Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
Chipsmall
采购部

OBDII STREAMER FAMILY

User Implementation Document

Table of Contents

Document Revision History

Version	Date	Changes	Author
2.00	10/30/08	Brought over from V1 Streamer and updated for new/revised functionality	Larry Reeves
2.01	1/28/09	Revised based on comments from PSA and a team. review.	Steve Sagerian
2.02	08/3/09	Added Bootloader update to UPDATE_COMPONENT command	Alexandr Kolodinsky
2.03	11/30/09	Added Keepalive message comtrol bypass command and revised the document. Made some minor clarification changes	Alexandr Kolodinsky
2.04	1/26/2010	Reviewed and updated with final changes to V2 prior to beta release	Steve Sagerian
2.05	2/8/2010	Updates for KWP15 protocol and sleep mode is gone complete, no set sleep mode, no wake from sleep and no set low power mode behavior.	Steve Sagerian
2.06	3/12/2010	Update bypass protocol definition	Matt Ollayos
2.07	5/3/2011	Incorrect behavior found describing threshold updates.	Matt Ollayos
2.08	12/13/11	Updated based on Management comments	Chris Politsch
2.09	1/27/12	Removed unused messages	Chris Politsch
2.10	2/14/12	Updated Command 0x21	Chris Politsch
2.11	3/19/13	Removed unsupported parameters, bypass mode references	Larry Reeves

Communication Protocol Definition

All commands and responses to or from the OBDII Streamer are formatted in this style.

Byte	**Byte**	**Byte(s)**	**Byte**	**Byte(s)**	**Byte**
Start of Frame	Control Length	Control Bytes	Data Length	Data Bytes	Checksum

Start of Frame `0x01` – always 0x01.

Control Length The number of control bytes in the current message. The control length of a request or response varies by command and will be defined under each command's heading in this document.

Control Byte(s) The first control byte in a request is the command being sent. In a response the control byte will be 0x80 greater than the request control byte.

Each command or response may have additional control bytes as defined by the specific command or response. Each message must have at least one control byte.

Data Length The number of data bytes in the current message. Zero data byte messages are valid.

Data Bytes This field contains the data portion of the message.

Checksum The checksum is a one-byte sum of all bytes including Start of Frame, Control Length, Control Bytes, Data Length, and Data Bytes.

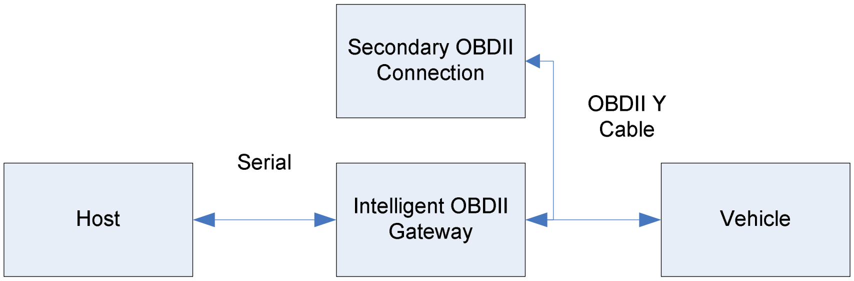
Checksum = *Sum* AND `0xFF`

Overview

The OBDII Streamer is an OBDII to Serial gateway. The raw data, multiple PIDs per parameter, and timing issues of the OBDII bus are abstracted from the host. The OBDII Streamer allows a developer to quickly access data from the vehicle without having knowledge of the OBDII bus.

The OBDII Streamer is intended to be permanently installed in a vehicle. Coupled with B&B Electronics OBDII Y-Cable the OBDII Streamer will immediately disconnect from the OBDII bus when another scan tool is connected to the bus.

The maximum update rate of data from the OBDII bus is 20Hz. Each parameter enabled constitutes one update from the vehicle. The update rate from the vehicle is not linked to any messages from the host. The host will be able to request data from the OBDII Streamer immediately after the previous response, but the OBDII data will be limited to an update rate of 20Hz.



Vehicle status parameters, such as the MIL, Fuel Level, and Battery Voltage, will be updated from the vehicle no faster than once every 2 seconds. The vehicle parameter DTC's will be updated from the vehicle no faster than once every minute. The host can request these parameters from the OBDII Streamer at any rate desired.

Startup Procedure

When the OBDII Streamer is powered on a CRC check will be performed to verify that the code on the device is valid. This will take approximately 10 seconds and during that time the device is unable to respond to commands.

On initial power on the OBDII Streamer will attempt to detect the vehicle type and supported parameters. This process can take up to 1 minute. During this time the OBDII Streamer will be responsive to host requests. If the OBDII Streamer has detected a vehicle type and supported parameters it will not attempt to detect them again, even if the vehicle type and supported parameters have changed.

While the host is detecting the vehicle's information most messages will be responded to with the following response.

```
0x01 0x01 0x81 0x00 0x83          //Vehicle Not (Yet) Detected Message
```

The following commands will respond normally during startup:

- READ_MODEL_NUMBER
- FIRMWARE_VERSION
- READ_SERIAL
- UPGRADE_FIRMWARE

When the OBDII Streamer has finished the initialization sequence it will send a DEVICE_CONFIGURED message.

After this message is received the OBDII Streamer will be ready for communications.

Non-Volatile Storage

Most configuration settings and parameters are stored in non-volatile memory and do not have to be reconfigured on startup. The following settings and parameters are saved in non-volatile memory.

- Serial Number
- BAUD
- OEM_ID
- Model Number
- VIN
- Firmware/EEPROM Versions
- Vehicle information (OBDII Bus and Available Parameters)
- Enabled/Disabled Parameters
- Update Modes (Time and Threshold based)

Transferring to a New Vehicle

The OBDII Streamer stores all vehicle information in non-volatile memory. To improve startup times in the event of a power loss all parameters will be reloaded from memory.

If the OBDII Streamer is removed from one vehicle and installed in another, the OBDII Streamer must be forced to detect. On startup the previous vehicle's data will be loaded. The behavior of an OBDII Streamer will be unpredictable when connected to a vehicle that is different than the vehicle detected. To force the OBDII Streamer to redetect vehicle information send the REDETECT_VEHICLE command.

Ignition Detection

The vehicles J1962 diagnostic connector is always powered, even with the vehicle ignition off. The OBDII Streamer detects the ignition status by analyzing data retrieved over the OBDII bus. There are two methods used to determine ignition status.

First, the engine speed and vehicle speed are monitored from the vehicle. When engine speed and vehicle speed are detected as zero for 2 minutes the OBDII Streamer will enter Standby mode.

Second, most vehicles stop responding to OBDII requests when the ignition is off. After 20 seconds of no responses from the vehicle the OBDII Streamer will enter Standby mode.

The OBDII Streamer polls the vehicle for ignition status every two seconds. When the ignition status changes from off to on the device will return to normal mode and will send an ignition status message to the host.

Low Power Modes

Standby Mode:

During Standby Mode the vehicle is polled periodically to find out its ignition status. Standby Mode will be the default mode used.

- Standby Mode is entered within 20 seconds of communications loss.
- Standby Mode is entered within 2 minutes of 0 RPM and 0 Vehicle Speed (Vehicle Speed was added as a condition because a hybrid vehicle can turn the engine off and be running on the electric engine).
- Standby Mode is exited within ten seconds after vehicle starts communicating
- LDV Streamer remains partially responsive to some communications while in Standby Mode.
- Maximum Power Consumption: 0.42W.
- Maximum Time from power-on to valid vehicle data: fifteen seconds when vehicle is running.
- When the vehicle’s battery voltage is below 12.6V the vehicle is no longer polled to save more power. When the ignition is turned on the alternator will increase the voltage above 12.6V and the maximum time to normal running mode will be met.

Special Cases:

- First Time Installation: The OBDII Streamer does not enter any low power modes until it has been installed on a vehicle and has detected the supported parameters.
- Firmware updates: After a firmware update, the OBDII Streamer will go into the appropriate power state for normal operation.
- Unintentional Reset (Battery removal, brownout, etc.): After an unintentional reset, the OBDII Streamer will go into the appropriate power state for normal operation.
- Move to Different Vehicle:
    - The LDV Streamer should remain in the appropriate low power mode until it has been determined that the engine is running.
    - The LDV Streamer requires a REDETECT_VEHICLE command to work properly after moving to a different vehicle.

Settings

The following variables make up all of the settings that should be known or changed when using the OBDII Streamer.

Overview

The OBDII Streamer is capable of transmitting messages to the host automatically based on user set criteria. By default these messages are disabled.

Time Based Updates

"Time Based" updates will provide an update of all enabled parameters at every time interval specified by the user.

Time based updating will work simultaneously with "Threshold Based" updating and "As Requested" updating.

Threshold Based Updates

"Threshold Based Updating" will send an update to the host whenever an enabled parameter with threshold checking enabled exceeds the threshold.

"Threshold Based" updating will work simultaneously with "Time Based" updating and "As Requested" updating.

As Requested Updates

"As Requested" update mode allows the host to request a single parameter, or all parameters, at any time and receive a snapshot of the parameter values.

"As Requested" update mode is not an exclusive mode. This means that "Time Based", and "Threshold Based" update modes can be enabled and parameters can still be requested at any time.

BAUD

The BAUD rate setting is only applicable to the RS-232 serial version of the OBDII Streamer. The BAUD rate will default to 115.2kbps.

OEM_ID

The OEM_ID field is available for the customer to program as they choose. This field is 10 bytes long and defaults to "LDV_OEM_ID".

An example use of this field is to key the software to a particular piece of hardware.

Update Mode

On initial power up all update modes are disabled. In order to use any of the update modes listed below they must be configured and enabled using the SET_UPDATE_MODE and CHANGE_PARAMETER_THRESHOLD commands.

- Time Based
- Threshold Based

In order to receive automatic messages based on time or threshold the CHANGE_PARAMETER_THRESHOLD command must be sent for each parameter.

Enable_Parameters

By default, all supported parameters will be updated from the vehicle. The Enable_Parameters command can be sent in order to enable updates on only the supported parameters.

Setting the Enable_Parameters command is desirable in some cases to increase the parameter update rate. The update rate for each parameter is equal to the total update rate from the vehicle divided by the number of enabled parameters.

The update rate from the vehicle is fixed and varies from vehicle to vehicle, but will not exceed 20Hz.

Parameters

This list matches a parameter with its identifier.

ID	Parameter	Return Size	Units	Range
0x00	Vehicle Speed	2 Bytes	MPH	0 to 160
0x01	Engine Speed	2 Bytes	RPM	0 to 16384
0x02	Throttle Position	2 Bytes	%	0 to 100
0x03	Odometer	4 Bytes	Miles	0 to 999,992
0x04	Fuel Level	2 Bytes	%	0 to 100
0x07	Engine Coolant Temp	2 Bytes	ºF	-40 to 983
0x08	Ignition Status	2 Bytes	On/Off	On/Off
0x09	MIL Status	2 Bytes	On/Off	On/Off
0x0C	Fuel Rate	2 Bytes	Gallons per Hour	0 to 29.99
0x0D	Battery Voltage	2 Bytes	Volts	0 to 18
0x0E	PTO Status	2 Bytes	On/Off	On/Off
0x0F	Seatbelt Fastened	2 Bytes	Yes/No	N/A
0x10	Misfire Monitor	2 Bytes	Status	N/A
0x11	Fuel System Monitor	2 Bytes	Status	N/A
0x12	Comprehensive Component Monitor	2 Bytes	Status	N/A
0x13	Catalyst Monitor	2 Bytes	Status	N/A
0x14	Heated Catalyst Monitor	2 Bytes	Status	N/A
0x15	Evaporative System Monitor	2 Bytes	Status	N/A
0x16	Secondary Air System Monitor	2 Bytes	Status	N/A
0x17	A/C System Refrigerant Monitor	2 Bytes	Status	N/A
0x18	Oxygen Sensor Monitor	2 Bytes	Status	N/A
0x19	Oxygen Sensor Heater Monitor	2 Bytes	Status	N/A
0x1A	EGR System Monitor	2 Bytes	Status	N/A
0x1B	Brake Switch Status	2 Bytes	Pressed/Not Pressed	N/A
0x22	Trip Odometer	4 Bytes	Miles	0 to 999,992
0x23	Trip Fuel Consumption	4 Bytes	Gallons	0 to 999,992

0x00 Vehicle Speed

Scaling Equation:
Result = Returned Data * 1 / 410
Units: Miles per hour

EX: Return Data = 26650
Result = 26650 * 1 / 410
Result = 65 Miles per hour

0x01 Engine Speed

Scaling Equation:
Result = Returned Data * 1 / 4
Units: Revolutions per Minute

EX: Return Data = 12000
Result = 12000 * 1 / 4
Result = 3000 Revolutions per Minute

0x02 Throttle Position

Scaling Equation:
Result = Returned Data * 1 / 655
Units: %

EX: Return Data = 12000
Result = 12000 * 1 / 655
Result = 18.32 % Throttle Pedal Position

0x03 Odometer

Scaling Equation:
Result = Returned Data * 1 / 1
Units: Miles

EX: Return Data = 58,125
Result = 58,125 * 1 / 1
Result = 58,125 Miles

0x04 Fuel Level

Scaling Equation:
Result = Returned Data * 1 / 655
Units: %

EX: Return Data = 23578
Result = 23578 * 1 / 655
Result = 35.99 %

0x07 Engine Coolant Temp

Scaling Equation:
Result = (Returned Data * 1 / 64) - 40
Units: ºF

EX: Return Data = 14080
Result = (14080 * 1 / 64) - 40
Result = 180ºF

0x08 Ignition Status

Return Value of 1 = Ignition Off
Return Value of 0 = Ignition On

0x09 MIL Status

Return Value of 1 = MIL Off
Return Value of 0 = MIL On

0x0C Fuel Rate

Scaling Equation:
Result = Returned Data * 1 / 2185
Units: Gallons per Hour

EX: Return Data = 25650
Result = 25650 * 1 / 2185
Result = 11.74 Gallons per Hour

0x0D Battery Voltage

Scaling Equation:
Result = Returned Data * 1 / 3641
Units: Volts

EX: Return Data = 45650
Result = 45650 * 1 / 3641
Result = 12.538 Volts

0x0E PTO Status

Return Value of 1 = PTO Off
Return Value of 0 = PTO On

0x0F Seatbelt Fastened

Return Value of 1 = Seat Belt not Fastened
Return Value of 0 = Seat Belt Fastened

0x10 — 0x1A Monitors

Return Value of 1 = Monitor Not Complete
Return Value of 0 = Monitor Complete

0x1B Brake Switch Status

Return Value of 1 = Brake Switch Off
Return Value of 0 = Brake Switch On

0x22 Trip Odometer

Scaling Equation:
Result = Returned Data * 1 / 10
Units: Miles

EX: Return Data = 58,125
Result = 58,125 * 1 / 10
Result = 5,812.5 Miles

0x23 Trip Fuel Consumption

Scaling Equation:
Result = Returned Data * 1 / 128
Units: Gallons
EX: Return Data = 8,128
Result = 8128 * 1 / 128
Result = 63.5 Gallons
NOTE: It is basically returned in Ounces.

Commands

0x02 – READ_MODEL_NUMBER

Description:
This command will return the model number as an alpha-numeric ASCII string.

The following model numbers can be returned depending on the type of Streamer used.
LDVDSV2-S RS-232 OBDII Streamer

Command to Send:
0x01 0x01 0x02 0x00 0x04

Response:
0x01 0x01 0x82 DL MODEL CS

Field Description:
MODEL Variable length model number up to 16 characters long. This field is returned as ASCII characters.

DL Data Length (Length of model number)

CS Summation checksum

Example:
TX: 0x01 0x01 0x02 0x00 0x04

RX: 0x01 0x01 0x82 0x09 0x4C 0x44 0x56 0x44 0x53 0x56 0x32 0x2D 0x53 0x12

The model number returned is LDVDSV2-S.

0x03 – GET_COMPONENT_VERSIONS

Description:
This command will return the version numbers for all updatable components in the device.

Command to Send:

```
0x01 0x01 0x03 0x00 0x05
```

Response:
`0x01 0x01 0x83 0x0F` SW1 SW2 SW3 HW1 HW2 HW3 DB1 DB2 DB3 SYS1 SYS2 SYS3 BL1 BL2 BL3 CS

Field Descriptions:

SW1-SW3	Software version digits 1-3
HW1-HW3	Hardware version digits 1-3
DB1-DB3	Database version digits 1-3
SYS1-SYS3	System manager version digits 1-3
BL1-BL3	Bootloader version digits 1-3
CS	Summation checksum

Example:

```
TX:   0x01 0x01 0x03 0x00 0x05

RX:   0x01 0x01 0x83 0x0F 0x02 0x00 0x06 0x02 0x00 0x03 0x01 0x08 0x00
      0x02 0x00 0x04 0x02 0x00 0x01 0xB3
```

The version numbers returned as hexadecimal are:

Software version	`0x02 0x00 0x0A = 2.0.10`
Hardware version	`0x02 0x00 0x03 = 2.0.3`
Database version	`0x01 0x08 0x00 = 1.8.0`
System manager version	`0x02 0x00 0x04 = 2.0.4`
Bootloader version	`0x02 0x00 0x01 = 2.0.1`

0x04 – OBDII_FIRMWARE_VERSION (Obsolete)

Description:
This command is no longer supported as of version 2.

0x05 – FIRMWARE_VERSION (Deprecated)

Description:
[Command 0x03 – GET_COMPONENT_VERSIONS should be used instead of this command.]
This command will return the version number of the software component.

Command to Send:
0x01 0x01 0x05 0x00 0x07

Response:
0x01 0x01 0x85 0x03 VER1 VER2 VER3 CS

Field Description:

VER1	Hex version number digit 1
VER2	Hex version number digit 2
VER3	Hex version number digit 3
CS	Summation checksum

Example:
TX: 0x01 0x01 0x05 0x00 0x07

RX: 0x01 0x01 0x85 0x03 0x01 0x00 0x01 0x8C

The version number returned is 0x01 0x00 0x01 = 1.0.1

0x06 – DATABASE _VERSION (Deprecated)

Description:
[Command 0x03 – GET_COMPONENT_VERSIONS should be used instead of this command.]
This command will return the version number of the database component.

Command to Send:
0x01 0x01 0x06 0x00 0x08

Response:
0x01 0x01 0x86 0x03 VER1 VER2 VER3 CS

Field Description:

VER1	Hex version number digit 1
VER2	Hex version number digit 2
VER3	Hex version number digit 3
CS	Summation checksum

Example:
TX: 0x01 0x01 0x06 0x00 0x08

RX: 0x01 0x01 0x86 0x03 0x01 0x00 0x01 0x8D

The version number returned is 0x01 0x00 0x01 = 1.0.1

0x07 – READ_SERIAL_NUMBER

Description:
This command will return the ten digit hardware serial number. The serial number is programmed at manufacturing time and should match the serial number sticker on the side of the hardware.

Command to Send:
0x01 0x01 0x07 0x00 0x09

Response:
0x01 0x01 0x87 0x0A SERIAL CS

Field Description:

SERIAL	10 digit serial number returned as numeric ASCII characters.
CS	Summation checksum

Example:
TX: 0x01 0x01 0x07 0x00 0x09

RX: 0x01 0x01 0x87 0x0A 0x31 0x35 0x33 0x31 0x38 0x32 0x36 0x34 0x33 0x37 0x9B

The serial number returned is 1531826437.

0x09 – CONFIG_OEM_ID

Description:
This command will set or read the 10 digit OEM ID. Each digit of the OEM_ID is 1 byte and can be any possible value. The command to write the OEM ID always expects 10 digits/values and will give an error with any values less than or greater than 10.

The OEM_ID will be saved in non-volatile memory.

The first byte of the data field indicates if the OEM_ID will be read or written.
0x00 = read
0x01 = write

Command to send to read the OEM_ID:
0x01 0x01 0x09 0x01 0x00 0x0C

Response:
0x01 0x01 0x89 0x0B 0x00 OEM_ID CS

Command to send to write the OEM_ID:
0x01 0x01 0x09 DL 0x01 OEM_ID CS

Response:
0x01 0x01 0x89 0x0B 0x01 OEM_ID CS

Field Description:

OEM_ID	10 digit OEM_ID
DL	Data field length
CS	Summation checksum

Example: Set the OEM_ID
TX: 0x01 0x01 0x09 0x0B 0x01 0x01 0x35 0x73 0x99 0x24 0x72 0xF3 0x17 0xAC 0xBB 0x60

RX: 0x01 0x01 0x89 0x0B 0x01 0x01 0x35 0x73 0x99 0x24 0x72 0xF3 0x17 0xAC 0xBB 0xE0

Example: Read the OEM_ID
TX: 0x01 0x01 0x09 0x01 0x00 0x0C

RX:
0x01 0x01 0x89 0x0B 0x00 0x01 0x35 0x73 0x99 0x24 0x72 0xF3 0x17 0xAC 0xBB 0xDF

0x15 – SERIAL_BAUD

Description:
This command will read or write RS-232 Baud Rate settings.

The configuration data is stored in non-volatile memory and will be loaded on system power-on.

Command to Send:
0x01 0x01 0x15 0x03 RD/WR BAUD 0x01 CS

Response:
0x01 0x01 0x95 0x03 RD/WR BAUD 0x01 CS

Field Description:

RD/WR

This indicates if the message will be a read or a write.
0x00 = read   0x01 = write

Note: If the request message is a configuration read then the BAUD and STATUSMSG fields should not be included in the request message, and the data length will be 1 byte.

BAUD

BAUD setting
0x00 = 9600kbps   0x01 = 19200kbps
0x02 = 38400kbps   0x03 = 56000kbps
0x04 = 115200kbps (Default)

CS

Summation checksum

Example: Baud = 19200kbps
TX: 0x01 0x01 0x15 0x03 0x01 0x01 0x01 0x1D

RX: 0x01 0x01 0x95 0x03 0x01 0x01 0x01 0x9D

0x20 – GET_SUPPORTED_PARAMETERS

Description:
This command will return an identifier byte for each supported parameter.

Only supported parameters will be requested from the vehicle. Parameter values reported for an unsupported parameter are invalid. Supported parameters are available after the vehicle has been detected.

Command to Send:
0x01 0x01 0x20 0x00 0x22

Response:
0x01 0x01 0xA0 DL SUPPORTED_PARAMETERS CS

Field Description:

DL	Data Length Byte
SUPPORTED_PARAMETERS	Variable length data field containing one byte for each supported parameter.
CS	Summation checksum

Example:
TX:
0x01 0x01 0x20 0x00 0x22

RX:
0x01 0x01 0xA0 0x06 0x00 0x02 0x03 0x08 0x09 0x11 0xCF

The supported parameters in this example are:
0x00 – Vehicle Speed
0x02 – Throttle Position
0x03 – Odometer
0x08 – Ignition Status
0x09 – MIL Status
0x11 – Fuel System Monitor

0x21 – ENABLE_PARAMETERS

Description:

This command will enable or disable only the specified parameters to be updated from the vehicle. If parameters are enabled that are not supported by the vehicle the OBDII Streamer will send a response indicating which parameters are not supported.

By default all supported parameters are enabled and queried from the vehicle. Because the update rate of the vehicle is limited to a maximum of 20 Hz, it may be desirable to disable some parameters to get better update rates on parameters of interest.

Command to Send:

0x01 0x01 0x21 DL RD/WR ENABLE/DISABLE PARAMETER_LIST CS

Response to write:

0x01 0x01 0xA1 DL 0x01 ENABLE/DISABLE NOT_SUPP CS

Response to read:

0x01 0x01 0xA1 DL 0x00 PARAMETER_LIST CS

Field Description:

DL
Data Length Byte

RD/WR
This indicates if the message will be a read or a write.
0x00 = read
0x01 = write

Note: If the request message is a configuration read then the ENABLE/DISABLE and PARAMETER_LIST fields should not be included, and the data length will be 1 byte.

ENABLE/DISABLE
This field in 1 byte that indicates if the parameters specified in PARAMETER_LIST should be enabled or disabled. The state of the parameters not specified in the PARAMETER_LIST field will not change.
0x00 = Enable specified parameters
0x01 = Disable specified parameters

This field will not be present in a response to a read enabled parameters command.

PARAMETER_LIST
Variable length data field containing one byte for each parameter to be enabled. Each byte corresponds to a parameter ID. This should not be included in the request message if reading the ENABLE_PARAMETERS data.

If one byte value 0xFF is used for PARAMETER_LIST then all the supported parameters will be either enabled or disabled according to the ENABLE/DISABLE byte.

NOT_SUPP Variable length field containing one byte for each parameter that was requested to be enabled/disabled, but is not supported by the vehicle. This may be a zero length field.

CS Summation checksum

Example 1: Enable Parameters
TX: 0x01 0x01 0x21 0x04 0x01 0x00 0x02 0x03 0x2D

RX: 0x01 0x01 0xA1 0x02 0x01 0x00 0xA6

The command sent requests Throttle Position and Odometer to be enabled. All parameters that were requested to be enabled are supported by the vehicle.

Example 2: Enable Parameters
TX: 0x01 0x01 0x21 0x04 0x01 0x00 0x02 0x03 0x2D

RX: 0x01 0x01 0xA1 0x03 0x01 0x00 0x02 0xA9

The command sent requests Throttle Position and Odometer to be enabled. All parameters that were requested to be enabled are supported by the vehicle except for Throttle Position.

Example 3: Read Enabled Parameters
TX: 0x01 0x01 0x21 0x01 0x00 0x24

RX: 0x01 0x01 0xA1 0x03 0x00 0x00 0x01 0xA7

The command sent requests a list of all enabled parameters. Vehicle Speed and Engine Speed are enabled and listed in the response.

Example 4: Disable Parameters
TX: 0x01 0x01 0x21 0x04 0x01 0x01 0x02 0x03 0x2E

RX: 0x01 0x01 0xA1 0x03 0x01 0x01 0x02 0xAA

The command sent requests that Throttle position and Odometer be disabled. The response indicates that all the requested parameters except Throttle position were supported and have been disabled.

Note: Engine Speed and Vehicle Speed cannot be disabled.

0x22 – GET_PARAMETER

Description:
This command will return the current value for the requested parameter. Since the most current parameter's results are stored in the OBDII Streamer the response delay will be very short.

Command to Send:
0x01 0x01 0x22 DL PARAMETER_ID CS

Response:
0x01 0x01 0xA2 DL PARAMETER_DATA CS

Field Description:

DL Data Length Byte

PARAMETER_ID Variable length data field containing one byte for each parameter requested. Each byte corresponds to a parameter ID.

Multiple parameters can be specified in the PARAMETER_ID field, or 0xFF can be specified to request all enabled parameters.

A maximum of 11 parameters can be specified in one message. If 0xFF is sent to request all enabled parameters then the response may be broken into multiple responses depending on the number of enabled parameters.

PARAMETER_DATA
The PARAMETER_DATA field is a variable length data field that contains both the requested parameter(s) identifier and the parameter data. All parameter identifiers are one byte long, and parameter data is variable length. The format of the data field is a repeating pattern of *parameter identifier1, Data1*, *parameter identifier2, Data2*,.. etc.

CS Summation checksum

Example:
TX: 0x01 0x01 0x22 0x03 0x01 0x02 0x03 0x2D

RX: 0x01 0x01 0xA2 0x09 0x01 0x01 0x23 0x02 0x98 0x76 0x03 0x55 0x66 0x77 0x88 0x9F

The breakdown of the response is:

Parameter 0x01: Data = 0x0123
Parameter 0x02: Data = 0x9876