Vehicle Control Unit (VCU)

Generated by Doxygen 1.8.5

Fri Jul 6 2018 12:25:14

Contents

1 FastChargeSAE VCU firmware 1

2 CAN Networks 3

3 Module Index 5

3.1 Modules 5

4 File Index 7

4.1 File List 7

5 Module Documentation 9

5.1 CAN_module_group 9

5.1.1 Detailed Description 9

5.1.2 Function Documentation 9

5.1.2.1 can_init 9

5.2 CAN_funzionale_group 10

5.2.1 Detailed Description 11

5.2.2 Function Documentation 11

5.2.2.1 CAN_FUNZ_BOOTUP_CB 11

5.2.2.2 CAN_FUNZ_GENERAL_CB 11

5.2.2.3 CAN_FUNZ_VENDOR_ID_CB 11

5.2.2.4 can_funzionale_init 11

5.2.2.5 can_funzionale_initialized 12

5.2.2.6 can_funzionale_online 12

5.2.2.7 can_funzionale_send_sync 13

5.2.2.8 get_torque_actual_value 13

5.2.2.9 inverter_regen_request 13

5.2.2.10 inverter_torque_request 14

5.3 CAN_servizi_group 15

5.3.1 Detailed Description 16

5.3.2 Function Documentation 16

5.3.2.1 CAN_SERV_BOOTUP_CB 16

5.3.2.2 CAN_SERV_GENERAL_CB 16

5.3.2.3 can_servizi_go_operational 17

5.3.2.4 can_servizi_init 17

5.3.2.5 can_servizi_initialized 18

5.3.2.6 can_servizi_online 18

5.3.2.7 get_servizi_apps_plausibility 18

5.3.2.8 get_servizi_brake 18

5.3.2.9 get_servizi_brake_plausibility 19

5.3.2.10 get_servizi_tps1 19

5.3.2.11 get_servizi_tps2 19

5.3.2.12 get_tcs_torque_coefficient 20

5.3.2.13 tcs_online 20

5.3.2.14 timeout 20

5.3.3 Variable Documentation 20

5.3.3.1 brake_percentage 20

5.3.3.2 tps1_percentage 20

5.3.3.3 tps2_percentage 21

5.4 Board_model_group 22

5.4.1 Detailed Description 24

5.4.2 Function Documentation 24

5.4.2.1 get_apps_plausibility 24

5.4.2.2 get_brake_percentage 25

5.4.2.3 get_brake_plausibility 25

5.4.2.4 get_SC_value 25

5.4.2.5 get_tps1_percentage 25

5.4.2.6 get_tps2_percentage 26

5.4.2.7 model_init 26

6 File Documentation 27

6.1 board_pinout.h File Reference 27

6.1.1 Detailed Description 28

6.2 can_funzionale.cpp File Reference 28

6.2.1 Detailed Description 29

6.3 can_funzionale.h File Reference 29

6.3.1 Detailed Description 30

6.4 CAN_ID.h File Reference 30

6.4.1 Detailed Description 31

6.5 can_servizi.cpp File Reference 31

6.5.1 Detailed Description 32

6.6 can_servizi.h File Reference 33

6.6.1 Detailed Description 33

6.7 CO_can.cpp File Reference 34
6.7.1 Detailed Description 34
6.8 CO_can.h File Reference 34
6.8.1 Detailed Description 34
6.9 common.h File Reference 35
6.9.1 Detailed Description 35
6.10 model.cpp File Reference 35
6.10.1 Detailed Description 37
6.11 model.h File Reference 38
6.11.1 Detailed Description 38
6.12 VCU.ino File Reference 38
6.12.1 Detailed Description 39
6.12.2 Function Documentation 39
6.12.2.1 loop 39
6.12.2.2 setup 39

Index **40**

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).

Chapter 3

Module Index

3.1 Modules

Here is a list of all modules:

CAN_module_group 9

  CAN_funzionale_group 10

  CAN_servizi_group 15

Board_model_group 22

Chapter 4

File Index

4.1 File List

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

board_pinout.h
Board pinout module header 27
can_funzionale.cpp
CAN funzionale module implementation 28
can_funzionale.h
CAN funzionale module header 29
CAN_ID.h
CAN nodes ID definitions module 30
can_servizi.cpp
CAN servizi module implementation 31
can_servizi.h
CAN servizi module header 33
CO_can.cpp
CAN setup module implementation 34
CO_can.h
CAN setup header module 34
common.h
Common macro definitions module 35
def.h . . . . **??**
filter.cpp . . . . **??**
filter.h . . . . **??**
model.cpp
Board model implementation file 35
model.h
Board model header file 38
rtds.cpp . . . . **??**
rtds.h . . . . **??**
states.cpp . . . . **??**
states.h . . . . **??**
VCU.ino
Main module file 38

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 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 = 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.

5.2.1 Detailed Description

5.2.2 Function Documentation

5.2.2.1 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.

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 void CAN_FUNZ_GENERAL_CB (CAN_FRAME ∗ *frame*)

This function manage TPDO from inverter and deserializes data:

TPDO num	NODE-ID	Data
1	INVERTER_NODE_ID	Torque Actual Val

Author

Arella Matteo
(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 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.

Author

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

Parameters

`in`	*frame*	CAN frame received from CAN servizi port

Definition at line 108 of file can_funzionale.cpp.

5.2.2.4 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.uniroma1.it)

Return values

true	CAN servizi initialized
false	CAN servizi not initialized

Author

Arella Matteo
(mail: arella.1646983@studenti.uniroma1.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 volatile bool can_funzionale_initialized ( )

This function returns CAN funzionale initialization status.

Author

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

Return values

true	CAN funzionale network initialized
false	CAN funzionale network not initialized

Definition at line 44 of file can_funzionale.cpp.

5.2.2.6 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 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 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 void inverter_regen_request (uint16_t *regen*)

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 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.

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

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 = 0x1FFFFF80); remaining mailboxes are configured for receiving TPDOs from CAN servizi slave nodes (filter = 0x00000080, mask = 0x1FFFFC80).
- 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 void CAN_SERV_BOOTUP_CB (CAN_FRAME ∗ *frame*)

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

Author

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

Parameters

`in`	*frame*	CAN frame received from CAN servizi port

Definition at line 115 of file can_servizi.cpp.

5.3.2.2 void CAN_SERV_GENERAL_CB (CAN_FRAME ∗ *frame*)

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

TPDO num	NODE-ID	Length	Data
1	SCU_FRONTAL_NODE_ID	4	APPS1 percentage
			APPS2 percentage

			Brake percentage
			APPS plausibility
			BRAKE plausibility
1	TCU_NODE_ID	1	TCU torque limiter

When PDOtx1 message is received from frontal SCU node then next_pedals_seq_num 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 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 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 = 0x1FFFFF80); 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.uniroma1.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 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

Definition at line 102 of file can_servizi.cpp.

5.3.2.6 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 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 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 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 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.

5.3.2.11 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 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 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 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.uniroma1.it`)

Definition at line 93 of file can_servizi.cpp.

5.3.3 Variable Documentation

5.3.3.1 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 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 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

  Pin on board dedicated to CAN funzionale port.
- #define CAN_SERVIZI Can0

  Pin on board dedicated to CAN servizi port.
- #define AIRcc 48

  Pin on board dedicated to AIR+.
- #define AIRGnd 49

  Pin on board dedicated to AIR-.
- #define PRE 47

  Pin on board dedicated to PRECHARGE.
- #define BUZZER 52

  Pin on board dedicated to buzzer for RTDS.
- #define AIRB 14

  Pin on board dedicated to AIR button.
- #define RTDB 15

  Pin on board dedicated to RTD button.
- #define FAN 9

  Pin on board dedicated to FAN.
- #define INVERTER_ANALOG_PIN DAC1

  Pin on board dedicated to inverter torque request analog signal.
- #define BRAKE_REGEN_PIN DAC0

  Pin on board dedicated to inverter regen request analog signal.
- #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 2482

  First APPS max output voltage (2V)
- #define TPS1_LOWER_BOUND 993

  First APPS min output voltage (0.8V)
- #define TPS2_UPPER_BOUND 1241

  Second APPS max output voltage (1V)
- #define TPS2_LOWER_BOUND 497

  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** = 2482
- volatile uint16_t **tps1_low** = 993
- volatile uint16_t **tps2_max** = 1241
- volatile uint16_t **tps2_low** = 497
- 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 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 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 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 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 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 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 void model_init ( )

This function initializes hardware board.

Author

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

Definition at line 299 of file model.cpp.

Chapter 6

File Documentation

6.1 board_pinout.h File Reference

Board pinout module header.

Macros

- #define CAN_FUNZIONALE Can1

  Pin on board dedicated to CAN funzionale port.
- #define CAN_SERVIZI Can0

  Pin on board dedicated to CAN servizi port.
- #define AIRcc 48

  Pin on board dedicated to AIR+.
- #define AIRGnd 49

  Pin on board dedicated to AIR-.
- #define PRE 47

  Pin on board dedicated to PRECHARGE.
- #define BUZZER 52

  Pin on board dedicated to buzzer for RTDS.
- #define AIRB 14

  Pin on board dedicated to AIR button.
- #define RTDB 15

  Pin on board dedicated to RTD button.
- #define FAN 9

  Pin on board dedicated to FAN.
- #define INVERTER_ANALOG_PIN DAC1

  Pin on board dedicated to inverter torque request analog signal.
- #define BRAKE_REGEN_PIN DAC0

  Pin on board dedicated to inverter regen request analog signal.
- #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)

6.1.1 Detailed Description

Board pinout module header.

Author

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

Date

2018

Definition in file board_pinout.h.

6.2 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 = 0x1-FFFFFFF); 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.

6.2.1 Detailed Description

CAN funzionale module implementation.

Author

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

Date

2018

Definition in file can_funzionale.cpp.

6.3 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 = 0x1FFFFFFF); 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.

6.3.1 Detailed Description

CAN funzionale module header.

Author

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

Date

2018

Definition in file can_funzionale.h.

6.4 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.

6.4.1 Detailed Description

CAN nodes ID definitions module.

Author

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

Date

2018

Definition in file CAN_ID.h.

6.5 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 = 0x1FFFFF80); remaining mailboxes are configured for receiving TPDOs from CAN servizi slave nodes (filter = 0x00000080, mask = 0x1FFFFC80).
- 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

6.5.1 Detailed Description

CAN servizi module implementation.

Author

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

Date

2018

Definition in file can_servizi.cpp.

6.6 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 = 0x1FFFFF80); remaining mailboxes are configured for receiving TPDOs from CAN servizi slave nodes (filter = 0x00000080, mask = 0x1FFFFC80).
- 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.

6.6.1 Detailed Description

CAN servizi module header.

Author

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

Date

2018

Definition in file can_servizi.h.

6.7 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.

6.7.1 Detailed Description

CAN setup module implementation.

Author

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

Date

2018

Definition in file CO_can.cpp.

6.8 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.

6.8.1 Detailed Description

CAN setup header module.

Author

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

Date

2018

Definition in file CO_can.h.

6.9 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.

6.9.1 Detailed Description

common macro definitions module

Author

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

Date

2018

Definition in file common.h.

6.10 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 2482

  First APPS max output voltage (2V)
- #define TPS1_LOWER_BOUND 993

  First APPS min output voltage (0.8V)
- #define TPS2_UPPER_BOUND 1241

  Second APPS max output voltage (1V)
- #define TPS2_LOWER_BOUND 497

  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** = 2482
- volatile uint16_t **tps1_low** = 993
- volatile uint16_t **tps2_max** = 1241
- volatile uint16_t **tps2_low** = 497
- 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

6.10.1 Detailed Description

Board model implementation file.

Author

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

Date

2018

Definition in file model.cpp.

6.11 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** ()

6.11.1 Detailed Description

Board model header file.

Author

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

Date

2018

Definition in file model.h.

6.12 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)

6.12.1 Detailed Description

Main module file.

Author

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

Date

2018

Definition in file VCU.ino.

6.12.2 Function Documentation

6.12.2.1 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.

6.12.2.2 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, 22
    get_SC_value, 25
    get_apps_plausibility, 24
    get_brake_percentage, 24
    get_brake_plausibility, 25
    get_tps1_percentage, 25
    get_tps2_percentage, 25
    model_init, 26
board_pinout.h, 27
brake_percentage
    CAN_servizi_group, 20

CAN_FUNZ_BOOTUP_CB
    CAN_funzionale_group, 11
CAN_FUNZ_GENERAL_CB
    CAN_funzionale_group, 11
CAN_FUNZ_VENDOR_ID_CB
    CAN_funzionale_group, 11
CAN_ID.h, 30
CAN_SERV_BOOTUP_CB
    CAN_servizi_group, 16
CAN_SERV_GENERAL_CB
    CAN_servizi_group, 16
CAN_funzionale_group, 10
    CAN_FUNZ_BOOTUP_CB, 11
    CAN_FUNZ_GENERAL_CB, 11
    CAN_FUNZ_VENDOR_ID_CB, 11
    can_funzionale_init, 11
    can_funzionale_initialized, 12
    can_funzionale_online, 12
    can_funzionale_send_sync, 13
    get_torque_actual_value, 13
    inverter_regen_request, 13
    inverter_torque_request, 14
CAN_module_group, 9
    can_init, 9
CAN_servizi_group, 15
    brake_percentage, 20
    CAN_SERV_BOOTUP_CB, 16
    CAN_SERV_GENERAL_CB, 16
    can_servizi_go_operational, 17
    can_servizi_init, 17
    can_servizi_initialized, 18
    can_servizi_online, 18
    get_servizi_apps_plausibility, 18
    get_servizi_brake, 18
    get_servizi_brake_plausibility, 19
    get_servizi_tps1, 19
    get_servizi_tps2, 19
    get_tcs_torque_coefficient, 19
    tcs_online, 20
    timeout, 20
    tps1_percentage, 20
    tps2_percentage, 20
CO_can.cpp, 34
CO_can.h, 34
can_funzionale.cpp, 28
can_funzionale.h, 29
can_funzionale_init
    CAN_funzionale_group, 11
can_funzionale_initialized
    CAN_funzionale_group, 12
can_funzionale_online
    CAN_funzionale_group, 12
can_funzionale_send_sync
    CAN_funzionale_group, 13
can_init
    CAN_module_group, 9
can_servizi.cpp, 31
can_servizi.h, 33
can_servizi_go_operational
    CAN_servizi_group, 17
can_servizi_init
    CAN_servizi_group, 17
can_servizi_initialized
    CAN_servizi_group, 18
can_servizi_online
    CAN_servizi_group, 18
common.h, 35

get_SC_value
    Board_model_group, 25
get_apps_plausibility
    Board_model_group, 24
get_brake_percentage
    Board_model_group, 24
get_brake_plausibility
    Board_model_group, 25
get_servizi_apps_plausibility
    CAN_servizi_group, 18
get_servizi_brake
    CAN_servizi_group, 18
get_servizi_brake_plausibility
    CAN_servizi_group, 19
get_servizi_tps1
    CAN_servizi_group, 19
get_servizi_tps2
    CAN_servizi_group, 19
get_tcs_torque_coefficient
    CAN_servizi_group, 19

get_torque_actual_value
    CAN_funzionale_group, 13
get_tps1_percentage
    Board_model_group, 25
get_tps2_percentage
    Board_model_group, 25

inverter_regen_request
    CAN_funzionale_group, 13
inverter_torque_request
    CAN_funzionale_group, 14

loop
    VCU.ino, 39

model.cpp, 35
model.h, 38
model_init
    Board_model_group, 26

setup
    VCU.ino, 39

tcs_online
    CAN_servizi_group, 20
timeout
    CAN_servizi_group, 20
tps1_percentage
    CAN_servizi_group, 20
tps2_percentage
    CAN_servizi_group, 20

VCU.ino, 38
    loop, 39
    setup, 39