CAN - CAR

TABLE OF CONTENT

CAN - CAR

1 OVERVIEW

1.1 Functionalities

The CAN-CAR system is an interface offering the ability to drive the motor of an automatic car through the CAN bus.

This hardware could be used in different laboratories such as Sin, SEm, PTR.

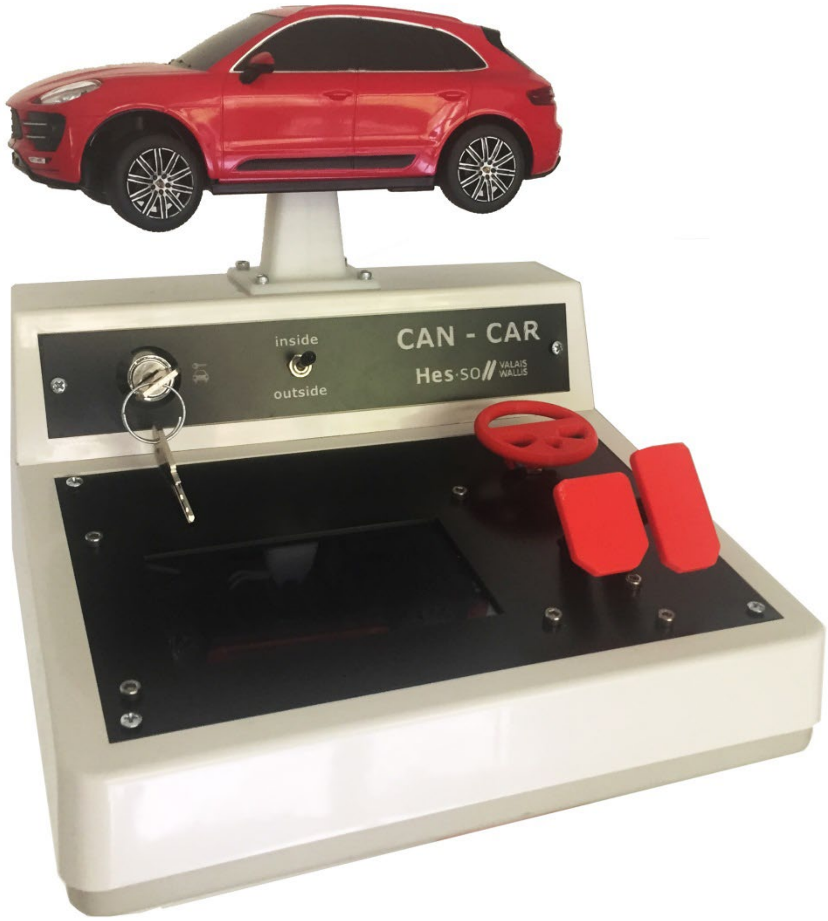


Figure 1: The CAN-CAR

This document intends to explain in detail the functionalities of this model.

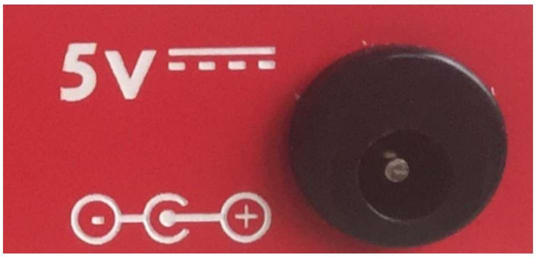
2 CAN-CAR HARDWARE

This chapter describes all the parts attached to this module.

2.1 The connections

2.1.1 POWER SUPPLY connection

The back of the model offers an DC10B connector to supply the system.



Supply connector

The model requests a supply of 5 volts with a current of 2 amperes maximum.

2.1.2 CAN connections

The back of the model offers two RJ-45 connectors for the CAN bus protocol.

CAN connectors

These two connectors are totally identical. There is no termination resistor on the model. The connections are always the same used in the school as shown.

RJ-45 pin number	***Signal description***
1	***-***
2	***-***
3	***-***
4	***CANH***
5	***CANL***
6	***-***
7, 8	***GND***

2.1.3 uSD connections

The back of the model offers one micro SD card connector. On this place a card could be inserted to have more functions.

uSD card connector

2.1.4 Car programming USB connector

The back of the model offers one USB mini connector. This is only useful to update the car firmware.

USB mini connector

2.2 The user interface

2.2.1 Contact key

The contact key is just a state given in the CAN protocol.

The contact key

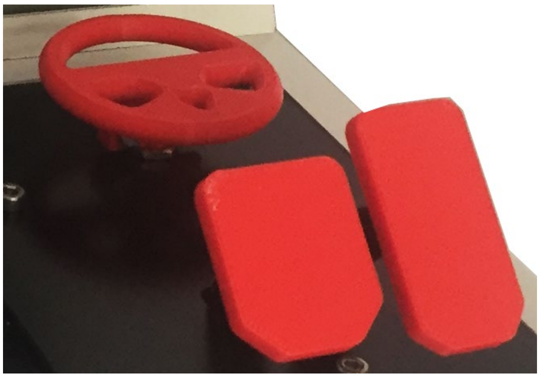
2.2.2 LCD view choice

The switch at the center called “inside-outside” offers the ability to choose what is visible on the user screen.

LCD view switch

2.2.3 Driving user parts

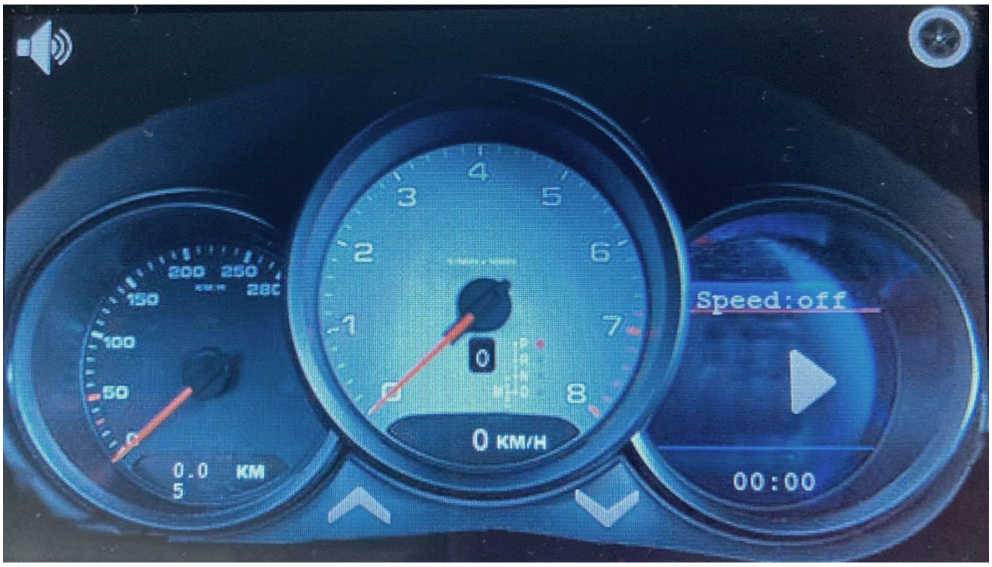
The user has a steering wheel, a brake pedal and an accelerator pedal. All these parts return their state on the CAN protocol.



User driving parts

2.2.4 LCD screen (inside view)

Depending on the LCD view choice, the LCD internal view is as below.



The user inputs (through the touchscreen) are:

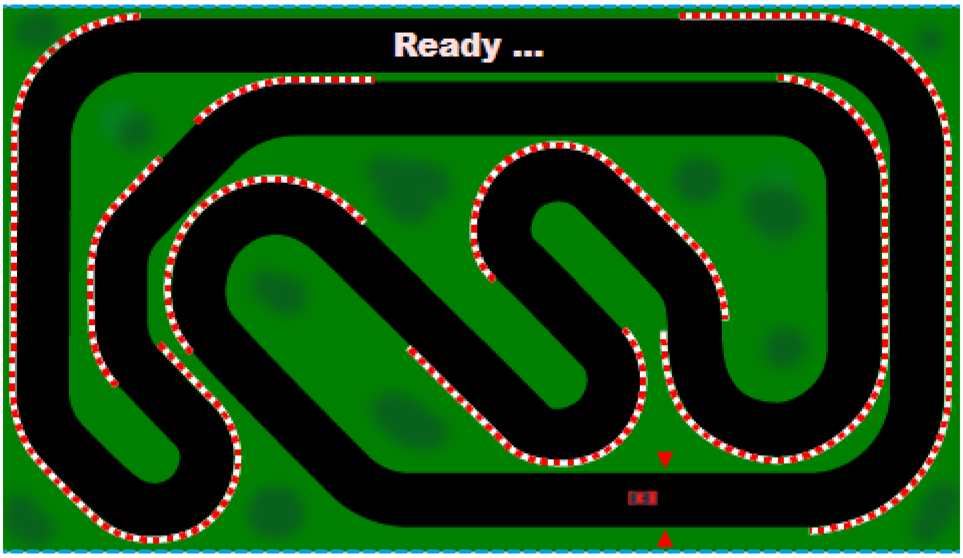
- The ‘play’ button (tempomat ON/OFF) who takes the current speed
- The ‘Up’,and ‘Down’ buttons (the gear chosen Park, Reverse, Neutral and Drive)
- The loudspeaker and wheel buttons (to have a quiet mode)

The user outputs (shown on the screen) are:

- The RPM needle (0 to 8000)
- The speed needle (0 to 280)
- The trip kilometers
- The car total kilometers
- The clock
- The current mode of drive (P R N D) with a red dot
- The current automatic gear engaged (0 to 5)

2.2.5 LCD screen (outside view)

Depending on the LCD view choice, the LCD external view is as below.



On this view, there is no user touch input. Only information can be displayed on the screen.

2.2.6 The car

The car itself has the following features:

- Headlights (off, low beam, high beam or any between)
- Backlights (off, low beam and brake light or any between)
- Front sensor (sensing of an object in front of the car)
- Wheels (turn as fast as the car speed, in both directions)
- Internal motor sound (depending on motor RPM)
- Slope (to simulate the car on a slope)

The car

3 CAR BEHAVIOUR

3.1 The park mode ‘P’

In the park mode (default on power on), the car is blocked, and it cannot move. To be able to change this mode, the contact key must be turned on.

When the contact key is turned on, the motor must be driven through the CAN protocol

3.2 The neutral mode ‘N’

In the neutral mode, the car wheels are totally free. It means if there is a slope, the car will move in the corresponding direction.

3.3 The reverse mode ‘R’ and the drive mode ‘D’

In the reverse or drive mode, the car could be driven if the automatic gear is not 0. The gear 0 means the same behavior as the neutral mode. In the reverse mode, only the gear 1 can be used. In the drive mode, 5 gears level are available.

4 CAN MESSAGES

This chapter describes all the CAN messages in details. The comprehension of this chapter is very important to be able to drive the car motor.

The CAN must be configured at a frequency of 250KB. All CAN messages are 11 bits length.

4.1 Messages from car to external controller

4.1.1 TEMPOMAT

Message name	**TEMPOMAT**
CAN ID	`(0x1 << 4)	CAR_ID`
Data[0]	The value ‘0’ means no tempomat, the value ‘1’ means tempomat activated
Data[1]	The tempomat speed chosen (speed when button pressed)
Description	Indication of the current state of the button ‘T’ tempomat of the car
Event Type	EVT (message is sent on button change)

The state of this button must be used to drive the speed automatically at the chosen speed indicated on the second data byte.

4.1.2 GEAR_SEL

Message name	GEAR_SEL
CAN ID	`(0x2 << 4) | CAR_ID`
Data[0]	The value 'P', 'R', 'N', or 'D' depending on the user gear choice
Description	Indication of the current state of the gear mode of the car
Event Type	EVT (message is sent on button change)

The state of this gear has to be used to drive the motor depending on the user choice.

4.1.3 FRONT_SENS_REQ

Message name	FRONT_SENS_REQ
CAN ID	`(0x3 << 4) | CAR_ID`
Data[0]	FRONT_H
Data[1]	FRONT_L
Description	Indication of an obstacle in front of the car. This 16 bits value is near from '0' when nothing is visible in front, and this value increases up to 0xFFF when the obstacle is near.
Event Type	RR (message is sent only after a remote request)

The usage of this sensor could be multiple, for examples, the headlights can be changed when something is in front of the car, or the car can brake automatically when an obstacle is visible.

4.1.4 EXT_SENSORS

Message name	EXT_SENSORS
CAN ID	`(0x3 << 4) | CAR_ID`
Data[0]	Front left
Data[1]	Front right
Description	Indication of an obstacle in front of the car at 30 degree from the left and right of the car, in "car race mode" only. These two 8 bits values (from 0 to 63) inform the controller the distance between the car and the track border. The value 0 means the car has crashed.
Event Type	EVT (message is sent when any sensor value has changed)

The usage of these sensors offers the ability to drive automatically the car in the "race" mode by controlling the speed and the steering wheel of the car.

4.1.5 MOTOR_STATUS

Message name	**MOTOR_STATUS**
CAN ID	`(0x4 << 4)	CAR_ID`
Data[0] – Data[1]	Motor RPM (16 bits MSB, LSB)
Data[2] – Data[3]	Motor speed (16 bits signed, MSB, LSB)
Description	Indication of the car status. The RPM of the motor is sent every 50 ms if the value has changed more than 50 RPM. The speed is sent too if changed minimum of 2 km/h
Event Type	EVT/TT (message is sent each 50 ms maximum only if values have changed enough)

The usage of this information's is absolutely necessary to drive the car.

4.1.6 BRAKE_PEDAL

Message name	**BRAKE_PEDAL**
CAN ID	`(0x6 << 4)	CAR_ID`
Data[0]	Brake pedal factor (0% to 100%)
Description	Indication of the current brake pedal position. The value 0 means the pedal is not pressed.
Event Type	EVT (message is sent when change is of minimum 6%)

The brake pedal doesn't control directly the brake of the car. This information will be directly used to set the PWR_BRAKE factor that controls the brake of the car.

4.1.7 ACCEL_PEDAL

Message name	**ACCEL_PEDAL**
CAN ID	`(0x7 << 4)	CAR_ID`
Data[0]	Accelerator pedal factor (0% to 100%)
Description	Indication of the current accelerator pedal position. The value 0 means the pedal is not pressed.
Event Type	EVT (message is sent when change is of minimum 6%)

The accelerator pedal doesn't control directly the motor RPM of the car. This information will be directly used to set the PWR_MOTOR factor that controls the motor RPM of the car.

4.1.8 CONTACT_KEY

Message name	**CONTACT_KEY**
CAN ID	`(0x8 << 4) | CAR_ID`
Data[0]	The current key position (0 is contact off, 1 is contact on)
Description	Indication of the current contact key position.
Event Type	EVT (message is sent on contact change)

When the contact key is turned on, the motor must be driven (PWR_MOTOR).

4.1.9 STEERING_W_REQ

Message name	**STEERING_W_REQ**
CAN ID	`(0x9 << 4) | CAR_ID`
Data[0]	Steering wheel position (-100/left to 100/right)
Description	Indication of the steering wheel position. The value 0 means the car goes forward.
Event Type	RR (message is sent only after a remote request)

The usage of this information is not useful.

4.1.10 BROKEN_CAR

Message name	**BROKEN_CAR**
CAN ID	`(0xA << 4) | CAR_ID`
Data[0]	Broken value indication code
Description	Indication of the car has broken for any reason.
Event Type	EVT (message is sent on error)

Normally, this message never happens. This is useful to debug the system. The error messages are described below:

- Code 1: Starter activated with RPM not 0
- Code 2: RPM too high (the motor RPM was over 8000 RPM)
- Code 3: Speed is too high (the car speed was over 280 km/h)
- Code 4: The gear selection is wrong (gear in P or N, delta gear > 2)
- Code 5: Power motor activated without starter
- Code 6: On race mode, car hits a wall
- Code 7: The race is finished

4.1.11 BAD_MESSAGE

Message name	BAD_MESSAGE
CAN ID	`(0xB << 4) | CAR_ID`
Data[0]	Bad message indication code
Description	Indication of the car received a bad message.
Event Type	EVT (message is sent on bad message received)

Normally, this message never happens. This is useful to debug the system. The error messages are described below:

- Code 0: The length of the message is wrong
- Code 1: The value of a parameter is out of bounds
- Code 2: The message identifier is unknown

4.1.12 SLOPE_REQ

Message name	SLOPE_REQ
CAN ID	`(0xC << 4) | CAR_ID`
Data[0]	Car slope value (-100/front down to 100/front up)
Description	Indication of the slope position. The value 0 means the car is on a flat surface.
Event Type	RR (message is sent only after a remote request)

The usage of this information is just for tests. This information is not useful to the user.

4.1.13 RACE

Message name	**RACE**
CAN ID	`(0xD << 4) | CAR_ID`
Data[0]	Current race mode code
Description	Indicates on the race mode the state of the race.
Event Type	EVT (message is sent on change)

The usage of this information is useful to control the car in automatic race mode. The race mode codes are:

- Code 0: not in race mode (internal view)
- Code 1: In race mode, wait start
- Code 2: Race start !

4.1.14 CAR_ID

Message name	**CAR_ID**
CAN ID	`(0xF << 4) | 0x0F`
Data[0]	Motor identifier (on answer only)
Description	Indicates the motor revision model.
Event Type	RR (message is sent on request only)

At start, it is absolutely necessary to know the CAR_ID of the car motor because this information will be transmitted on all identifier messages (except for the CAR_ID message itself). The request has to be sent with remote request bit set.

4.2 Messages from external controller to car

4.2.1 LIGHT_FRONT

Message name	**LIGHT_FRONT**
CAN ID	`(0x11 << 4) | CAR_ID`
Data[0]	The value ‘0’ means no headlights, the value ‘50’ means headlights low beam and 100 means headlights high beam
Description	Sets the headlights lightning level

If contact key is off, the lights have to be off (at least after some seconds). If contact key is on, the headlights must be turned on. The usage of the high/low beams are depending on the student ideas

4.2.2 LIGHT_BACK

Message name	**LIGHT_BACK**
CAN ID	`(0x12 << 4) | CAR_ID`
Data[0]	The value ‘0’ means no backlights, the value ‘50’ means backlights low beam and 100 means brake lights
Description	Sets the backlights lightning level

If contact key is off, the lights must be off (at least after some seconds). If contact key is on, the headlights must be turned on. When the brake pedal is pressed or when the car automatically brakes, the brake lights have to be chosen.

4.2.3 TIME

Message name	**TIME**
CAN ID	`(0x13 << 4) | CAR_ID`
Data[0]	The current hours
Data[1]	The current minutes
Data[2]	The colon (between hours and minutes) 0-not visible, 1-visible
Description	Sets the time in cockpit

Use this message to set the current time in cockpit. The colon must be used to blink each second.

4.2.4 GEAR_LVL

Message name	**GEAR_LVL**
CAN ID	`(0x14 << 4) | CAR_ID`
Data[0]	The current gear level (0 to 5)
Description	Sets the current automatic gear level

Use this message change the current gear level. The value 0 means the wheels are not driven by the motor (neutral behavior). The gear level 0 to 5 must be used for the drive forward direction. The gear level 0 or 1 only must be used for the reverse direction. Caution, don’t change more than one gear each time.

4.2.5 AUDIO

Message name	AUDIO
CAN ID	`(0x15 << 4) | CAR_ID`
Data[0]	The motor audio level (0 to 100%)
Data[1]	The wheels drive (0: no drive, 1: drive wheels)
Description	Used for quiet mode in laboratories

Use this message to change the audio volume of the motor and to disable the wheels driving for quiet mode in laboratories.

4.2.6 PWR_MOTOR

Message name	PWR_MOTOR
CAN ID	`(0x16 << 4) | CAR_ID`
Data[0]	The motor power applied (0 to 100%)
Data[1]	Starter (0 or 1), set to 1 when RPM is 0, else set to 0
Description	To control the motor RPM

This is the control of the motor RPM.

The usage of “Start and Stop” must be made with the following respect:

When the brake pedal is pressed, the motor must stop. When the brake pedal is released, the motor must restart.

4.2.7 PWR_BRAKE

Message name	PWR_BRAKE
CAN ID	`(0x17 << 4) | CAR_ID`
Data[0]	The car brake factor applied (0 to 100%)
Description	To control the car wheels brake

This is the control of the car brake wheels. For example, it can hold the car stopped even in a slope.

4.2.8 TEMPO_OFF

Message name	TEMPO_OFF
CAN ID	`(0x18 << 4) | CAR_ID`
Description	To stop the tempomat mode

This message must be sent for example when the driver presses the brake pedal in tempomat mode.

4.2.9 KM_PULSE

Message name	KM_PULSE
CAN ID	`(0x19 << 4)	CAR_ID`
Description	Indicate that 100 meter have passed

This message must be sent each time the distance of 100 meters is reached.

4.2.10 AUTO_STEERING

Message name	AUTO_STEERING
CAN ID	`(0x1A << 4)	CAR_ID`
Data[0]	Steering wheel position (-100 to 100)
Data[1]	0: no automatic control, 1: automatic control
Description	Used to drive the car in race mode

This message offers the ability to drive the steering wheel of the car in race mode. The EXT_SENSORS message analyze permit to select this option.

4.2.11 CAR_RST

Message name	CAR_RST
CAN ID	`(0x1F << 4)	CAR_ID`
Description	Used to reset the car after a crash

Normally never use this message except for debug.

4.2.12 TOT_KM

Message name	TOT_KM
CAN ID	`(0x20 << 4)	CAR_ID`
Data[0]	Special control code 0xAA
Data[1]	Special control code 0x55
Data[2]-Data[4]	24 bits total kilometers (MSB-LSB)
Data[5]	CAR_ID (motor revision number (0 to 9))
Description	Used to change the motor revision number or total kilometers information

This message is only usable in factory Please dear students, don’t use it.

4.2.13 SETUP_ANALOG

Message name	SETUP_ANALOG
CAN ID	`(0x21 << 4)	CAR_ID`
Data[0]	Special control code 0xAA
Data[1]	Special control code 0x55
Data[2]	Code
Description	Used to setup the potentiometers limits

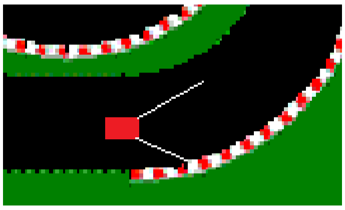
This message is only usable in factory Please dear students, don't use it.

To setup the analog sensors, the three next steps have to be respected.

1. Don't touch pedals, place steering wheel on the maximum position left, place the front of the car down, select Code '1' and send the message
2. Press both pedals, place steering wheel on the maximum position right, place the front of the car up, select Code '2' and send the message
3. Select the code '3' to store these parameters in car FLASH memory.

5 RACE MODE

The driving of the car in race mode can be controlled automatically. The message EXT_SENSORS returns the distance between the car and the obstacle. The picture below shows the car in red and the sensor measure in white.



Sensing system

In this example, the sensor front left returns the maximum value (63) because no obstacle is visible. The sensor front right returns a smaller value because an obstacle is visible. In this case, we can start to turn to the left.

If the car touches a border of the track, the car crash and a message indicating the time elapsed and the speed at the crash is displayed.

If the car arrives at the finish position, a message indicates the time elapsed for the race and the maximum speed reach.

6 DEMO MODE

The demonstration mode just shows the behavior of the car motor expected but with problems.

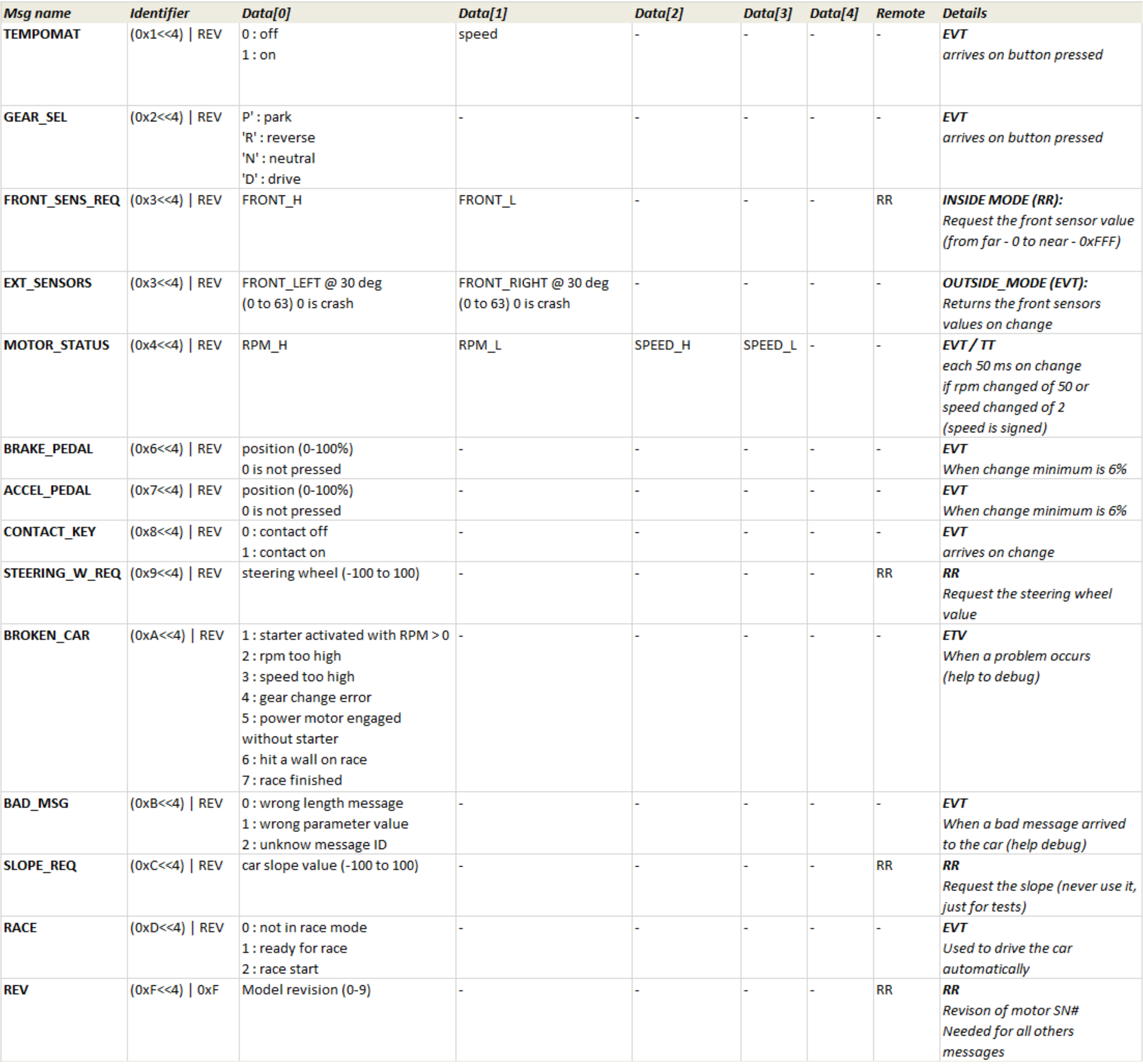
To enter in demo mode, the user must respect the state below when the startup screen is pressed.

- Press the accelerator and the brake pedals at the maximum.

8 SUMMARY OF CAN MESSAGES (FW 2.00)

The messages bitrate is 250kB. The 11 bits standard identifier is used.

8.1 From car to controller



Msg name	*Identifier*	*Data[0]*	*Data[1]*	*Data[2]*	*Data[3]*	*Data[4]*	*Remote*	*Details*
TEMPOMAT	(0x1<<4) | REV	0 : off 1 : on	speed	-	-	-	-	***EVT*** *arrives on button pressed*
GEAR_SEL	(0x2<<4) | REV	P' : park 'R' : reverse 'N' : neutral 'D' : drive	-	-	-	-	-	***EVT*** *arrives on button pressed*
FRONT_SENS_REQ	(0x3<<4) | REV	FRONT_H	FRONT_L	-	-	-	RR	***INSIDE MODE (RR):*** *Request the front sensor value (from far - 0 to near - 0xFFF)*
EXT_SENSORS	(0x3<<4) | REV	FRONT_LEFT @ 30 deg (0 to 63) 0 is crash	FRONT_RIGHT @ 30 deg (0 to 63) 0 is crash	-	-	-	-	***OUTSIDE_MODE (EVT):*** *Returns the front sensors values on change*
MOTOR_STATUS	(0x4<<4) | REV	RPM_H	RPM_L	SPEED_H	SPEED_L	-	-	***EVT / TT*** *each 50 ms on change if rpm changed of 50 or speed changed of 2* *(speed is signed)*
BRAKE_PEDAL	(0x6<<4) | REV	position (0-100%) 0 is not pressed	-	-	-	-	-	***EVT*** *When change minimum is 6%*
ACCEL_PEDAL	(0x7<<4) | REV	position (0-100%) 0 is not pressed	-	-	-	-	-	***EVT*** *When change minimum is 6%*
CONTACT_KEY	(0x8<<4) | REV	0 : contact off 1 : contact on	-	-	-	-	-	***EVT*** *arrives on change*
STEERING_W_REQ	(0x9<<4) | REV	steering wheel (-100 to 100)	-	-	-	-	RR	***RR*** *Request the steering wheel value*
BROKEN_CAR	(0xA<<4) | REV	1 : starter activated with RPM > 0 2 : rpm too high 3 : speed too high 4 : gear change error 5 : power motor engaged without starter 6 : hit a wall on race 7 : race finished	-	-	-	-	-	***ETV*** *When a problem occurs (help to debug)*
BAD_MSG	(0xB<<4) | REV	0 : wrong length message 1 : wrong parameter value 2 : unknow message ID	-	-	-	-	-	***EVT*** *When a bad message arrived to the car (help debug)*
SLOPE_REQ	(0xC<<4) | REV	car slope value (-100 to 100)	-	-	-	-	RR	***RR*** *Request the slope (never use it, just for tests)*
RACE	(0xD<<4) | REV	0 : not in race mode 1 : ready for race 2 : race start	-	-	-	-	-	***EVT*** *Used to drive the car automatically*
REV	(0xF<<4) | 0xF	Model revision (0-9)	-	-	-	-	RR	***RR*** *Revison of motor SN#* *Needed for all others messages*

8.2 From controller to car

Msg name	*Identifier*	*Data[0]*	*Data[1]*	*Data[2]*	*Data[3]*	*Data[4]*	*Remote*	*Details*
LIGHT_FRONT	(0x11<<4) | REV	0 - 100 (off to max)	-	-	-	-	-	*Off if contact key is off, else user free usage*
LIGHT_BACK	(0x12<<4) | REV	0 - 100 (off to max)	-	-	-	-	-	*Off if contact key is off, brake when braking or brake pedal pressed, free for other cases*
TIME	(0x13<<4) | REV	hours	minutes	colon 0 : no colon 1 : colon visible	-	-	-	*To set time in cockpit, colon may be used to blink at second period*
GEAR_LVL	(0x14<<4) | REV	0 : motor not driven 1 - 5 : gear level	-	-	-	-	-	*Gear 0 has same behavior as neutral* *For reverse, use only the gear 1*
AUDIO	(0x15<<4) | REV	Motor acoustic volume (0-100%)	Drive wheels 0 : no wheels 1 : wheels driven	-	-	-	-	*For debug in classroom, (less noisy)*
PWR_MOTOR	(0x16<<4) | REV	(0-100%)	Starter 0 : no starter 1 : starter active	-	-	-	-	*Caution, use starter with respect to norm; over 8000 rpm, motor explode* *if speed is over 280, car break*
PWR_BRAKE	(0x17<<4) | REV	(0-100%)	-	-	-	-	-	*Brake factor applied to car*
TEMPO_OFF	(0x18<<4) | REV	-	-	-	-	-	-	*Disable the tempomat selection (on brake for example)*
KM_PULSE	(0x19<<4) | REV	-	-	-	-	-	-	*One pulse each 100 meters*
AUTO_STEERING	(0x1A<<4) | REV	steering position (-100/LEFT to 100/RIGHT)	0 : no automatic control 1 : automatic control	-	-	-	-	*Offers the ability to control the steering wheel (race mode)*
CAR_RST	(0x1F<<4) | REV	-	-	-	-	-	-	*Reset the car (after a broken state, normally, never)*
TOT_KM	(0x20<<4) | REV	0xAA	0x55	KM_HH	KM_H	KM_L	REV	Sets the km total of car and the motor revision number**(students forbiden)**
SETUP_ANALOG	(0x21<<4) | REV	0xAA	0x55	1 : store pos L 2 : store pos R 3 : save on flash	-	-	-	1: steering left, pedals not pressed, car front down 2: steering right, pedal pressed, car front up **(students forbiden)**