CAN data aquisition module transmission protocol

ADVANCED SECURITY SYSTEMS

ver. 0.09

The purpose of the following document is to present the protocol allowing communication between the host device (security / telemetry / monitoring system) and the family of OEM Digital Systems CAN bus data aquisition modules / products.

1. Basic information

1.1. Protocol is in some terms similar to Modbus-ASCII. The communication is based on a Master-Slave querry/response) principle. Host device is polling the Digital Systems module. The module responds only to the querries and cannot initiate the transmission. The transmission is half-duplex, depending on the device, via one of RS standards: RS485 or RS232 or RS232 3.3V level lines. Transmission parameters are: 115200 baud, 8bits, 1 stop bits, no parity (115200,8N1).

1.2. Byte values are coded in hex-ASCII form: every byte is sent as 2 ASCII characters, from 00 to FF. The accepted character set consists of hexadecimal digits 0,1,2,3,4,5,6,7,8,9,0,A,B,C,D,E,F,a,b,c,d,e,f and additional frame separating characters <:>, <$>, <*>.

1.3. Frame structure

querry	<:> <*function code*> <*querry data*> <CRC8> <CR><LF>
response (ok)	<$> < *function code* > <*response data*> <CRC8> <CR><LF>
response (error)	<$> < *function code* | 0x80> <*error code* > <CRC8> <CR><LF>

The frame starts with ':', then, function code and data bytes with querry data are being sent. Then the CRC8 is sent and finally, the <CR> and <LF> characters limits the frame.

1.4. Checksum

The CRC8 is used as a protection against transmission errors. The CRC8 is calculated from <*function code*> and <*querry data*> binary values (not hex-ASCII characters!).

1.5. Error codes.

If the function cannot be executed / is not implemented in module / is not recognized the device returns the function code with the highest bit (128, 0x80) set. The first data byte contains error code. The error response is 2 bytes long.

0x01 - general error - no additional information is provided

0x02 - function not known - function with given code is not implemented / recognized in the module (e.g. function code not known)

0x03 - function not allowed - the function code is recognized in the protocol , but the function cannot be executed by the this module (e.g. module does not provide such data)

0x04 - incorrect function data - the additional querry parameters in querry data are not correct

2. Description of module functions

The module functions are described below. Every function is recognized by its **function code**.

2.1. Configuration and HW/SW firmware check functions.

0x05 checking CAN bus configuration and communication with CAN module

The function checks the presence and the preset configuration parameters of DS module. The function return the CAN bus configuration "level". The CAN bus parameter "level" is a value which describes how the module reads information from vehicle CAN bus (bus speed and CAN frame identifiers, scaling etc.)

Latest list with CAN bus levels for given vehicles can be obtained from DigitalSystems.

Tx: *:05*

Rx: *$05 <CAN1level>< CAN2level>< CAN3level>< CAN4level>< LINlevel>*

E.g.

Tx: *:053F<cr><lf>*

Rx: *$057B5800000032<cr><lf>*

means that the CAN module works OK and is set: CAN1 for level 123 (Mercedes Axor, dedicated OEM CAN bus), CAN2 level 88 (CAN bus according to J1939 - in this vehicle tachograph CAN) and that the module does not use LIN bus.

The response *$050300000000A5<cr><lf>* means that module uses only CAN1, CAN2, CAN3, CAN4 and LIN are not used. CAN2-4 are disabled.

0x06 CAN configuration of DS module

The function configures the CAN1 and CAN2 of the module to work with a given vehicle.

Tx: *:06 <CAN1level>< CAN2level>>< CAN3level>< CAN4level>< LINlevel><CRC>**

Rx: *$06 <CAN1level>< CAN2level>>< CAN3level>< CAN4level>< LINlevel><CRC>**

Eg.

Tx *:* *:060300000000FC<cr><lf>*

Rx: *$060300000000FC<cr><lf>*

configures the module to work in VAG group vehicles (VW Golf V/Passat B6 based) - connection of CAN1 bus. The CAN2 and LIN are not used.

0x11 Hardware and firmware version check

The function configures the CAN1 and CAN2 of the module to work with a given vehicle.

Tx: :11

Rx: $11<HW version><SW version>

<HW version> 16bit integer - module type identifier. 0x1cc=DS460, 0x1cd=DS461, 0x1d6=DS470

<SW version> 16bit integer 16bit, firmware version identifier 0x0001 for the version 0001

E.g.

Tx: :11C3<cr><lf>

Rx: $1101D6000BD6<cr><lf>

means DS470-0011

2.2. Vehicle state information reading (vehicle protection systems)

In the following chapter, the functions concerning the information about vehicle state, applicable for vehicle protection/security systems are described. The data are placed in variables named SIGNAL1, SIGNAL2 and REMOTE. These variables hold the information about status of door switches, ignition switch, parking lights, hazard lights, about locking and unlocking the vehicle with OEM remote, triggering of OEM alarm. The information is coded in bits, as described in Tab.1.

Tab. 1: Vehicle state data

SIGNAL1		
Bit	**Signal**	**Description**
1	Door FL	Front left door switch status, 1 = open
2	Door FR	Front right door switch status, 1 = open
4	Door RL	Rear left switch status, 1 = open
8	Door RR	Rear right door switch status, 1 = open
16	bonnet	Bonnet switch status, 1 = open
32	Trunk	Trunk switch status, 1 = open
64	ACC	1 = key in ignition switch or key in ACC position (e.g. radio enabled)
128	IGNITION	1 = ignition switch in ignition state

SIGNAL2		
Bit	**Signal**	**Description**
1	Hazard lights	1= hazard lights enabled - lamp blinks
2	Reverse gear	1= reverse gear selected
4	Parking lights	1 = parking lights enabled
8	-	
16	Locked without trunk state	1= the vehicle was locked by OEM remote but the trunk has been opened with the 3rd button on remote.
32	Locked state	1= the vehicle was locked by OEM remote
64	OEM alarm	1= the OEM alarm in the vehicle is alarming
128	Semi-alarm	(*)

(*) - the device realises the vehicle protection algorithm (semi-alarm). Opening any door, trunk, bonnet, or switching on the ignition if the vehicle is locked by OEM remote is recognized as an alarm state and signalled on this bit until the driver press the lock or unlock button on remote.

If the 3rd button on remote is used, the protection of trunk is temporary disabled (signalled by *Locked wo trunk state* bit), The protection returns:

- after the LOCK button on remote has been pressed.
- after 5 seconds after closing the trunk (vehicles in which when trunk can be opened only once),
- after 30 seconds , if the trunk has not been opened

SIGNAL3		
Bit	**Signal**	**Description**
1	lock	Vehicle was locked by OEM remote
2	unlock	Vehicle was unlocked by OEM remote
4	unlock trunk	The trunk was opened by OEM remote
8	-	-
16	-	-
32	-	-
64	-	-
128	-	-

The SIGNAL variables contain information about actual (momentary) state of the vehicle. The module has also the "latched" copies of that variables - noted L_SIGNAL1, L_SIGNAL2, L_REMOTE.
If the event occurs (e.g. the door is opened) the bit is set and latched even if the door is closed again. That solution allows to detect even short press on remote or fast opening and closing the bonnet switch, even if the DS module is being polled with low frequency. After reading of latched values, they should be cleared.

0x01 momentary values of SIGNAL1 / SIGNAL2 / REMOTE variables

The 0x01 function reads values of the variables specified above, thus describing the momentary state of the vehicle.

Tx: *:01*
Rx: *$01<SIGNAL1><SIGNAL2><REMOTE>*
E.g.
Tx *:015E<cr><lf>*
Rx *$0182010066<cr><lf>*
means, that at the given moment: ignition is on, front right door is open, hazard lights are on, the remote control buttons were not pressed, the vehicle is not locked.

0x02 clear latched variables L_SIGNAL1 / L_SIGNAL2 / L_REMOTE

The 0x02 function clears the latched copies of variables specified above.

Tx: *:02*
Rx: *$02*
E.g.
Tx: *:02BC<cr><lf>*
Rx: *$02BC<cr><lf>*

0x03 state of latched variables L_SIGNAL1 / L_SIGNAL2 / L_REMOTE:

Tx: *:03*
Rx: *$03<L_SIGNAL1><L_SIGNAL2><L_REMOTE>*
E.g.
Tx: *:03E2<cr><lf>*
Rx: *$0382010061<cr><lf>*
means that from last call of the 0x02 command: the ignition was (or is) switched on, the front right door was (or is) opened, the hazard lights were (or are) on, the remote was not used and the vehicle is not locked.

0x04 state of mixed variables SIGNAL1 / SIGNAL2 / L_REMOTE. Clear latched variables.

This function reads momentary values of SIGNAL1 and SIGNAL2 and latched variable L_REMOTE. Moreover, it clears latched variables L_SIGNAL1, L_SIGNAL2, L_REMOTE. Such a mix is convenient for the vehicle security systems. The momentary values of signals give the current information about the

vehicle state. The change of value of REMOTE variable when using the remote can be so short, that if the polling of module is not very frequent, the detection of remote can be missed. That is why it so convenient to use this function - combined reading of momentary switches and latched remote and clearing the latch variables.

Tx: *:04*

Rx: *$04<SIGNAL1><SIGNAL2><L_REMOTE>*

E.g.

Tx: *:0461<cr><lf>*

Rx: *$0482010252 <cr><lf>*

means that: ignition is on, front right door is opened, hazard lights are on and from the last call of 0x02 or 0x04 the UNLOCK button on remote has been pressed.

2.3. CAN bus vehicle control.

This chapter describes the functions that allow controlling the vehicle equipment via CAN bus. The module can control, depending on the vehicle, various devices (hazard lights, central door locking). The list of control signals is presented in the description of function 0x07.

0x07 CAN bus vehicle control

On some vehicles, the module can send CAN bus frames that allow to lock and unlock central door locking system, roll-up power windows, enable hazard lights.

Tx: *:07 <CAN bus ID><command ID>*

Rx: *$07 <CAN bus ID><command ID>*

<*CAN bus ID*> - the number describing on which CAN bus in the module the control frames should be sent. The valid numbers are 1 or 2.

<*command ID*>

1	lock central door locking system
2	unlock central door locking system
3	unlock trunk
4	blink the vehicle hazard lights (start blinking)
5	stop blinking of the vehicle hazard lights *(it is required to send alternatively the code 4 and the code 5 in the desired pattern of blinking).*
6	start rolling-up power windows
7	stop rolling-up power windows

E.g.

Tx: *:070101E0<cr><lf>*

Rx: *$070101E0<cr><lf>*

locks the vehicle central door locking system via CAN.

2.4. Reading vehicle monitoring data.

The following chapter presents the functions which allow reading vehicle data, required by monitoring and telemetry systems. The data can be read from one of following CAN buses:

a) dedicated vehicle OEM CAN bus - mostly passenger vehicles
(the list of signals present in given vehicles is supported in electronic form by Digital Systems).

b) CAN-FMS (Fleet Management Systems) / J1939 - CAN bus dedicated for connection of telemetry and monitoring systems to trucks and buses or truck/bus CAN-bus according to SAE J1939 e.g. tachograph CAN bus.

c) EOBD CAN bus (On Board Diagnosis) in passenger vehicles; connection to EOBD socket, obligatory in US or EU market vehicles. The data present in vehicles is usually speed, RPM, engine temperature and VIN.

The data read from the vehicle has been divided into 2 sets.

- basic vehicle data set, read both in passenger vehicles, trucks, lorries and buses via all of 3 CAN buses described above (a/b/c).
- extended set of vehicle data, read only on FMS/J1939 CAN bus.

Tab. 2. Basic vehicle data

Data ID	Data name	Data type	Physical value scaling	parameter ID and FMS frame
1	Vehicle speed	int8	1bit = 1km/h, 0-254km/h	SPN1624/ TCO1
2	Engine RPM	int8	1bit = 50 rpm, 0-12700rpm	Non standard scale
3	Engine temperature	int8	1bit = 1°C, offset -40°C, -40°C-215°C	SPN110 / ET1
4	Accelerator pedal	int8	1bit = 0.4% 0-101.6%	SPN91 / EEC2
5	Engine Load	int8	1bit = 1% 0-125%	SPN92 / EEC2
6	Fuel level unit	int8	0/1/2/3	-
7	Fuel level	int8	1bit = 0.4% or 1bit= 0.5ltr	(*) SPN96/DD
8	Total mileage	int32	1bit = 5m	SPN917 / VDHR
9	Ambient temperature	int8	1bit = 1°C, offset -40°C, -40°C-215°C	(*)

Data scaling is presented in Tab.2. Most of vehicle data is scaled according to J1939 / FMS scales. In order to simplify the scale search in pdf documentation of J1939 or FMS, the parameter name (SPN... from J1939 spec.) and frame name (from FMS spec.) have been given in 3rd column.
For the practical reasons, non-standard scales are used for:

- engine rpm (pos.2)
- ambient temperature (pos. 9) - the scale is the same as for the engine temperature (pos. 3)
- fuel level (pos. 7) - the data is being sent in 1 of 3 possible formats, depending on the vehicle type (the format of data which is present in the vehicle). The format is specified in pos. 6 - fuel level unit

fuel level unit =0 - no fuel level information in the vehicle,
fuel level unit =1 - the fuel level is being read as the [%] of fuel tank capacity - occurs in FMS/J1939 trucks, some personal vehicles, EOBD data. The scale is same as in the FMS specification: 1bit = 0.4%.
fuel level unit =2 - the fuel level is being read in liters (some personal vehicles). The scale is 1bit=0.5ltr.
fuel level unit =3 - the fuel level is being read in liters, but it is the number of litres below the maximum capacity of fuel tank. Some GM-Opel vehicles.

The extended set of vehicle data, which can be read from the FMS / J1939 (excluding the data already presented in basic set of data) is described in Tab. 3. As the parameter values/scaling is is quite complicated (but strictly following the FMS/J1939 specification) the parameter names, SPN... numbers and FMS frame names (e.g. TCO1) are presented in order to simplify the data search in J1939 / FMS specification.

Tab. 3. Extended set of vehicle data (FMS / J1939 data)

Data ID	Data name	Data type	Physical value	parameter ID and FMS frame
10	TachoOverspeed	int8	tacho - vehicle overspeed	SPN1614/ TCO1
11	TachoDirection	int8	tacho - vehicle direction (0=forward)	SPN1619/ TCO1
12	TachoMotion	int8	tacho - 1=vehicle is in motion	SPN1611/ TCO1
13	TachoDrv1WorkingState	int8	tacho - 1st driver status (work/rest...)	SPN1612/ TCO1
14	TachoDrv2WorkingState	int8	tacho - 2nd driver status (work/rest...)	SPN1613/ TCO1
15	TachoDrv1Card	int8	tacho - 1st driver card status	SPN1615/ TCO1
16	TachoDrv2Card	int8	tacho - 2nd driver card status	SPN1616/ TCO1
17	TachoDrv1Time	int8	tacho - 1st driver work time	SPN1618/ TCO1
18	TachoDrv2Time	int8	tacho - 2nd driver work time	SPN1619/ TCO1
19	TachoPerformance	int8	tacho - workmode 0=normal, 1=performance analysis	SPN1620/ TCO1
20	TachoHandling	int8	tacho - request handling (e.g. no paper in printer)	SPN1621/ TCO1
21	TachoSystemEvent	int8	tacho - system event is present (e.g. system failure)	SPN1622/ TCO1
22	PTOstate	int8	Power take-off state	SPN 976 / CCVS
23	PTOengaged	int8	At least one Power take-off is enabled	SPN3948/ PTODE
24	ClutchSwitch	int8	Clutch is pressed	SPN 598 / CCVS
25	BrakeSwitch	int8	Brak is applied	SPN 597 / CCVS
26	CruiseControlSwitch		Cruise Control is enabled	SPN 595 / CCVS
27	TotalFuelUsed	int32	Total fuel used by the vehicle	SPN250 / LFC1
28	HR_TotalFuelUsed	int32	High accuracy total fuel used by the vehicle (up to 0.001ltr)	SPN5054/ HRLFC
29	EngineTotalHrs	int32	Total engine working time	SPN247 / HOURS
30	ServiceDistance	int16	Distance till next service	SPN914 / SERV
31	AxleWeight [4]	4x int16	Vehicle weight on axles	SPN582 / VW
32	FMS_SWtruck[2]	2x int8	FMS trucks software version (in the vehicle)	SPN2806/ FDD1
33	FMS_SWbus[2]	2x int8	FMS buses software version (in the vehicle)	SPN2805/ FDD1
34	FMS_SWreqsupp	int8	Does vehicle handle requests	SPN2804/ FDD1
35	FMS_SWdiagsupp	int8	Does vehicle handle diagnostics	SPN2803/ FDD1
36	FuelRate	int16	Fuel consumption in ltrs/hour	SPN183/ LFE
37	InstanousFuelEconomy	int16	Fuel consumption in km/ltr	SPN184/ LFE

0x08 reading the basic set of vehicle data

The function reads basic set of vehicle data (which occurs in most vehicles) from the device. The data value is being send as the hexadecimaly coded integer (2-8 characters). If the data is not present in the vehicle, the maximal value for that type of data (0xff or 0xffff or 0xffffffff) is being sent. The values are send with scaling as specified in Tab.1.

Tx: *:08*

Rx: *$08<vehicle speed><engine RPM><engine temperature>*
<acceleration pedal><engine load>
<fuel level unit><fuel level>
<total mileage><ambient temperature>

E.g.

Tx: :08C2*<cr><lf>*

Rx: *$082A1482FFFF00FFFFFFFFFF3EF7<cr><lf>*

means that the speed is 42km/h, engine RPM is 1000RPM, the accelerator pedal is 4% pressed, engine temperature is 90°C, fuel scale is in liters, 49 ltrs are in the fuel tank, no information concerning mileage, ambient temperature is 22°C.

0x09 reading VIN (vehicle identification number)

The function reads vehicle VIN. The VIN is read as the 20-bytes array of ASCII characters, limited by 0x00 (null), coded as 40 hexadecimal characters.

Tx: *:09*

Rx: *$09< VIN >*

E.g.

Tx: *:099C<cr><lf>*

Rx: *094A4D42584A43573857385A343032333638200000<crc><cr><lf>*

means that the VIN is JMBXJCW8W8Z402368

0x0A reading single data from the basic or extended set of data.

The function reads the value of single vehicle data

Tx: *:0A<data ID >*

Rx: *$0A<data ID> <data value>*

<*data ID*> is the No. of data in Tab.1. or Tab.2. The data value is being send as the hexadecimaly coded integer (2-8 characters). If the data is not present in the vehicle, the maximal value for that type of data (0xff or 0xffff or 0xffffffff) is being sent. The 0x0A function allows reading data from basic set 0f data (*data ID* from Tab.1) or extended set of data (*data ID* from Tab.2)

E.g.

Tx: :0A0BC7<cr><lf> querry about FMS: **TachoDirection** - direction of vehicle movement

Rx: $0A0B0049<cr><lf> vehicle drives forward

or $0A0B0117<cr><lf> vehicle drives backward

or $0A0B03AB<cr><lf> data not present in the vehicle

0x0B reading tachograph data (FMS/J1939)

The function reads all the tachograph data (FMS / J1939): pos. 10-21 from Tab.2

Tx: *:0B*
Rx: *$0B<TachoOverspeed><TachoDirection><TachoMotion><TachoDrv1State>*
<TachoDrv2State><TachoDrv1Card><TachoDrv2Card><TachoDrv1Time><TachoDrv2Time>
<TachoPerformance><TachoHandling><TachoSystemEvent>

E.g.
Tx: *:0B20<cr><lf>*
Rx: *$0B000301020000010304030303 68<cr><lf>*
means that the maximum speed is not exceeded, no information about vehicle direction is present, the vehicle is in motion, driver No1 is working, driver No2 is resting, Driver 1 card is present, driver 2 card is present, in 15 minutes, the 1st driver reaches 9 hours of work, the driver No.2 rests 9 hours.

0x0C reading switch data (FMS/J1939)

The function reads data concerning brake switch, clutch switch, power take-off and cruise control - pos. 22-26 from Tab.2.
Tx: *:0C*
Rx: *$0C<PTOstate><PTOengaged><ClutchSwitch><BrakeSwitch><CruiseControlSwitch><CRC>**
E.g.
Tx: *:0CA3<cr><lf>*
Rx: $0C05010000017E<cr><lf>
Means that the Power Take-Off is on, there is 2nd frame with information about PTO (*PTO engaged*), brake and clutch pedals are depressed and the Cruise Control is on.

0x0D reading fuel level, mileage, engine hours of operation and distance to service

The function reads a mixed set of data concerning the operation of the vehicle, both from basic and the extended set of data: fuel level, total fuel used by the vehicle, fuel economy in both units (fuel rate and instanous fuel economy), total working hours of the engine, total mileage and distance to next service.
Tx: *:0D*
Rx: *$0D<Fuel level unit><Fuel level><TotalFuelUsed><FuelRate>*
<InstanousFuelEconomy><EngineTotalHrs><Total Mileage><ServiceDistance>
E.g.
Tx: *:0DFD<cr><lf>*
Rx: $0D01C8000007D001903C0100004E20016E3600854F27<cr><lf>
means that: fuel level is read in [%] and is 80% of fuel tank capacity, the vehicle has used 1000ltr, has driven 120000km and the engine has worked 1000 hours since being new. Current consumption is 20 ltrs/hour or 30km/ltr.

0x0E reading vehicle weight (FMS/J1939)

The function reads FMS data concerning the vehicle (axle) weights. The function read weight of 4 axles. The 1st axle is the front one. Actually, not all axle weights are being measured in the vehicle - see example.

Tx: *:0E*

Rx: *$0E<1st axle weight><2nd axle weight><3rd axle weight><4th axle weight>*

E.g.

Tx: *:0E1F<cr><lf>*

Rx: *$0E27104E20FFFF00008A<cr><lf>*

means that the 1st axle weight is 5000kg, 2nd axle weight is 10000kg, 3rd axle is not measured and 4th axle weight is 0kg, which means that probably it is raised.

2.5. Checking presence of the data in the vehicle

The following chapter describes, how to check instantly, which data from Tab.2 and Tab.3 are transmitted (present) in the vehicle and which are not. There are 6 STATUS variables with bits corresponding to vehicle data presented in Tab. 2 and Tab. 3.

These STATUS variables are cleared e.g. after reset or after reprogramming the module or after requesting a dedicated function in this protocol.

The first occurrence of data sets the corresponding bit in one of STATUS variables. The relation between bits and data is presented in Tab. 4.

Tab. 4. The flags concerning occurence of data in the vehicle

STATUS 1 - vehicle state / VIN		
Bit	**Data**	**Data ID**
1	VIN occured	-
2	Door open signals were read	-
4	Bonnet open signal was read	-
8	Trunk open signal was read	-
16	Ignition signal was read	-
32	OEM alarm signal was read.	-
64	Remote Control signals were detected	-
128	-	

STATUS 2 - basic data set		
Bit	**Data**	**Data ID**
1	Prędkość pojazdu	1
2	Obroty silnika	2
4	Temperatura silnika	3
8	Położenie pedału gazu	4
16	Obciążenie silnika	5
32	Poziom paliwa	7
64	Przebieg całkowity	8
128	Temperatura zewnętrzna	9

STATUS 3 - FMS data set - tachograph		
Bit	**Data**	**Data ID**
1	TachoDrv1WorkingState	13
2	TachoDrv2WorkingState	14
4	TachoDrv1Card	15
8	TachoDrv2Card	16
16	TachoDrv1Time	17
32	TachoDrv2Time	18
64	TachoOverspeed	10
128	TachoMotion	12

STATUS 4 - FMS data set - tachograph, fuel data		
Bit	**Data**	**Data ID**
1	TachoDirection	11
2	TachoPerformance	19
4	TachoHandling	20
8	TachoSystemEvent	21
16	TotalFuelUsed	27

32	HR_TotalFuelUsed	28
64	FuelRate	36
128	InstanousFuelEconomy	37

STATUS 5 - FMS data - switches, Total Hours, Service Distance		
Bit	**Nazwa wielkości**	**Data ID**
1	PTOstate	22
2	PTOengaged	23
4	ClutchSwitch	24
8	BrakeSwitch	25
16	CruiseControlSwitch	26
32	EngineTotalHrs	29
64	ServiceDistance	30
128		

STATUS ZMIENNYCH 6 - FMS data - axle weight / FMS gateway parameters		
Bit	**Nazwa wielkości**	**Data ID**
1	AxleWeight [axle 1]	31
2	AxleWeight [axle 2]	31
4	AxleWeight [axle 3]	31
8	AxleWeight [axle 4]	31
16	FMS software version - truck	32
32	FMS software version - bus	33
64	FMS Request Support	34
128	FMS Diagnostic Support	35

0x0F reading vehicle data occurrence STATUS variables

The function reads 6 status variables, thus giving information which data is present in the vehicle and which is not.

Tx: *:0F*
Rx: *$0F<STATUS1><STATUS2><STATUS3><STATUS4><STATUS5><STATUS6>*
E.g.
Tx: *:0F41<cr><lf>*
Rx: *$0F1E870000000090<cr><lf>*
means that the following data have been read from the given vehicle: VIN, vehicle speed, engine RPM, engine temperature. Moreover, the door, trunk, bonnet and ignition signals were detected.

0x10 clearing vehicle data occurence flags in STATUS variables

The function clears the flags corresponding to the vehicle data in STATUS variables. The flags are set after the device reads the signals from the vehicle, which occurs e.g. after switching on the ignition, opening door, using remote. The flags are automatically cleared if the configuration function 0x06 was used.
Tx: *:10*
Rx: *$10*
e.g. *:109D<cr><lf>*
$109D<cr><lf>