC Handler ()
- void model\_init ()

This function initializes hardware board.

volatile uint8\_t get\_tps1\_percentage ()

This function returns the value of the first APPS in percentage, retrieved by CAN servizi network, if online, or by analog signal.

• volatile uint8\_t get\_tps2\_percentage ()

This function returns the value of the second APPS in percentage, retrieved by CAN servizi network, if online, or by analog signal.

• volatile uint8\_t get\_brake\_percentage ()

This function returns the value of the brake pedal position sensor in percentage, retrieved by CAN servizi network, if online, or by analog signal.

· volatile bool get apps plausibility ()

This function returns the value of APPS plausibility retrieved by CAN servizi network, if online, or by analog signal.

volatile bool get\_brake\_plausibility ()

This function returns the value of brake plausibility retrieved by CAN servizi network, if online, or by analog signal.

• volatile uint16 t get SC value ()

This function returns the value of the SC.

- void model\_enable\_calibrations ()
- void model\_disable\_calibrations ()

#### **Variables**

- volatile uint8\_t tps1\_adc\_percentage = 0
- volatile uint8\_t tps2\_adc\_percentage = 0
- volatile uint8\_t brake\_adc\_percentage = 0
- volatile bool apps\_adc\_plausibility = true

APPS plausibility status retrieved by analog acquisition.

volatile bool brake adc plausibility = true

Brake plausibility status retrieved by analog acquisition.

• volatile uint16\_t tps1\_value = 0

First APPS value retrieved directly by analog tps1 signal (TPS1\_PIN) and filtered after DMA buffer is filled entirely.

• volatile uint16\_t tps2\_value = 0

Second APPS value retrieved directly by analog tps2 signal (TPS2\_PIN) and filtered after DMA buffer is filled entirely.

• volatile uint16\_t brake\_value = 0

Brake pedal position sensor value retrieved directly by analog brake signal (BRAKE\_PIN) and filtered after DMA buffer is filled entirely.

volatile uint16\_t SC\_value = 0

SC value retrieved directly by analog SC signal (SC\_PIN) and filtered after DMA buffer is filled entirely.

- volatile uint16 t tps1\_max = 3723
- volatile uint16\_t tps1\_low = 993
- volatile uint16\_t tps2\_max = 1861
- volatile uint16\_t **tps2\_low** = 496
- volatile uint16 t brake max = 0
- volatile uint16 t brake low = 4095
- volatile int bufn
- · volatile int obufn
- volatile uint16\_t buf [4][128 \*4]

DMA buffers: #BUFFERS number of buffers each of BUFFER\_LENGTH size; DMA is configured in cyclic mode: after one of #BUFFERS is filled then DMA transfer head moves to next buffer in circular indexing.

volatile bool calibrate = false

## 5.4.1 Detailed Description

#### 5.4.2 Function Documentation

## 5.4.2.1 get\_apps\_plausibility()

```
volatile bool get_apps_plausibility ( )
```

This function returns the value of APPS plausibility retrieved by CAN servizi network, if online, or by analog signal.

## Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

## Return values

true	APPS plausibility
false	APPS implausibility

Definition at line 346 of file model.cpp.

### 5.4.2.2 get\_brake\_percentage()

```
volatile uint8_t get_brake_percentage ( )
```

This function returns the value of the brake pedal position sensor in percentage, retrieved by CAN servizi network, if online, or by analog signal.

#### **Author**

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

#### Returns

Brake pedal position sensor percentage value

Definition at line 342 of file model.cpp.

## 5.4.2.3 get\_brake\_plausibility()

```
volatile bool get_brake_plausibility ( )
```

This function returns the value of brake plausibility retrieved by CAN servizi network, if online, or by analog signal.

## **Author**

### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

## Return values

true	Brake plausibility
false	Brake implausibility

Definition at line 350 of file model.cpp.

## 5.4.2.4 get\_SC\_value()

```
volatile uint16_t get_SC_value ( )
```

This function returns the value of the SC.

## Author

#### Arella Matteo

```
(mail: arella.1646983@studenti.uniroma1.it)
```

Returns

SC value

Definition at line 354 of file model.cpp.

```
5.4.2.5 get_tps1_percentage()
```

```
volatile uint8_t get_tps1_percentage ( )
```

This function returns the value of the first APPS in percentage, retrieved by CAN servizi network, if online, or by analog signal.

**Author** 

```
Arella Matteo
```

```
(mail: arella.1646983@studenti.uniroma1.it)
```

Returns

First APPS percentage value

Definition at line 334 of file model.cpp.

```
5.4.2.6 get_tps2_percentage()
```

```
volatile uint8_t get_tps2_percentage ( )
```

This function returns the value of the second APPS in percentage, retrieved by CAN servizi network, if online, or by analog signal.

Author

```
Arella Matteo
```

```
(mail: arella.1646983@studenti.uniroma1.it)
```

Returns

Second APPS percentage value

Definition at line 338 of file model.cpp.

```
5.4.2.7 model_init()
```

```
void model_init ( )
```

This function initializes hardware board.

Author

```
Arella Matteo
```

```
(mail: arella.1646983@studenti.uniromal.it)
```

Definition at line 299 of file model.cpp.

# **Chapter 6**

# **File Documentation**

# 6.1 can\_funzionale.cpp File Reference

CAN funzionale module implementation.

```
#include "can_funzionale.h"
#include "def.h"
#include <due_can.h>
#include <DueTimer.h>
```

#### **Functions**

· volatile bool can\_funzionale\_initialized ()

This function returns CAN funzionale initialization status.

• void can\_funzionale\_send\_sync ()

This function sends a periodic CANOpen sync message to inverter slave node.

void CAN\_FUNZ\_BOOTUP\_CB (CAN\_FRAME \*frame)

This function manage boot-up message sent over CAN funzionale network by inverter slave node. Upon boot-up message reception the VCU send a SDO client request for check inverter vendor ID; then inverter is considered online over CAN funzionale network.

void CAN\_FUNZ\_VENDOR\_ID\_CB (CAN\_FRAME \*frame)

This function manage SDO server response with inverter Vendor ID. VCU sends NMT operational and PDOs to enable PWM; then inverter is considered correctly configured and a timer is started for sending periodic sync messages.

• void CAN FUNZ GENERAL CB (CAN FRAME \*frame)

This function manage TPDO from inverter and deserializes data:

bool can\_funzionale\_init ()

This function initialize CAN funzionale hardware port with baudrate CAN\_FUNZ\_BAUDRATE. Mailbox 0 is configured for receiving boot-up messages from inverter slave node (filter = 0x00000700 + INVERTER\_NODE\_ID, mask = 0x1FFFFFF); mailbox 1 is configured for receiving vendorID SDO response from inverter (filter = 0x00000580 + INVERTER\_NODE\_ID, mask = 0x1FFFFFFF); remaining mailboxes are configured for receiving TPDOs from inverter slave node (filter = 0x00000080, mask = 0x1FFFFCFF).

• volatile bool can\_funzionale\_online ()

This function returns if inverter is online and active over CAN funzionale.

void inverter\_torque\_request (uint16\_t torque)

This function send torque request to inverter. If inverter is active over CAN funzionale network then the request is done via RPDO1 viceversa it's done via analog signal.

void inverter\_regen\_request (uint16\_t regen)

This function send regen request to inverter.

• volatile uint16\_t get\_torque\_actual\_value ()

This function return the torque value requested by inverter to motor retrieved from TPDO1 from inverter over CAN funzionale network.

#### **Variables**

volatile bool can\_funz\_initialized = false

CAN funzionale initialization status flag (true if initialized)

• volatile bool inverter\_online = false

Inverter online status flag (true if online)

volatile bool inverter\_configured = false

Inverter configured status flag (true if configured)

volatile uint16 t torque actual value = 0

Torque requested by inverter to motor.

# 6.1.1 Detailed Description

CAN funzionale module implementation.

**Author** 

```
Arella Matteo
```

```
(mail: arella.1646983@studenti.uniromal.it)
```

Date

2018

# 6.2 can\_funzionale.h File Reference

CAN funzionale module header.

```
#include "common.h"
```

#### **Functions**

• bool can\_funzionale\_init ()

This function initialize CAN funzionale hardware port with baudrate CAN\_FUNZ\_BAUDRATE. Mailbox 0 is configured for receiving boot-up messages from inverter slave node (filter = 0x00000700 + INVERTER\_NODE\_ID, mask = 0x1FFFFFFF); mailbox 1 is configured for receiving vendorID SDO response from inverter (filter = 0x00000580 + INVERTER\_NODE\_ID, mask = 0x1FFFFFFFF); remaining mailboxes are configured for receiving TPDOs from inverter slave node (filter = 0x00000080, mask = 0x1FFFFCFF).

volatile bool can\_funzionale\_initialized ()

This function returns CAN funzionale initialization status.

• volatile bool can\_funzionale\_online ()

This function returns if inverter is online and active over CAN funzionale.

void inverter\_torque\_request (uint16\_t torque)

This function send torque request to inverter. If inverter is active over CAN funzionale network then the request is done via RPDO1 viceversa it's done via analog signal.

• void inverter\_regen\_request (uint16\_t regen)

This function send regen request to inverter.

• volatile uint16\_t get\_torque\_actual\_value ()

This function return the torque value requested by inverter to motor retrieved from TPDO1 from inverter over CAN funzionale network.

# 6.2.1 Detailed Description

CAN funzionale module header.

Author

```
Arella Matteo
(mail: arella.1646983@studenti.uniroma1.it)

Date
2018
```

# 6.3 CAN\_ID.h File Reference

CAN nodes ID definitions module.

#### **Macros**

```
    #define VCU_NODE_ID 2
        VCU Node ID.
    #define INVERTER_NODE_ID 1
        Inverter Node ID.
    #define SCU_FRONTAL_NODE_ID 1
        Frontal SCU Node ID.
    #define TCU_NODE_ID 4
        TCU Node ID.
```

# 6.3.1 Detailed Description

CAN nodes ID definitions module.

Author

```
Arella Matteo
(mail: arella.1646983@studenti.uniromal.it)

Date
2018
```

# 6.4 can\_servizi.cpp File Reference

CAN servizi module implementation.

```
#include "can_servizi.h"
#include <due_can.h>
#include <DueTimer.h>
```

#### **Functions**

void timeout ()

This function is executed periodically after CAN servizi 'go Operational' NMT request is sent. When timeout occurs if next\_pedals\_seq\_num is greater than curr\_pedals\_seq\_num then frontal SCU is considered active, viceversa it is considered offline.

volatile bool can servizi initialized ()

This function returns CAN servizi initialization status.

void CAN\_SERV\_BOOTUP\_CB (CAN\_FRAME \*frame)

This function manage boot-up messages sent over CAN servizi network by slave nodes.

void CAN SERV GENERAL CB (CAN FRAME \*frame)

This function manage PDOs received over CAN servizi network and deserializes data:

• bool can\_servizi\_init ()

This function initialize CAN servizi hardware port with baudrate CAN\_SERV\_BAUDRATE. Mailbox 0 is configured for receiving boot-up messages from CAN servizi slave nodes (filter = 0x00000700, mask = 0x1FFFF80); remaining mailboxes are configured for receiving TPDOs from CAN servizi slave nodes (filter = 0x00000080, mask = 0x1FFF← FC80).

· void can servizi go operational ()

This function send a CANOpen master NMT message for request 'go to Operational' state to CAN servizi slave nodes (SCUs and TCU).

volatile bool can servizi online ()

This function returns if CAN servizi network is online.

volatile bool tcs online ()

This function returns if TCU node is active and online on the CAN servizi network.

volatile uint8\_t get\_servizi\_tps1 ()

This function returns the value of the first APPS in percentage, retrieved by frontal SCU node over CAN servizi network.

volatile uint8\_t get\_servizi\_tps2 ()

This function returns the value of the second APPS in percentage, retrieved by frontal SCU node over CAN servizi network.

volatile uint8\_t get\_servizi\_brake ()

This function returns the value of brake pedal position sensor in percentage, retrieved by frontal SCU node over CAN servizi network.

volatile bool get\_servizi\_apps\_plausibility ()

This function returns the value of APPS plausibility retrieved by frontal SCU node over CAN servizi network.

· volatile bool get servizi brake plausibility ()

This function returns the value of brake plausibility retrieved by frontal SCU node over CAN servizi network.

volatile uint8\_t get\_tcs\_torque\_coefficient ()

This function returns the value of torque limiter percentage retrieved by TCU node over CAN servizi network.

#### **Variables**

• volatile bool can\_serv\_initialized = false

CAN servizi initialization status flag (true if initialized)

• volatile bool SCU\_F\_online = false

Frontal SCU online status flag (true if online)

• volatile bool TCS\_online = false

TCS online status flag (true if online)

volatile uint32 t curr pedals seq num = 0

Frontal SCU PDOtx1 current sequence number.

• volatile uint32 t next pedals seq num = 0

Frontal SCU PDOtx1 next sequence number.

• volatile uint8\_t tps1\_percentage = 0

First APPS percentage value retrieved by frontal SCU node.

• volatile uint8 t tps2 percentage = 0

Second APPS percentage value retrieved by frontal SCU node.

volatile uint8\_t brake\_percentage = 0

Brake pedal position sensor percentage value retrieved by frontal SCU node.

volatile bool apps\_plausibility = true

APPS plausibility status retrieved by frontal SCU node.

volatile bool brake\_plausibility = true

Brake plausibility status retrieved by frontal SCU node.

• volatile uint8 t tcs coefficient = 0

torque limiter percentage retrieved by TCU node

#### 6.4.1 Detailed Description

CAN servizi module implementation.

**Author** 

```
Arella Matteo
```

(mail: arella.1646983@studenti.uniromal.it)

Date

2018

# 6.5 can\_servizi.h File Reference

CAN servizi module header.

```
#include "common.h"
```

### **Functions**

bool can\_servizi\_init ()

This function initialize CAN servizi hardware port with baudrate CAN\_SERV\_BAUDRATE. Mailbox 0 is configured for receiving boot-up messages from CAN servizi slave nodes (filter = 0x00000700, mask = 0x1FFFF80); remaining mailboxes are configured for receiving TPDOs from CAN servizi slave nodes (filter = 0x00000080, mask =  $0x1FFF \leftarrow FC80$ ).

· volatile bool can servizi initialized ()

This function returns CAN servizi initialization status.

void can\_servizi\_go\_operational ()

This function send a CANOpen master NMT message for request 'go to Operational' state to CAN servizi slave nodes (SCUs and TCU).

volatile bool can\_servizi\_online ()

This function returns if CAN servizi network is online.

volatile bool tcs\_online ()

This function returns if TCU node is active and online on the CAN servizi network.

• volatile uint8\_t get\_servizi\_tps1 ()

This function returns the value of the first APPS in percentage, retrieved by frontal SCU node over CAN servizi network.

volatile uint8\_t get\_servizi\_tps2 ()

This function returns the value of the second APPS in percentage, retrieved by frontal SCU node over CAN servizi network.

• volatile uint8 t get servizi brake ()

This function returns the value of brake pedal position sensor in percentage, retrieved by frontal SCU node over CAN servizi network.

• volatile bool get\_servizi\_apps\_plausibility ()

This function returns the value of APPS plausibility retrieved by frontal SCU node over CAN servizi network.

volatile bool get\_servizi\_brake\_plausibility ()

This function returns the value of brake plausibility retrieved by frontal SCU node over CAN servizi network.

volatile uint8\_t get\_tcs\_torque\_coefficient ()

This function returns the value of torque limiter percentage retrieved by TCU node over CAN servizi network.

# 6.5.1 Detailed Description

CAN servizi module header.

**Author** 

```
Arella Matteo (mail: arella.1646983@studenti.uniroma1.it)
```

Date

2018

# 6.6 CO\_can.cpp File Reference

CAN setup module implementation.

```
#include "CO_can.h"
#include "can_servizi.h"
#include "can_funzionale.h"
```

#### **Functions**

• bool can\_init ()

This function initializes both CAN funzionale and CAN servizi networks.

# 6.6.1 Detailed Description

CAN setup module implementation.

Author

```
Arella Matteo
(mail: arella.1646983@studenti.uniroma1.it)

Date
2018
```

# 6.7 CO\_can.h File Reference

CAN setup header module.

```
#include "common.h"
#include "board_pinout.h"
```

# **Functions**

• bool can\_init ()

This function initializes both CAN funzionale and CAN servizi networks.

# 6.7.1 Detailed Description

CAN setup header module.

Author

Date

```
Arella Matteo (mail: arella.1646983@studenti.uniromal.it)

2018
```

# 6.8 common.h File Reference

common macro definitions module

```
#include <stdint.h>
#include "CAN_ID.h"
#include "board_pinout.h"
```

#### **Macros**

#define CAN\_FUNZ\_BAUDRATE 1000000

Defines CAN funzionale baudrate.

• #define CAN\_SERV\_BAUDRATE 1000000

Defines CAN servizi baudrate.

• #define SERIAL\_BAUDRATE 115200

Defines serial baudrate.

#define INVERTER\_TORQUE\_MIN 0

Defines inverter torque request lower bound.

• #define INVERTER\_TORQUE\_MAX 32767

Defines inverter torque request upper bound.

• #define CAN\_FUNZ\_SYNC\_PERIOD 5000

Defines CAN funzionale sync message trasmission period.

• #define CAN\_SERVIZI\_TIMEOUT\_PERIOD 30000

Defines CAN servizi timeout period for fault check.

#### 6.8.1 Detailed Description

common macro definitions module

Author

```
Arella Matteo (mail: arella.1646983@studenti.uniromal.it)
```

Date

2018

# 6.9 model.cpp File Reference

Board model implementation file.

```
#include "model.h"
#include "filter.h"
#include "can_servizi.h"
#include <DueFlashStorage.h>
```

#### **Macros**

#define ADC\_BUFFER\_SIZE 128

Size (bytes) of buffer for store each ADC channel data.

- #define BUFFERS 4
- #define ADC MIN 0

ADC lower bound value.

• #define ADC\_MAX 4095

ADC upper bound value.

#define APPS\_PLAUS\_RANGE 10

Size (bytes) of each ADC buffer.

• #define ADC\_CHANNELS\_LIST TPS1\_ADC\_CHAN\_NUM | TPS2\_ADC\_CHAN\_NUM | BRAKE\_ADC\_CHAN\_NUM | SC\_ADC\_CHAN\_NUM

List of ADC channels dedicated to each board pinout.

#define ADC\_CHANNELS 4

Number of ADC channels.

#define TPS1\_ADC\_OFFSET 0

Offset from DMA buffer.

#define TPS2\_ADC\_OFFSET 1

Offset from DMA buffer.

• #define BRAKE ADC OFFSET 2

Offset from DMA buffer.

• #define SC\_ADC\_OFFSET 3

Offset from DMA buffer.

#define BUFFER LENGTH ADC BUFFER SIZE \* ADC CHANNELS

Length, in bytes, of each DMA buffer.

• #define TPS1\_UPPER\_BOUND 3723

First APPS max output voltage (3V)

#define TPS1\_LOWER\_BOUND 993

First APPS min output voltage (0.8V)

• #define TPS2\_UPPER\_BOUND 1861

Second APPS max output voltage (1.5V)

#define TPS2\_LOWER\_BOUND 496

Second APPS min output voltage (0.4V)

• #define BRAKE UPPER BOUND 0

Brake sensor max output voltage (TODO: check Voutmax)

• #define BRAKE\_LOWER\_BOUND ADC\_MAX

Brake sensor min output voltage (TODO: check Voutmin)

#### **Functions**

- void ADC\_Handler ()
- void model\_init ()

This function initializes hardware board.

volatile uint8\_t get\_tps1\_percentage ()

This function returns the value of the first APPS in percentage, retrieved by CAN servizi network, if online, or by analog signal.

volatile uint8\_t get\_tps2\_percentage ()

This function returns the value of the second APPS in percentage, retrieved by CAN servizi network, if online, or by analog signal.

volatile uint8\_t get\_brake\_percentage ()

This function returns the value of the brake pedal position sensor in percentage, retrieved by CAN servizi network, if online, or by analog signal.

volatile bool get\_apps\_plausibility ()

This function returns the value of APPS plausibility retrieved by CAN servizi network, if online, or by analog signal.

· volatile bool get brake plausibility ()

This function returns the value of brake plausibility retrieved by CAN servizi network, if online, or by analog signal.

volatile uint16\_t get\_SC\_value ()

This function returns the value of the SC.

#### **Variables**

- volatile uint8\_t tps1\_adc\_percentage = 0
- volatile uint8\_t tps2\_adc\_percentage = 0
- volatile uint8 t brake adc percentage = 0
- volatile bool apps\_adc\_plausibility = true

APPS plausibility status retrieved by analog acquisition.

volatile bool brake\_adc\_plausibility = true

Brake plausibility status retrieved by analog acquisition.

• volatile uint16 t tps1 value = 0

First APPS value retrieved directly by analog tps1 signal (TPS1\_PIN) and filtered after DMA buffer is filled entirely.

• volatile uint16\_t tps2\_value = 0

Second APPS value retrieved directly by analog tps2 signal (TPS2\_PIN) and filtered after DMA buffer is filled entirely.

volatile uint16\_t brake\_value = 0

Brake pedal position sensor value retrieved directly by analog brake signal (BRAKE\_PIN) and filtered after DMA buffer is filled entirely.

volatile uint16\_t SC\_value = 0

SC value retrieved directly by analog SC signal (SC\_PIN) and filtered after DMA buffer is filled entirely.

- volatile uint16 t **tps1 max** = 3723
- volatile uint16 t tps1\_low = 993
- volatile uint16 t tps2 max = 1861
- volatile uint16\_t tps2\_low = 496
- volatile uint16 t brake max = 0
- volatile uint16\_t brake\_low = 4095
- volatile int bufn
- · volatile int obufn
- volatile uint16 t buf [4][128 \*4]

DMA buffers: #BUFFERS number of buffers each of BUFFER\_LENGTH size; DMA is configured in cyclic mode: after one of #BUFFERS is filled then DMA transfer head moves to next buffer in circular indexing.

• volatile bool calibrate = false

# 6.9.1 Detailed Description

Board model implementation file.

#### Author

```
Arella Matteo
```

```
(mail: arella.1646983@studenti.uniroma1.it)
```

Date

2018

6.10 model.h File Reference 41

#### 6.10 model.h File Reference

Board model header file.

```
#include <Arduino.h>
```

#### **Functions**

• void model\_init ()

This function initializes hardware board.

• volatile uint8\_t get\_tps1\_percentage ()

This function returns the value of the first APPS in percentage, retrieved by CAN servizi network, if online, or by analog signal.

volatile uint8\_t get\_tps2\_percentage ()

This function returns the value of the second APPS in percentage, retrieved by CAN servizi network, if online, or by analog signal.

volatile uint8\_t get\_brake\_percentage ()

This function returns the value of the brake pedal position sensor in percentage, retrieved by CAN servizi network, if online, or by analog signal.

volatile bool get\_apps\_plausibility ()

This function returns the value of APPS plausibility retrieved by CAN servizi network, if online, or by analog signal.

volatile bool get\_brake\_plausibility ()

This function returns the value of brake plausibility retrieved by CAN servizi network, if online, or by analog signal.

volatile uint16\_t get\_SC\_value ()

This function returns the value of the SC.

- void model\_enable\_calibrations ()
- void model\_disable\_calibrations ()

# 6.10.1 Detailed Description

Board model header file.

**Author** 

```
Arella Matteo (mail: arella.1646983@studenti.uniroma1.it)
```

Date

2018

# 6.11 VCU.ino File Reference

Main module file.

```
#include "common.h"
#include "CO_can.h"
#include "can_servizi.h"
#include "model.h"
#include "states.h"
```

#### **Functions**

void setup ()

This function perform basic board setup.

• void loop ()

This function is called into endless while main loop. It takes care of dispatching states of the finite state machine (TODO: see states)

#### 6.11.1 Detailed Description

Main module file.

Author

Date

```
Arella Matteo (mail: arella.1646983@studenti.uniromal.it)

2018
```

#### 6.11.2 Function Documentation

```
6.11.2.1 loop()
void loop ( )
```

This function is called into endless while main loop. It takes care of dispatching states of the finite state machine (TODO: see states)

**Author** 

```
Arella Matteo (mail: arella.1646983@studenti.uniroma1.it)
```

Definition at line 63 of file VCU.ino.

```
6.11.2.2 setup() void setup ( )
```

This function perform basic board setup.

- It starts initializing both CAN funzionale (with inverter) and CAN servizi (with the two SCUs and TCS); if the comunication between inverter and VCU can't be established via CAN bus then the VCU is configured to request torque value to inverter by analog signal.
- It initializes board hardware (TODO: see model)
- If the configuration over CAN servizi with the frontal SCU was successful then VCU (master) send an NMT request to go in 'Operational' state (TODO: see later).

Author

```
Arella Matteo (mail: arella.1646983@studenti.uniroma1.it)
```

Definition at line 43 of file VCU.ino.

# Index

Board_model_group, 25	timeout, 24
get SC value, 28	tps1_percentage, 24
get_apps_plausibility, 27	tps2 percentage, 24
get_brake_percentage, 27	CO_can.cpp, 36
get_brake_plausibility, 28	CO_can.h, 37
get tps1 percentage, 29	can_funzionale.cpp, 31
get_tps2_percentage, 29	can_funzionale.h, 32
model_init, 29	can_funzionale_init
brake_percentage	CAN_funzionale_group, 13
CAN_servizi_group, 24	can_funzionale_initialized
5 1/	CAN_funzionale_group, 14
CAN_FUNZ_BOOTUP_CB	can_funzionale_online
CAN_funzionale_group, 12	CAN_funzionale_group, 14
CAN_FUNZ_GENERAL_CB	can_funzionale_send_sync
CAN_funzionale_group, 12	CAN_funzionale_group, 15
CAN_FUNZ_VENDOR_ID_CB	can_init
CAN_funzionale_group, 13	CAN_module_group, 9
CAN_ID.h, 33	can_servizi.cpp, 33
CAN_SERV_BOOTUP_CB	can_servizi.h, 35
CAN_servizi_group, 18	can_servizi_go_operational
CAN_SERV_GENERAL_CB	CAN_servizi_group, 19
CAN_servizi_group, 19	can_servizi_init
CAN_funzionale_group, 11	CAN_servizi_group, 19
CAN_FUNZ_BOOTUP_CB, 12	can servizi initialized
CAN_FUNZ_GENERAL_CB, 12	CAN_servizi_group, 20
CAN_FUNZ_VENDOR_ID_CB, 13	can_servizi_online
can_funzionale_init, 13	CAN_servizi_group, 21
can_funzionale_initialized, 14	common.h, 37
can_funzionale_online, 14	· · · · · · · · · · · · · · · · · · ·
can_funzionale_send_sync, 15	get_SC_value
get_torque_actual_value, 15	Board_model_group, 28
inverter_regen_request, 15	get_apps_plausibility
inverter torque request, 16	Board model group, 27
CAN_module_group, 9	get_brake_percentage
can init, 9	Board model group, 27
CAN servizi group, 17	get_brake_plausibility
brake_percentage, 24	Board_model_group, 28
CAN_SERV_BOOTUP_CB, 18	get_servizi_apps_plausibility
CAN_SERV_GENERAL_CB, 19	CAN_servizi_group, 21
can_servizi_go_operational, 19	get_servizi_brake
can_servizi_init, 19	CAN_servizi_group, 21
can_servizi_initialized, 20	get_servizi_brake_plausibility
can_servizi_online, 21	CAN servizi group, 22
get_servizi_apps_plausibility, 21	get_servizi_tps1
get_servizi_brake, 21	CAN_servizi_group, 22
get_servizi_brake_plausibility, 22	get_servizi_tps2
get_servizi_tps1, 22	CAN_servizi_group, 22
get servizi tps2, 22	get_tcs_torque_coefficient
get_tcs_torque_coefficient, 23	CAN_servizi_group, 23
tcs online, 23	get torque actual value

44 INDEX

```
CAN_funzionale_group, 15
get_tps1_percentage
    Board_model_group, 29
get_tps2_percentage
    Board_model_group, 29
inverter_regen_request
    CAN_funzionale_group, 15
inverter_torque_request
    CAN_funzionale_group, 16
loop
    VCU.ino, 42
model.cpp, 38
model.h, 41
model_init
    Board_model_group, 29
setup
    VCU.ino, 42
tcs_online
    CAN_servizi_group, 23
timeout
    CAN_servizi_group, 24
tps1_percentage
    CAN_servizi_group, 24
tps2_percentage
    CAN_servizi_group, 24
VCU.ino, 41
    loop, 42
    setup, 42
```