Sign In or Register

CODEMASTERS COMMUNITY

BROWSE ACTIVITY

Forums Clubs Calendar Staff Online Users Leaderboard Search...

Home > F1 Games > F1® 2019 Game Forum > Technical Assistance > All Activity

NVIDIA Graphics Card? Crashing? Click here for solution - UPDATED 27/04

Want to report an issue? Have a bug to report? Please look here first!!!!

Archived

This topic is now archived and is closed to further replies.

F1 2017 D-Box and UDP Output Specification

By Hoo, June 12, 2017 in Technical Assistance

★★★★★

1 2 3 4 5 6 NEXT » Page 1 of 11

Hoo

Codemasters Staff

Codemasters

163

1,199 posts

Posted June 12, 2017

Overview

The F1 series of games support output of game data to external devices such as D-BOX, other motion platforms, steering wheels and LED devices that are connected to your PC or console. The purpose of this document is to summarise the data that is outputted so that developers of supporting hardware or software are able to configure these to work with the F1 game correctly.

This thread provides an updated spec for the UDP output system in F1 2017. The previous thread for F1 2016 was located here: http://forums.codemasters.com/discussion/46726/d-box-and-udp-telemetry-information. Please use this thread to post any bugs with the F1 2017 UDP system, or to suggest further refinements you'd like to see in future.

D-BOX Output

D-BOX output is currently supported on the PC platform. In F1 2017, the D-BOX activation is controlled via the menus instead of the hardware settings config file. Navigate to Game Options->Settings->UDP Telemetry Settings->D-BOX to activate this on your system.

Enabling the UDP Telemetry Output

In F1 2017, UDP telemetry output is controlled via the menus on all platforms. To enable this, enter the options menu from the main menu (triangle / Y), then enter the settings menu - the UDP option will be at the bottom of the list. From there you will be able to enable / disable the UDP output, configure the IP address and port for the receiving application, toggle broadcast mode and set the send rate. Broadcast mode transmits the data across the network subnet to allow multiple devices on the same subnet to be able to receive this information. When using broadcast mode it is not necessary to set a target IP address, just a target port for applications to listen on.

Advanced PC Users

You can additionally edit the game’s configuration XML file to configure UDP output. The file is located here (after an initial boot of the game):

...\Documents\My Games\<game_folder>\hardwaresettings\hardware_settings_config.xml

You should see the tag:

```
<motion>

  ...

  <udp enabled="false" broadcast=”false” ip="127.0.0.1" port=

          ...

</motion>
```

Here you can set the values manually. Note that any changes made within the game when it is running will overwrite any changes made manually.

New This Year

- Fixed bug where UDP information wasn't saved after a restart
- Added UDP option into in-game pause menu so settings can be configured and tested straight away
- Updated PC version to use in-game UI menus
- Added additional vehicle and session information into the UDP output

Hoo

Codemasters Staff

Codemasters

163
1,199 posts

Posted June 12, 2017

UDP Packet Structure

The data is sent as raw data in the UDP packet, converted to a char array, with packing *enabled* (no padding to align different sized types). To decode this into something usable it should be a case of casting the packet data back to the UDPPacket struct (or another structure with the same layout). The layout of the UDP data is as follows:

```
// Packet size – 1289 bytes

struct UDPPacket

{

    float m_time;

    float m_lapTime;

    float m_lapDistance;

    float m_totalDistance;

    float m_x; // World space position

    float m_y; // World space position

    float m_z; // World space position

    float m_speed; // Speed of car in MPH
float m_xv; // Velocity in world space

float m_yv; // Velocity in world space

float m_zv; // Velocity in world space

float m_xr; // World space right direction

float m_yr; // World space right direction

float m_zr; // World space right direction

float m_xd; // World space forward direction

float m_yd; // World space forward direction

float m_zd; // World space forward direction

float m_susp_pos[4]; // Note: All wheel arrays have the order:

float m_susp_vel[4]; // RL, RR, FL, FR

float m_wheel_speed[4];

float m_throttle;

float m_steer;

float m_brake;

float m_clutch;

float m_gear;

float m_gforce_lat;

float m_gforce_lon;

float m_lap;

float m_engineRate;

float m_sli_pro_native_support; // SLI Pro support

float m_car_position; // car race position

float m_kers_level; // kers energy left

float m_kers_max_level; // kers maximum energy

float m_drs; // 0 = off, 1 = on

float m_traction_control; // 0 (off) - 2 (high)

float m_anti_lock_brakes; // 0 (off) - 1 (on)

float m_fuel_in_tank; // current fuel mass
```

```
    float m_fuel_capacity; // fuel capacity

    float m_in_pits; // 0 = none, 1 = pitting, 2 = in pit area

    float m_sector; // 0 = sector1, 1 = sector2, 2 = sector3

    float m_sector1_time; // time of sector1 (or 0)

    float m_sector2_time; // time of sector2 (or 0)

    float m_brakes_temp[4]; // brakes temperature (centigrade)

    float m_tyres_pressure[4]; // tyres pressure PSI

    float m_team_info; // team ID

    float m_total_laps; // total number of laps in this race

    float m_track_size; // track size meters

    float m_last_lap_time; // last lap time

    float m_max_rpm; // cars max RPM, at which point the rev limiter will kick in

    float m_idle_rpm; // cars idle RPM

    float m_max_gears; // maximum number of gears

    float m_sessionType; // 0 = unknown, 1 = practice, 2 = qualifying, 3 = race

    float m_drsAllowed; // 0 = not allowed, 1 = allowed, -1 = invalid / unknown

    float m_track_number; // -1 for unknown, 0-21 for tracks

    float m_vehicleFIAFlags; // -1 = invalid/unknown, 0 = none, 1 = green, 2 = blue, 3 = yellow, 4 = red

    float m_era;                    // era, 2017 (modern) or 1980 (classic)

    float m_engine_temperature;   // engine temperature (centigrade)

    float m_gforce_vert; // vertical g-force component

    float m_ang_vel_x; // angular velocity x-component

    float m_ang_vel_y; // angular velocity y-component

    float m_ang_vel_z; // angular velocity z-component

    byte  m_tyres_temperature[4]; // tyres temperature (centigrade)

    byte  m_tyres_wear[4]; // tyre wear percentage
    byte  m_tyre_compound; // compound of tyre – 0 = ultra soft, 1 = super soft, 2 = soft, 3 = medium, 4 = hard, 5 = inter, 6 = wet

    byte  m_front_brake_bias;          // front brake bias (percentage)

    byte  m_fuel_mix;                  // fuel mix - 0 = lean, 1 = standard, 2 = rich, 3 = max

    byte  m_currentLapInvalid;      // current lap invalid - 0 = valid, 1 = invalid

    byte  m_tyres_damage[4]; // tyre damage (percentage)

    byte  m_front_left_wing_damage; // front left wing damage (percentage)

    byte  m_front_right_wing_damage; // front right wing damage (percentage)

    byte  m_rear_wing_damage; // rear wing damage (percentage)

    byte  m_engine_damage; // engine damage (percentage)

    byte  m_gear_box_damage; // gear box damage (percentage)

    byte  m_exhaust_damage; // exhaust damage (percentage)

    byte  m_pit_limiter_status; // pit limiter status – 0 = off, 1 = on

    byte  m_pit_speed_limit; // pit speed limit in mph

    float m_session_time_left;  // NEW: time left in session in seconds

    byte  m_rev_lights_percent;  // NEW: rev lights indicator (percentage)

    byte  m_is_spectating;  // NEW: whether the player is spectating

    byte  m_spectator_car_index;  // NEW: index of the car being spectated



    // Car data

    byte  m_num_cars;                // number of cars in data

    byte  m_player_car_index;          // index of player's car in the array

    CarUDPData  m_car_data[20];    // data for all cars on track



    float m_yaw;  // NEW (v1.8)
    float m_pitch;  // NEW (v1.8)

    float m_roll;  // NEW (v1.8)

    float m_x_local_velocity;          // NEW (v1.8) Velocity in local space

    float m_y_local_velocity;          // NEW (v1.8) Velocity in local space

    float m_z_local_velocity;          // NEW (v1.8) Velocity in local space

    float m_susp_acceleration[4];   // NEW (v1.8) RL, RR, FL, FR

    float m_ang_acc_x;                 // NEW (v1.8) angular acceleration x-component

    float m_ang_acc_y;                 // NEW (v1.8) angular acceleration y-component

    float m_ang_acc_z;                 // NEW (v1.8) angular acceleration z-component



};



struct CarUDPData

{

    float m_worldPosition[3]; // world co-ordinates of vehicle

    float m_lastLapTime;

    float m_currentLapTime;

    float m_bestLapTime;

    float m_sector1Time;

    float m_sector2Time;

    float m_lapDistance;

    byte  m_driverId;

    byte  m_teamId;

    byte  m_carPosition;     // UPDATED: track positions of vehicle

    byte  m_currentLapNum;
    byte  m_tyreCompound; // compound of tyre – 0 = ultra soft, 1 = super soft, 2 = soft, 3 = medium, 4 = hard, 5 = inter, 6 = wet

    byte  m_inPits;           // 0 = none, 1 = pitting, 2 = in pit area

    byte  m_sector;           // 0 = sector1, 1 = sector2, 2 = sector3

    byte  m_currentLapInvalid; // current lap invalid - 0 = valid, 1 = invalid

    byte  m_penalties;  // NEW: accumulated time penalties in seconds to be added

};
```

Changes to data since F1 2016

- Brakes temperature value is now in Centigrade, rather than Kelvin
- Wheel data has all been converted to 4 value arrays, with order Rear Left, Rear Right, Front Left & Front Right. The actual data has not changed
- Fixed car race position

Hoo

Codemasters Staff

Codemasters

163

1,199 posts

Posted June 12, 2017

Track and Team IDs

Track ID	Track
0	Melbourne
1	Sepang
2	Shanghai
3	Sakhir (Bahrain)

4	Catalunya
5	Monaco
6	Montreal
7	Silverstone
8	Hockenheim
9	Hungaroring
10	Spa
11	Monza
12	Singapore
13	Suzuka
14	Abu Dhabi
15	Texas
16	Brazil
17	Austria
18	Sochi
19	Mexico
20	Baku (Azerbaijan)
21	Sakhir Short
22	Silverstone Short
23	Texas Short

24	Suzuka Short

Team	Team ID
Mercedes	4
Redbull	0
Ferrari	1
Force India	6
Williams	7
McLaren	2
Toro Rosso	8
Haas	11
Renault	3
Sauber	5

Classic Team	Team ID
Williams 1992	0
McLaren 1988	1
McLaren 2008	2
Ferrari 2004	3
Ferrari 1995	4
Ferrari 2007	5

McLaren 1998	6
Williams 1996	7
Renault 2006	8
Ferrari 2002	10
Redbull 2010	11
McLaren 1991	12

Driver	ID
Lewis Hamilton	9
Valtteri Bottas	15
Daniel Ricciardo	16
Max Verstappen	22
Sebastian Vettel	0
Kimi Räikkönen	6
Sergio Perez	5
Esteban Ocon	33
Felipe Massa	3
Lance Stroll	35
Fernando Alonso	2
Stoffel Vandoorne	34
Carlos Sainz Jr.	23

Daniil Kvyat	1
Romain Grosjean	7
Kevin Magnussen	14
Nico Hulkenberg	10
Jolyon Palmer	20
Marcus Ericsson	18
Pascal Wehrlein	31

Classic Driver	ID
Arron Barnes	23
Martin Giles	1
Alex Murray	16
Lucas Roth	68
Igor Correia	2
Sophie Levasseur	3
Jonas Schiffer	24
Alain Forest	4
Jay Letourneau	20
Esto Saari	6
Yasar Atiyeh	9
Callisto Calabresi	18

Naota Izum	22
Howard Clarke	10
Lars Kaufmann	8
Marie Laursen	14
Flavio Nieves	31
Peter Belousov	7
Klimek Michalski	0
Santiago Moreno	5
Benjamin Coppens	15
Noah Visser	32
Gert Waldmuller	33
Julian Quesada	34

steviejay69

Members
983
4,814 posts

Posted June 12, 2017

I don't understand the driver ID logic. What logic? Exactly.

bax

Posted June 13, 2017

> Hoo said:
>
> // Car data

B

Members
● 0
65 posts

```
    byte  m_num_cars;              // number of cars in data

    byte  m_player_car_index;        // index of player's car in
the array

    CarUDPData  m_car_data[20];   // data for all cars on track
```

GREAT !!!!! now we can have a custom HUD for live coverage !!!!

bax

B

Members
● 0
65 posts

Posted June 14, 2017

I'm not a F1 2017 tester, sadly, so I'm asking: are these new UDP features already active in F1 2017 demo?
Can I ask to my friends (testers) to try my app?

Hoo

Codemasters Staff

Codemasters

⊕ 163
1,199 posts

Posted June 14, 2017

These features should be in the current F1 2017 beta, so your friends should be able to try this out for you.

cjorgens79

C

Members
⊕ 3
168 posts

Posted June 16, 2017

Hi Hoo,
I have managed to spend some time implementing and testing the new interface. Overall its a great improvement over what was available in F1 2016, so i think other integrators will also be quite happy with what they now have access to.

That being said, i do have some comments/issues/suggestions to note. In no particular order..

First up is the structure packing. In your UDP Packet Structure post you said "The data is sent as raw data in the UDP packet, converted to a char array, with packing disabled.", however this is incorrect as the data being sent is actually packed. I looked back on my earlier post back in the F1 2016 forum and noticed that i had referred to structure packing, when i intended to say structure padding, im not sure if this has had any impact on the packing decision. I guess it can be done either way, however NOT packing seems to be the preferred method as it removes any potential endianess issues associated with decoding packed data at the other end at the expense of potentially larger data. In this case with packing disabled there would only be two bytes of padding inserted at position 330 (which is right before the start of the participants array, as it is currently unaligned there). A decision will need to be made one way or the other on this, all i can say is my preference would be packing disabled which is also inline with what project cars does with its udp telemetry. Otherwise you could manually pad it by adding two reserved bytes right before the participants array so that everything is perfectly aligned, however you would still need to remember this in future if more changes are made.

We could do with car enums for all the historic cars, as currently when using a historic car there is no way to know in the telemetry what car it is. The existing team enum could perhaps just be extended (with a higher range set aside, eg 50+) for custom team/car identifiers that we can map to the correct car names.

Could we have the players login name included in the telemetry? I see "cjorgens79" showing up in game, it would be very useful to have this in the telemetry for displaying the player's name against other participants. I also intend to link the telemetry to my new cloud service which keeps a record of session/lap details from every lap driven in any game that RS Dash links to. Having the player name as seen in game will make it easier to do friend/opponent lookups in the cloud service for users. It not the end of the world if you cant, i can force the user to enter it online themselves to deal with it if its not practical.

While we get values in the telemetry for the following fields, their values never change.
- Tyre Temp
- Tyre Pressure
- Brake Temp
- Fuel (i suspect but havent yet tested that this one changes based on starting fuel load, but it certainly doesn't decrease while driving)
- Engine Temp

- Damage (I assume eventually this will be able to be turned on in the settings?)

We currently have telemetry indicators for ABS and TC, could be good to also add a brakingAssist indicator to the output which will help people keep track of aids being used. Maybe this could take up on of the "padding" bytes, or it could just share bits with one of the other TC/ABS settings i guess.

When traction control is off in the menu, the telemetry outputs it as 0.44999998807907. Im guessing that the game internally always has some level of TC so the cars are actually playable. Not sure if you want to do anything about this, or maybe just make a note of it. A value of 0.5 is TC medium and i think 1 was TC full. So you could say anything < 0.5 is considered off in the notes.

How often is the participant info going to be updated. Will it be updated as fast as the player data? If so then you could potentially consider removing the duplication of player fields from the main body where the players timing related info (and a couple of other things) is in the participants struct as well as in the main body of the struct. Useful if you need to free up more space in the structure so something else can be added, otherwise just a cleanliness thing if anything.

FYI the comments on the m_speed and m_pit_speed_limit fields in the structure are wrong, both indicate the value is in MPH but the value is actually in m/s (meters per second).

Am i correct in that the game can only currently be played in Time Trial mode? I wanted to test out the participant data, but cannot find anyway to start a game with other players in it.

One last question, is there a way to see telemetry in game yet? From memory on F1 2016 you could use the MFD to show brake temps, etc, will there be something like the available in F1 2017 too? i want to use it to validate the telemetry data for those things matches what the game is displaying.

Anyhow, great job so far :)

Hoo

Posted June 16, 2017

Thanks for all of the feedback. I'll pass this on to the dev team to look at.

Regarding the temperatures and participant data, this isn't available in the Time Trial beta (wear and temperature is locked to optimal values). This will hopefully be available soon.

Codemasters Staff

Codemasters

⊕ 163
1,199 posts

Hoo

Posted June 16, 2017

I've just the traction control values with the handling team and they said this value is correct for the 2002 Ferrari as it has some in-built traction control. If you are seeing this with the modern cars or the 88 McLaren then there is probably a bug in there.

We'll get the IDs for the some classic teams and drivers released soon.

Codemasters Staff

Codemasters

⊕ 163
1,199 posts

cjorgens79

Posted June 16, 2017

C

Members
⊕ 3
168 posts

> Hoo said:
>
> I've just the traction control values with the handling team and they said this value is correct for the 2002 Ferrari as it has some in-built traction control. If you are seeing this with the modern cars or the 88 McLaren then there is probably a bug in there.
>
> We'll get the IDs for the some classic teams and drivers released soon.

Hi Hoo,

I definately wasnt using the 2002 Ferrari, but its possible i was using the 1992 williams, otherwise i was definately in a modern car. Do you need me to check for sure or are they able to check it out relatively easily?

Hoo

Codemasters Staff

Codemasters

163

1,199 posts

Posted June 16, 2017

If you are able to check this when you get a moment that would be much appreciated.

cjorgens79

C

Members

3

168 posts

Posted June 16, 2017

> Hoo said:
>
> If you are able to check this when you get a moment that would be much appreciated.

The 1992 williams definately has it, TC is 0.449999998807907. The 2002 ferrari also has it, its TC value is 0.5. The mclaren does not, and the modern cars do not (except while under AI control)

Hoo

Codemasters Staff

Codemasters

163

1,199 posts

Posted June 16, 2017

Thanks. That sounds like it's working as expected.

Kafumanto

K

Members

Posted June 19, 2017

Hi, "m_steer" seems to be a percentage. I would have expected it in degrees/radians. There is some convention on how to convert it to an angle?

3
66 posts

Hoo

Codemasters Staff

Codemasters

163
1,199 posts

Posted June 19, 2017

@Kafumanto - this is a percentage of the maximum steering input. This is the raw input like the throttle and brake values. Converting it to an angle would need the final steering result outputted from the game, which is different to how we treat the other values.

Kafumanto

K

Members
3
66 posts

Posted June 19, 2017

> Hoo said:
>
> @Kafumanto - this is a percentage of the maximum steering input. This is the raw input like the throttle and brake values. Converting it to an angle would need the final steering result outputted from the game, which is different to how we treat the other values.

@Hoo - thanks for the explanation!

mantazzo

Members
87
570 posts

Posted June 20, 2017

float m_vehicleFIAFlags; // -1 = invalid/unknown, 0 = none, 1 = green, 2 = blue, 3 = yellow, 4 = red

Red Flags back in game confirmed? (unless it was same in 2016)

And also one more thing...

byte m_fuel_mix;   // fuel mix - 0 = lean, 1 = standard, 2 = rich, 3 = max
New fuel mode, or just Time Trial mode? Just wondering.

bax

Members
● 0
65 posts

Posted June 20, 2017

Some question about struct CarUDPData, since my field of interest is the live coverage overlay

1) what is exactly this parameter?
byte m_trackPosition;   // track positions of vehicle
is it similar to *float m_car_position; // car race position* ?
If not... I can calculate the race car position using m_lapDistance & m_currentLapNum but ... but wow :#

2) calculating the delay between 2 drivers is not a problem using timers but there is one important info missing to determine the FINAL race car position: PENALTIES !! Time penalties greatly affect the final classification of the race in case of strict rules and it is typical to give this information to spectators during a live coverage. I don't think you will change now the total amount of bytes for each car but maybe you can send via UDP:

float m_car_penalty; // (total amount of penalties not already paid during pit stops) (s)
instead of
float m_bestLapTime // (s)

since best lap time is (not so important and) easily deductible from last lap time if following the race from the beginning.

MachSpeed

Members

Posted June 21, 2017

> Hoo said:
>
> UDP Packet Structure

● 0
223 posts

The data is sent as raw data in the UDP packet, converted to a char array, with packing disabled. To decode this into something usable it should be a case of casting the packet data back to the UDPPacket struct (or another structure with the same layout). The layout of the UDP data is as follows:

```
// Packet size – 1210 bytes

struct UDPPacket

{

    float m_time;

    float m_lapTime;

    float m_lapDistance;

    float m_totalDistance;

    float m_x; // World space position

    float m_y; // World space position

    float m_z; // World space position

    float m_speed; // Speed of car in MPH

    float m_xv; // Velocity in world space

    float m_yv; // Velocity in world space

    float m_zv; // Velocity in world space

    float m_xr; // World space right direction

    float m_yr; // World space right direction

    float m_zr; // World space right direction

    float m_xd; // World space forward direction

    float m_yd; // World space forward direction

    float m_zd; // World space forward direction

    float m_susp_pos[4]; // Note: All wheel arrays have the order:

    float m_susp_vel[4]; // RL, RR, FL, FR

    float m_wheel_speed[4];

    float m_throttle;
    float m_steer;
    float m_brake;
    float m_clutch;
    float m_gear;
    float m_gforce_lat;
    float m_gforce_lon;
    float m_lap;
    float m_engineRate;
    float m_sli_pro_native_support; // SLI Pro support
    float m_car_position; // car race position
    float m_kers_level; // kers energy left
    float m_kers_max_level; // kers maximum energy
    float m_drs; // 0 = off, 1 = on
    float m_traction_control; // 0 (off) - 2 (high)
    float m_anti_lock_brakes; // 0 (off) - 1 (on)
    float m_fuel_in_tank; // current fuel mass
    float m_fuel_capacity; // fuel capacity
    float m_in_pits; // 0 = none, 1 = pitting, 2 = in pit area
    float m_sector; // 0 = sector1, 1 = sector2, 2 = sector3
    float m_sector1_time; // time of sector1 (or 0)
    float m_sector2_time; // time of sector2 (or 0)
    float m_brakes_temp[4]; // brakes temperature (centigrade)
    float m_tyres_pressure[4]; // tyres pressure PSI
    float m_team_info; // team ID
    float m_total_laps; // total number of laps in this race
    float m_track_size; // track size meters
    float m_last_lap_time; // last lap time
    float m_max_rpm; // cars max RPM, at which point the rev limiter will kick in
    float m_idle_rpm; // cars idle RPM
    float m_max_gears; // maximum number of gears

    float m_sessionType; // 0 = unknown, 1 = practice, 2 = qualifying, 3 = race

    float m_drsAllowed; // 0 = not allowed, 1 = allowed, -1 = invalid / unknown

    float m_track_number; // -1 for unknown, 0-21 for tracks

    float m_vehicleFIAFlags; // -1 = invalid/unknown, 0 = none, 1 = green, 2 = blue, 3 = yellow, 4 = red

    float m_era;                     // era, 2017 (modern) or 1980 (classic)

    float m_engine_temperature;   // engine temperature (centigrade)

    float m_gforce_vert; // vertical g-force component

    float m_ang_vel_x; // angular velocity x-component

    float m_ang_vel_y; // angular velocity y-component

    float m_ang_vel_z; // angular velocity z-component

    byte  m_tyres_temperature[4]; // tyres temperature (centigrade)

    byte  m_tyres_wear[4]; // tyre wear percentage

    byte  m_tyre_compound; // compound of tyre – 0 = ultra soft, 1 = super soft, 2 = soft, 3 = medium, 4 = hard, 5 = inter, 6 = wet

    byte  m_front_brake_bias;          // front brake bias (percentage)

    byte  m_fuel_mix;                 // fuel mix - 0 = lean, 1 = standard, 2 = rich, 3 = max

    byte  m_currentLapInvalid;      // current lap invalid - 0 = valid, 1 = invalid

    byte  m_tyres_damage[4]; // tyre damage (percentage)

    byte  m_front_left_wing_damage; // front left wing damage (percentage)

    byte  m_front_right_wing_damage; // front right wing damage (percentage)

    byte  m_rear_wing_damage; // rear wing damage (percentage)
    byte  m_engine_damage; // engine damage (percentage)

    byte  m_gear_box_damage; // gear box damage (percentage)

    byte  m_exhaust_damage; // exhaust damage (percentage)

    byte  m_pit_limiter_status; // pit limiter status – 0 = off, 1 = on

    byte  m_pit_speed_limit; // pit speed limit in mph



    // Car data

    byte  m_num_cars;               // number of cars in data

    byte  m_player_car_index;         // index of player's car in the array

    CarUDPData  m_car_data[20];   // data for all cars on track

};


struct CarUDPData

{

    float m_worldPosition[3]; // world co-ordinates of vehicle

    float m_lastLapTime;

    float m_currentLapTime;

    float m_bestLapTime;

    float m_sector1Time;

    float m_sector2Time;

    float m_lapDistance;

    byte  m_driverId;

    byte  m_teamId;

    byte  m_trackPosition;     // track positions of vehicle

    byte  m_currentLapNum;

    byte  m_tyreCompound; // compound of tyre – 0 = ultra soft, 1 = super soft, 2 = soft, 3 = medium, 4 = hard, 5 = inter, 6 = wet

    byte  m_inPits;           // 0 = none, 1 = pitting, 2 = in pit area
```

> byte m_sector; // 0 = sector1, 1 = sector2, 2 = sector3
>
> byte m_currentLapInvalid; // current lap invalid - 0 = valid, 1 = invalid
>
> };
>
> Changes to data since F1 2016
>
> · Brakes temperature value is now in Centigrade, rather than Kelvin
>
> · Wheel data has all been converted to 4 value arrays, with order Rear Left, Rear Right, Front Left & Front Right. The actual data has not changed
>
> · Fixed car race position

Excellent read.

cjorgens79

Members

⊕ 3

168 posts

Posted June 21, 2017

> bax said:
>
> Some question about struct CarUDPData, since my field of interest is the live coverage overlay
>
> 1) what is exactly this parameter?
> *byte m_trackPosition; // track positions of vehicle*
> is it similar to *float m_car_position; // car race position* ?
> If not... I can calculate the race car position using m_lapDistance & m_currentLapNum but ... but wow :#
>
> 2) calculating the delay between 2 drivers is not a problem using timers but there is one important info missing to determine the FINAL race car position: PENALTIES !! Time penalties greatly affect the final classification of the race in case of strict rules and it is typical to give this information to spectators during a live coverage.
> I don't think you will change now the total amount of bytes for

> each car but maybe you can send via UDP:
>
> *float m_car_penalty; // (total amount of penalties not already paid during pit stops) (s)*
> instead of
> *float m_bestLapTime // (s)*
>
> since best lap time is (not so important and) easily deductible from last lap time if following the race from the beginning.

I took m_trackPosition to mean race position, since its a byte. Car position on the track is m_WorldPosition[]

bestLapTime is important if the game allows players to join a multi player session while its in progress (eg qualifying or practice). It is then essential to have best lap time provided so anyone joining mid session can see the fastest times on the leaderboard.

having split time ahead provided by the telemetry means that players using slower telemetry tick update speeds can still have accurate split times, as calculating it yourself gets progressively less accurate as the tick rate is reduced.

There is still room in the cars struct to also include penalties, it may not need to be a float though as i presume its usually just +x seconds?

I haven't had a chance to test this yet, but im curious whether the telemetry will stop streaming the moment the player crosses the finish line at the end of a race, or whether it will continue to stream until all cars results are known. We really need it to go to the end so we can get accurate final results from the game.

bax

B

Members
● 0
65 posts

Posted June 21, 2017

> cjorgens79 said:
>
> There is still room in the cars struct to also include penalties, it may not need to be a float though as i presume its usually just +x seconds?

I agree, there is still room... and yes... penalties are usually sum of +3, +5 or +10, only integer values and approximately +30 cause a

DSQ (so the amount will not be so big)

BDub1027

B

Members
● 0
16 posts

Posted June 22, 2017

@Hoo This is awesome, and already such a big improvement over last year. Would it be possible to add a packet with the car's setup data as well? I would love to be able to use the lap data plus setup data to be able to compare and fine tune the setups in testing. One packet per lap, or even just when the setup changes would do it.
Really looking forward to release day.

dwin20

D

Members
⊕ 81
334 posts

Posted June 26, 2017

This is really good news! I have really liked a lot of what is being discussed in the upcoming F2017 game, but the biggest weakness this game has had is telemetry. While this doesn't take care of in-game telemetry (which I am still hoping for) it will at least provide better data for third-party applications - that with enhanced data I will be glad to pay for. @Hoo is there any possibility for in-game telemetry? Thanks

Kafumanto

K

Members
⊕ 3
66 posts

Posted June 27, 2017

> BDub1027 said:
>
> @Hoo This is awesome, and already such a big improvement over last year. Would it be possible to add a packet with the car's setup data as well? I would love to be able to use the lap data plus setup data to be able to compare and fine tune the setups in testing. One packet per lap, or even just when the setup changes would do it.
> Really looking forward to release day.

Low frequency packets with the car setup would be great! +1 :)

1 2 3 4 5 6 NEXT » Page 1 of 11

GO TO TOPIC LISTING
Technical Assistance

Home > F1 Games > F1® 2019 Game Forum > Technical Assistance > All Activity

CONTACT US
TERMS & CONDITIONS
PRIVACY POLICY