	<b>oXigen system</b>	Doc.
	Descrizione Communication protocol	Pag. 1/ 13



This document outlines the communication protocol utilized by the oXigen USB dongle, specifically referring to oXigen 4, version 4.15 and later. The recommended dongle type is O204b, also known as the "blue dongle". Using the old O204a dongle ("yellow dongle"), while possible, is strongly discouraged except for migration purposes. Nota also that some features may not be available on such legacy device.

The primary objective of this communication protocol is to:

- transmit all data collected from the controllers of the cars to the PC via USB. This encompasses various information such as lap times, pit lane activity, battery levels, and more
- transfer dongle-specific data, race-wide parameters (e.g., race status, top speed), and car-specific details (e.g., individual car braking or fuel-based strategy limitations) from the PC's Race Management Software (RMS) to the dongle, subsequently distributing it to the controllers.

In an oXigen system, racing can be in one of four possible states, selected via RMS (Race Management System):

### **Started / Stopped / Paused / Flagged**

#### **Started:**

Race is normally on. Lap counting is enabled. Max speed is set through the 'max speed knob'

#### **Stopped:**

Race is over.. Pressing 'Start' clears all Lap Counting data

#### **Paused:**


Race is, well, paused. It can be restarted or stopped. If restarted, lap counting is not affected.

#### **Flagged:**

This is the 'safety car' condition. Speed may be limited and / or lane changing disabled, or not.

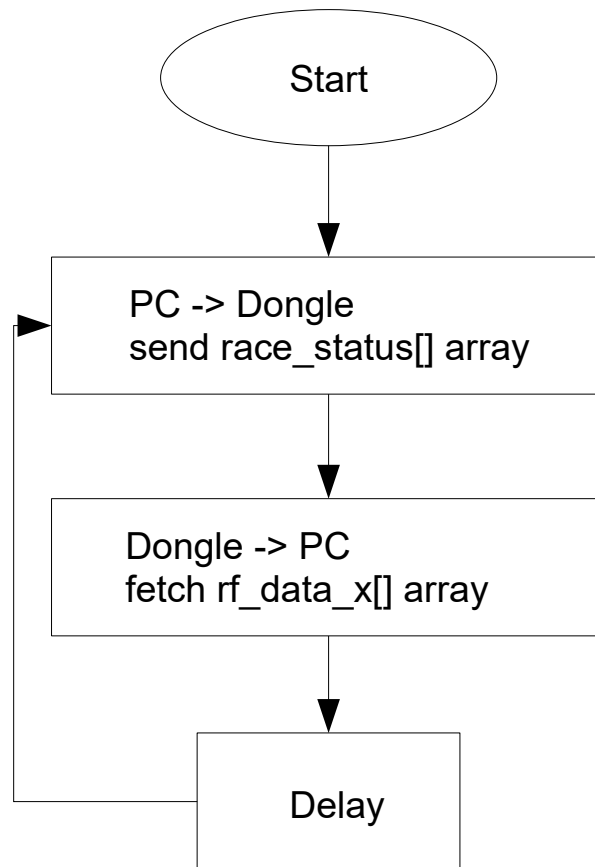
**NOTE:** The PC recognises the dongle device as a virtual COM port. The ID vendor is 0x1FEE and the ID product is 0x0002.

Esecutore C.Anceschi	Firma approvazione M.Ferrari	Rev. 4.06	Data 07/11/24
-------------------------	---------------------------------	--------------	------------------

	<b>oXigen system</b>	Doc.
	Descrizione <p style="text-align: center;">Communication protocol</p>	Pag. <p style="text-align: center;">2/ 13</p>

Communication between the PC and the dongle is bidirectional. The specific logic governing data exchange varies depending on the model of the dongle being utilized.

In the case of the O204a dongle, also called "yellow dongle", the transmission of collected data (for up to 3 cars per USB packet) occurs subsequent to the receipt of a transmitted packet from the PC. The following flowchart provides a visual representation of the described process. *Use of this dongle with oXigen 4.x series is strongly discouraged.*




The 'delay' value is intricately linked to the speed of the PC and the workload it is handling, which consequently impacts the performance of the RMS.

For the O204b dongle, also called "blue dongle", the transmission of collected data (normally one car per USB packet) occurs every 10ms. Unlike the yellow dongle, the blue dongle does not wait for a packet from the PC to initiate data transmission.

It's important to note that each controller transmits its data to the dongle at intervals of 300 ms (i.e. every 300 ms) in a round robin cycle.

Esecutore C.Anceschi	Firma approvazione M.Ferrari	Rev. 4.06	Data 07/11/24
-------------------------	---------------------------------	--------------	------------------

	<h1>oXigen system</h1>	Doc.
	Descrizione <b>Communication protocol</b>	Pag. 3/ 13

## Dongle - PC communication protocol:

As described above, the PC sends the race status to the dongle before reading the information collected from the cars. Hence, the protocol consists of two parts: TX and RX.

### TX (PC → Dongle):

TX protocol: payload 10 bytes.


*race\_status[0], [1]* : hold general information valid for all the controller/car pairs  
*race\_status[2], [5], [6]* : hold specific information only valid for one controller/car pair  
*race\_status[3], [4]* : hold specific information for all cars (global)

**NOTE:** The 00h value of this byte is reserved for system communications, currently not implemented.

byte	bit	value (hex)	meaning
0	0..4	1	race <b>stopped</b> . (set <i>race_status[1]</i> to 0x00)
		2	not used
		3	race <b>running</b> .
		4	race <b>paused</b> .
		5	race <b>flagged, LC enabled</b> .
		6	dongle command. Refer to 'Dongle Commands' paragraph
		15h	race <b>flagged, LC disabled</b> .
	5	0/1	pitlane lap trigger (valid only if bit 6 = 1): 0: lap counter is increased at the pitlane entry (lap time reference is the first magnet of the two ones used to define the pitlane entry) 1: lap counter is increased at the pitlane exit (lap time reference is the magnet used to define the pitlane exit)
			6
	7		not used (*)

(\*) When the chrono is started, the dongle receives the FFh command which means free race. The command is sent only when the chrono is started and in no other operating mode. The aim is to collect the number of controllers turns. The same can be achieved by starting the chrono in PAUSE mode.

Esecutore <b>C.Anceschi</b>	Firma approvazione <b>M.Ferrari</b>	Rev. 4.06	Data 07/11/24
--------------------------------	--	--------------	------------------

	<b>oXigen system</b>	Doc.
	Descrizione <b>Communication protocol</b>	Pag. <b>4/ 13</b>

byte	meaning	value (hex)	meaning
1	maximum speed	0-FFh	maximum global allowed speed (for all cars). This can be changed at will according to the status of the race. If the race is stopped or paused, this value is ignored by the controllers as the speed is forced to 0.

**Important:** to initiate data transmission, the dongle must receive a packet beginning with race\_status[0]=61h or race\_status[0]=0Fh and race\_status[1]=FFh. The values of the remaining bytes must adhere to the current protocol.

byte	meaning	value (hex)	meaning
2	controller ID for car-specific information tx	0-27h	set this byte to the ID of the car that specific information (e.g. speed limitation) must be sent to

byte	bit	value (hex)	meaning
3	0..6	0-7Fh	command value. Refer to 'Command description' section for an explanation
	7	0	0: global command. (for all controllers)

byte	meaning	value (hex)	meaning
4	command argument	0-FFh	command argument. Refer to 'Command description' section for an explanation


**Note:** each instruction for the controller/car is made of a 'command', (race\_status[3], bits 0..6, and a 'argument', race\_status[4])

byte	bit	value (hex)	meaning
5	0..6	0-7Fh	command value. Refer to 'Command description' section for an explanation
	7	0/1	1: car-specific command (car ID according to race_status[2])

byte	meaning	value (hex)	meaning
6	command argument	0-FFh	command argument. Refer to 'Command description' section for an explanation

**Note:** each instruction for the controller/car is made of a 'command', (race\_status[3], bits 0..6, and a 'argument', race\_status[4])

Esecutore <b>C.Anceschi</b>	Firma approvazione <b>M.Ferrari</b>	Rev. <b>4.06</b>	Data <b>07/11/24</b>
--------------------------------	--	---------------------	-------------------------

	<b>oXigen system</b>		Doc.
	Descrizione	Communication protocol	Pag. 5/ 13

byte	bit	value (hex)	meaning
7	0..6		reserved
	7	0/1	specification of the required power mean value: 0: controllers are requested to transmit the mean value of the trigger position 1: controllers are requested to transmit the average value of the PWM sent to the car.


The details of the global/individual instructions are described in the Device Commands section.

The following three bytes are designated for communicating the race timer value to the dongle. When a USB packet is prepared for transmission to the dongle by the PC, the current race timer value is appended to the packet.

Upon loading the USB packet for the PC, the dongle forwards the stored race timer value directly. This allows the PC to determine the time delay between sending and receiving the USB packet. By adding this time delay to the value stored in rf\_data\_x[4] byte of the buffer sent to the PC by the dongle, the race time value of the lap trigger can be calculated.

byte	meaning	value (hex)	meaning
8, 9, 10	race timer value	0-FFFFFFh	Value, in centiseconds, of the race timer saved at the moment the USB packet is sent to the dongle

Esecutore C.Anceschi	Firma approvazione M.Ferrari	Rev. 4.06	Data 07/11/24
-------------------------	---------------------------------	--------------	------------------

	<b>oXigen system</b>	Doc.
	Descrizione <b>Communication protocol</b>	Pag. 6/ 13

### **RX (Dongle → PC):**

RX protocol: for each controller-car pair, 13 bytes are received, denoted as rf\_data\_x[13]. Nine bytes capture the state of the controller-car pair, while four bytes, containing the value of the race timer saved when the dongle receives the packet containing new lap information from the controller, are appended to the packet.


In the case of the yellow dongle, with a maximum payload of 52 bytes (the first one is used to specify the packet size), data from up to 3 controller-car pairs can be sent following the reception of the packet by the PC. This means data is retrieved from each controller-car pair in a round-robin fashion, with a maximum of 3 pairs per USB payload.

As for the blue dongle, data is sent to the PC every 10ms, corresponding to the time when the dongle queries a controller. Therefore, the dongle typically sends data related to only one controller-car pair, but could potentially send data for up to a maximum of four controller-car pairs.

byte	bit	value (hex)	meaning
0	0	0/1	car reset event: 0: car power supply hasn't changed 1: car has just been powered up / reset (info available for 2 seconds)
	1	0/1	controller-car link check: 0: controller link with its paired car hasn't changed 1: controller has just got the link with its paired car (info available for 2 seconds) (e.g.:link dropped and restarted)
	2	0	not used
	3	0	not used
	4	0/1	car pit-lane status: 0: car is not in pit-lane; 1: car is in pit-lane
	5	0	not used
	6	0	not used
	7	1	always equal to 1

byte	bit	value (hex)	meaning
1	0..7	0..27h	controller-car pair ID in hexadecimal format (it certifies the controller is powered up)

Esecutore <b>C.Anceschi</b>	Firma approvazione <b>M.Ferrari</b>	Rev. <b>4.06</b>	Data <b>07/11/24</b>
--------------------------------	--	---------------------	-------------------------

	<b>oXigen system</b>		Doc.
	Descrizione	Communication protocol	

byte	bit	value (hex)	meaning
2	0..7	0..FFh	last lap time (high byte) in cs (hundredths of sec): see Note1

byte	bit	value (hex)	meaning
3	0..7	0..FFh	last lap time (low byte) in cs (hundredths of sec): see Note1

**Note 1:**

Get the last lap time in seconds with the following formula

$$\text{last lap time [s]} = [(\text{rf\_data\_x [2]} * 256) + (\text{rf\_data\_x [3]})] / 99,25$$

byte	bit	value (hex)	meaning
4	0..7	0..FFh	time delay, in centiseconds, between the moment the lap trigger is detected by the car and the moment the new lap info is received by the dongle

byte	bit	value (hex)	meaning
5	0..7	0..FFh	total lap number (low byte): see Note2


byte	bit	value (hex)	meaning
6	0..7	0..FFh	total lap number ( high byte): see Note2

**Note 2:**

Get the total lap number with the following formula

$$\text{lap number} = (\text{rf\_data\_x [6]} * 256) + (\text{rf\_data\_x [5]})$$

Esecutore C.Anceschi	Firma approvazione M.Ferrari	Rev. 4.06	Data 07/11/24
-------------------------	---------------------------------	--------------	------------------

	<b>oXigen system</b>	Doc.
	Descrizione <b>Communication protocol</b>	Pag. <b>8/ 13</b>

byte	bit	value (hex)	meaning
7	0..6	0..7Fh	power mean value: see Note3
	7	0/1	car fuel and feedback: 0: the car is not on the track 1: the car is on the track. Info available only if the paired controller is powered up

**Note 3:**

This data can be used to calculate the fuel consumption. Depending on the value of bit 7 in the byte race\_status[7] transmitted from the PC to the dongle, the power mean encapsulates different information. When bit 7 is equal to 1, the power mean represents the average value of the driving PWM (Pulse Width Modulation) sent from the controller to the car. Conversely, when bit 7 is equal to 0, it signifies the average value of the controller's trigger position.

Currently the Chrono RMS uses the following formula to scale it to 10:

$$\text{tmpvalue} = ((\text{rf\_data\_x}[6] \text{ AND } 7\text{Fh}) / 127 * 10)$$


byte	bit	value (hex)	meaning
8	0..4	0..1Fh	sub software release of the device specified by bit 7
	5, 6	0..3h	(main software - 4) release of the device specified by bit 7 The value in these of the two bits is to be added to 4 to obtain the main software. The new protocol applies to main release 4 or higher.
	7	0/1	device software release owner: 0: controller software release; 1: car software release

**Note 4:** if the SCP controller is reset, when the car crosses the finish line, the dongle sends the correct lap number to the PC along with a lap time value of zero. This is done to prevent incorrect lap time calculations following a controller reset, as lap times are computed by the SCP and can serve as an indicator of controller resets. This precaution is particularly crucial in scenarios where a controller reset occurs while the car is in the pit lane and is simultaneously removed from the track. Such conditions could potentially result in an extra lap being erroneously added upon the car's exit from the pit lane.

In the event of a controller reset, the total count of laps completed by the car remains intact. The dongle is designed to include the laps recorded by the controller after the reset into the existing total. The lap count is then transmitted to the RMS, which is responsible for deducting or adding additional laps as necessary for penalty applications or other requirements.


Esecutore <b>C.Anceschi</b>	Firma approvazione <b>M.Ferrari</b>	Rev. <b>4.06</b>	Data <b>07/11/24</b>
--------------------------------	--	---------------------	-------------------------



	<b>oXigen system</b>		Doc.
	Descrizione	Communication protocol	Pag. 9/ 13

byte	bit	value (hex)	meaning
9	0	0	not used
	1	0	not used
	2	0/1	controller battery level warning: 0: controller battery level is OK 1: controller battery level is low
	3	0/1	track call check: 0: no track call from controller 1: track call from controller (info available for 2 seconds). Round button on controller (ID equal to <i>rf_data_x [1]</i> ) was pressed for more than 0,5 seconds. This information can be used to trigger a track call, if, for example, the relative car is off track AND this condition occurs. Race can then be paused; and restarted manually or automatically after a certain time delay
	4	0/1	lap time information: 0: lap time does not include any 'short laps' 1: lap time includes 'short laps.' This means the car has recorded a lap time shorter than the minimum time set in the chrono. The controller waits until it receives a total of n partial laps whose combined times exceed the minimum time value before transmitting the lap time to the dongle.
	5	0/1	arrow up button status: 0: button not pressed 1: button pressed
	6	0/1	arrow down button status: 0: button not pressed 1: button pressed
	7	0/1	round button status: 0: button not pressed 1: button pressed

Esecutore <b>C.Anceschi</b>	Firma approvazione <b>M.Ferrari</b>	Rev. <b>4.06</b>	Data <b>07/11/24</b>
--------------------------------	--	---------------------	-------------------------

	<b><i>oXigen system</i></b>	Doc.
	Descrizione <b>Communication protocol</b>	Pag. <b>10/ 13</b>

byte	meaning	value (hex)	meaning
10..12	race timer value	0-FFFFFFh	Value, in centiseconds, of the race timer saved at the moment the dongle receives the packet containing a new lap info from the controller.


**Note 5:** The dongle counts the race timer, in centiseconds. It is cleared when the dongle receives the START command from the PC. Value, in centiseconds, of the race timer saved at the moment the dongle receives the packet containing a new lap info from the controller is concatenated, splitted in three bytes, to the packet.

Get the race timer value, in centiseconds, corresponding to the last lap recorded with the following formula

$$\text{race timer (last lap)} = ((\text{rf\_data\_x}[10] * 65536) + (\text{rf\_data\_x}[11] * 256) + (\text{rf\_data\_x}[12]) - (\text{rf\_data\_x}[4]))$$

This value is used to manage the race table in case of multiples cars with the same lap counter value.

Esecutore <b>C.Anceschi</b>	Firma approvazione <b>M.Ferrari</b>	Rev. <b>4.06</b>	Data <b>07/11/24</b>
--------------------------------	--	---------------------	-------------------------

	<b>oXigen system</b>	Doc.
	Descrizione <b>Communication protocol</b>	Pag. <b>11/ 13</b>

## Device commands

Two command types are currently available:

- commands for one or all controllers;
- commands for the dongle.

## Controller(s) commands


Command description	race_status[3] [0..6]	race_status[4]															
No Action	0000000 b	00h															
Set Pit Lane Speed	0000001 b	New pit-lane speed value															
Set Maximum Speed	0000010 b	Maximum speed value															
Set Minimum Speed	0000011 b	bit 7: 1/0 force exchange such as arrow ↑ SCP button bit 6: 1/0 force exchange such as arrow ↓ SCP button bit 5..0: minimum speed limitation divided by 2															
Set RF Tx Power Level	0000100 b	Transmission power value															
		<table border="1"> <thead> <tr> <th>Value</th> <th>SCP2</th> <th>SCP3</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>-18dBm</td> <td>-8dBm</td> </tr> <tr> <td>1</td> <td>-12dBm</td> <td>-4dBm</td> </tr> <tr> <td>2</td> <td>-6dBm</td> <td>0dBm</td> </tr> <tr> <td>3</td> <td>0dbm</td> <td>+4dBm</td> </tr> </tbody> </table>	Value	SCP2	SCP3	0	-18dBm	-8dBm	1	-12dBm	-4dBm	2	-6dBm	0dBm	3	0dbm	+4dBm
		Value	SCP2	SCP3													
		0	-18dBm	-8dBm													
		1	-12dBm	-4dBm													
2	-6dBm	0dBm															
3	0dbm	+4dBm															
Set Maximum Brake	0000101 b	Maximum brake value															
Set Minimum Lap Time	0000111 b	Minimum acceptable lap time in 1/10" (ex. 105=10.5")															

In general, race information has priority over controller/car specific information. This is true for the commands 'Set RF Tx Power Level' and 'Set Minimum Speed', used to implement the Ghost mode.

The speeds (maximum and in pit lane) and brake values, are limited by the lowest of the two informations, global and individual: for example, if the global maximum speed is 90%, and the maximum speed for a specific car is 80%, that car will set its limit to 80%. If however the global top speed is turned down to 70%, then the car will set its speed limit to 70%. This principle applies to pit lane speed and brake value.

The MSB of race\_status[3] and race\_status[5] discriminate between a *global* command (all controllers) (set MSB to '0' in race\_status[3]) or a *specific* controller command (set MSB to '1' in race\_status[5]). Hence, the value in brackets in column 2 describe the command for the specific controller. Refer to TX (PC→ Dongle) section, byte 2, ..., 6 explanation

Esecutore <b>C.Anceschi</b>	Firma approvazione <b>M.Ferrari</b>	Rev. <b>4.06</b>	Data <b>07/11/24</b>
--------------------------------	--	---------------------	-------------------------

	<b>oXigen system</b>	Doc.
	Descrizione <p style="text-align: center;">Communication protocol</p>	Pag. <p style="text-align: center;">12/ 13</p>

#### Examples:

- Set Maximum Speed to 7Fh, for controller-car pair with ID 3:
  - race\_status[2] = 03h; → controller/car number
  - race\_status[5] = 82h; → command type
  - race\_status[6] = 7Fh; → upper speed limit value
- Set Minimum Speed to 40h, for all controller-car pairs:
  - race\_status[2] = *not important in this case type*;
  - race\_status[3] = 03h; → command type
  - race\_status[4] = 20h; → minimum speed value. The final value will be 40h.

To cancel a previously sent command, resend the command with a different value.

#### Examples:

- Enable top speed 100% for controller-car pair with ID 3:
  - race\_status[2] = 03h; → controller/car number
  - race\_status[5] = 82h; → command type
  - race\_status[6] = FFh; → upper speed limit value
- Remove minimum speed limitation for all controller-car pairs:
  - race\_status[2] = *not important in this case type*;
  - race\_status[3] = 03h; → command type
  - race\_status[4] = 00h; → minimum speed default value (speed 0 and no lane exchange forced).

### ***Dongle Commands***

Currently, we have developed only the dongle software release query, with the following command:

```

race_status[0]: 06h;
race_status[1]: 06h;
race_status[2]: 06h;
race_status[3]: 06h;
race_status[4]: XXh (any value);
race_status[5]: XXh (any value);
race_status[6]: XXh (any value);


```

The dongle will reply with five bytes: main sw release in first one, sub sw release in second one. The others three are 'zero' value bytes.

### ***Communication Reset***

Using the Slot.it chrono with the O204a dongle, the interruption of the radio communication between the dongle and the controllers was sometime detected. This is evidenced by the turning off of the green light that distinguishes the active connection of the controllers.

Esecutore C.Anceschi	Firma approvazione M.Ferrari	Rev. 4.06	Data 07/11/24
-------------------------	---------------------------------	--------------	------------------

	<b><i>oXigen system</i></b>	Doc.
	Descrizione <p style="text-align: center;">Communication protocol</p>	Pag. <p style="text-align: center;">13/ 13</p>

It is possible to try to restore the radio communication by sending the string 'SPISTART' to the dongle via USB. The dongle restarts the radio communication without clearing the lap count. This is valid from firmware version 3.13. This applies to the O204a "yellow" dongle.

```

race_status[0]: 'S';
race_status[1]: 'P';
race_status[2]: 'I';
race_status[3]: 'S';
race_status[4]: 'T';
race_status[5]: 'A';
race_status[6]: 'R';
race_status[7]: 'T';

```

Esecutore <b>C.Anceschi</b>	Firma approvazione <b>M.Ferrari</b>	Rev. <b>4.06</b>	Data <b>07/11/24</b>
--------------------------------	--	---------------------	-------------------------