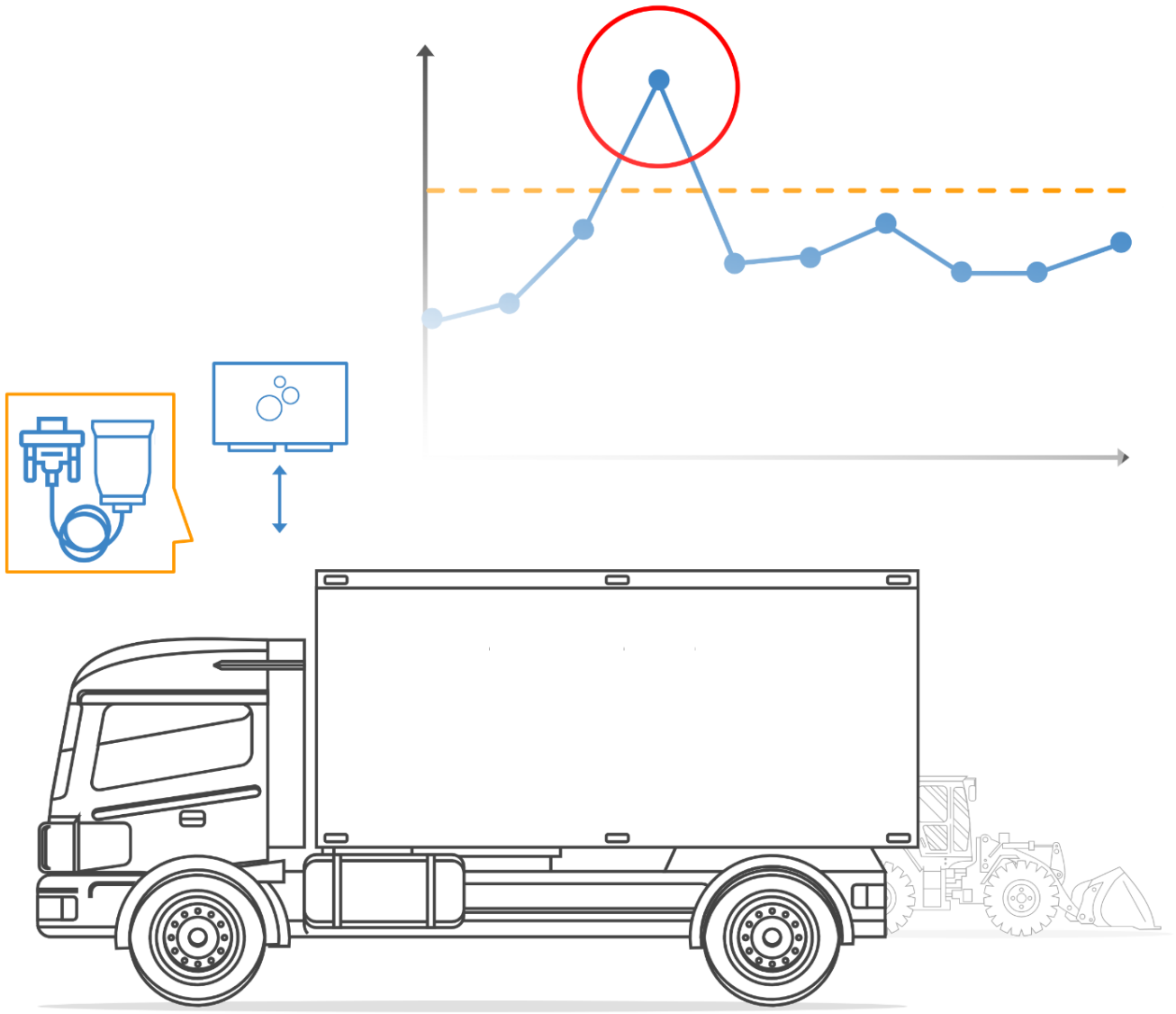
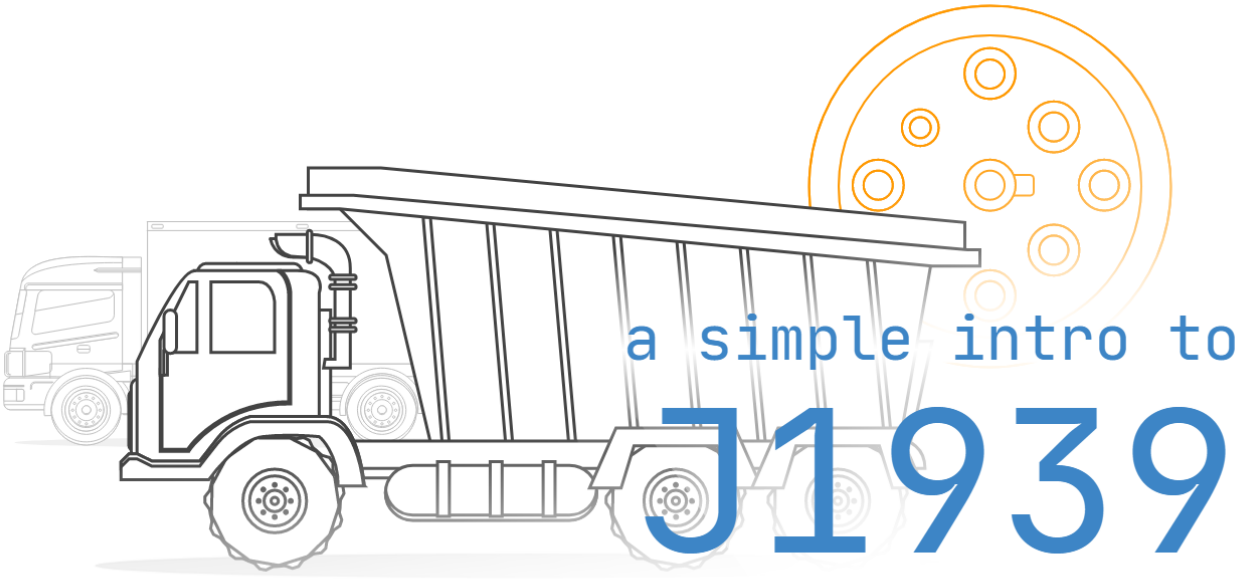


J1939 data

starter pack

CSS Electronics | www.csselectronics.com | contact@csselectronics.com
Last modified: April 20, 2022

Table of Contents



J1939 Explained - A Simple Intro

In this guide we introduce the J1939 protocol basics incl. PGNs and SPNs. This is a practical intro so you will also learn how to decode J1939 data via DBC files, how J1939 logging works, key use cases and practical tips.

What is J1939?

In short, SAE J1939 is a set of standards that define how ECUs communicate via the CAN bus in heavy-duty vehicles.
As explained in our CAN bus intro, most vehicles today use the Controller Area Network (CAN) for ECU communication.
However, CAN bus only provides a "basis" for communication (like a telephone) - not a "language" for conversation.

In most heavy-duty vehicles, this language is the SAE J1939 standard defined by the Society of Automotive Engineers (SAE). In more technical terms, J1939 provides a higher layer protocol (HLP) based on CAN as the "physical layer".
What does that mean, though?

One standard across heavy-duty vehicles

In simple terms, J1939 offers a standardized method for communication across ECUs, or in other words:
J1939 provides a common language across manufacturers.
In contrast, e.g. cars use proprietary OEM specific protocols.

J1939 application examples

Heavy-duty vehicles (e.g. trucks and buses) is one of the most well-known applications. However, several other key industries leverage SAE J1939 today either directly or via derived standards (e.g. ISO 11783, MilCAN, NMEA 2000, FMS):

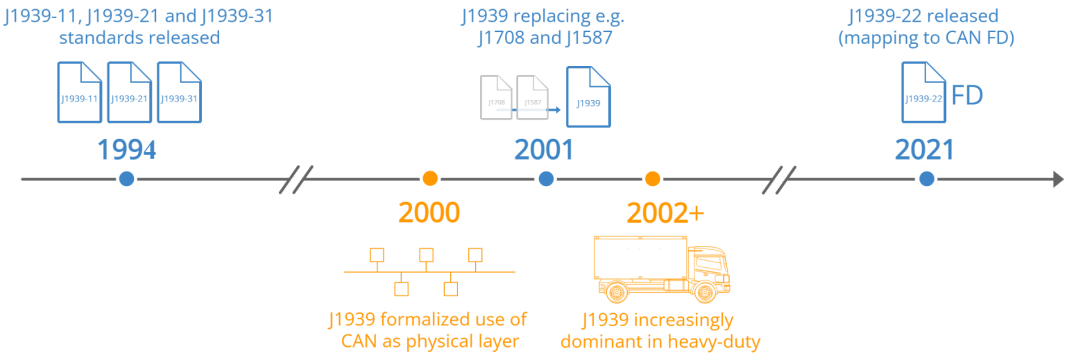
- Foresting machines (e.g. delimbers, forwarders, skidders)
- Mining vehicles (e.g. bulldozers, draglines, excavators, ...)
- Military vehicles (e.g. tanks, transport vehicles, ...)
- Agriculture (e.g. tractors, harvesters, ...)

- Construction (e.g. mobile hydraulics, cranes, ...)
- Fire & Rescue (e.g. ambulances, fire trucks, ...)
- Other (e.g. ships, pumping, e-buses, power generation, ...)

J1939 history & future

History

- 1994: First docs were released (J1939-11, J1939-21, J1939-31)
- 2000: The initial top level document was published
- 2000: CAN formally included as part of J1939 standard
- 2001: J1939 starts replacing former standards SAE J1708/J1587



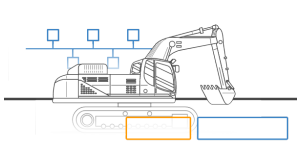
Future

With the rise of heavy-duty telematics, J1939 will increasingly play a role in the market for connected vehicles. In turn, this will increase the need for secure J1939 IoT loggers. In parallel, OEMs will increasingly shift from Classical CAN to CAN FD as part of the transition to J1939 with flexible data-rate. In turn, this will increase the need for J1939 FD data loggers.

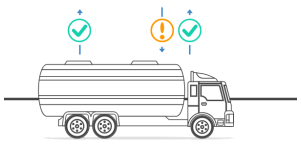
"The market for in-vehicle connectivity - the hardware and services bringing all kinds of new functionality to drivers and fleet owners - is expected to reach EUR 120 billion by 2020."
- Boston Consulting Group, Connected Vehicles and the Road to Revenue

4 key characteristics of J1939

The J1939 protocol has a set of defining characteristics outlined below:

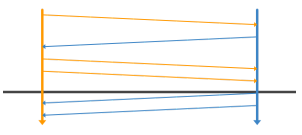


250K baud rate & 29-bit extended ID
The J1939 baud rate is typically 250K (though recently with support for 500K) - and the identifier is extended 29-bit (CAN 2.0B)



Broadcast + on-request data
Most J1939 messages are broadcast on the CAN-bus, though some data is only available by requesting the data via the CAN bus

PGN identifiers & SPN parameters
J1939 messages are identified by 18-bit Parameter Group Numbers (PGN), while J1939 signals are called Suspect Parameter Numbers (SPN)



Multibyte variables & Multi-packets
Multibyte variables are sent least significant byte first (Intel byte order). PGNs with up to 1785 bytes are supported via J1939 transport protocol

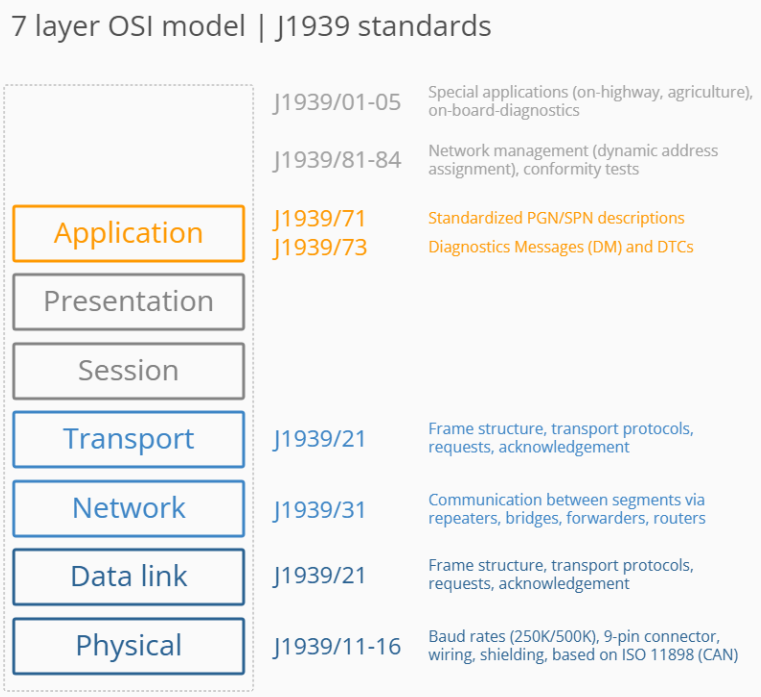
Additional J1939 characteristics

Below are a set of additional characteristics of the J1939 protocol:

- Reserved: J1939 includes a large range of standard PGNs, though PGNs 00FF00 through 00FFFF are reserved for proprietary use
- Special Values: A data byte of 0xFF (255) reflects N/A data, while 0xFE (254) reflects an error
- J1939 address claim: The SAE J1939 standard defines a procedure for assigning source addresses to J1939 ECUs after network initialization via an 8-bit address in a dynamic way

Technical: J1939 'higher layer protocol' explained

J1939 is based on CAN, which provides the basic "physical layer" and "data link layer", the lowest layers in the OSI model. Basically, CAN allows the communication of small packets on the CAN bus, but not a lot more than that. Here, J1939 serves as a higher layer protocol on top, enabling more complex communication.



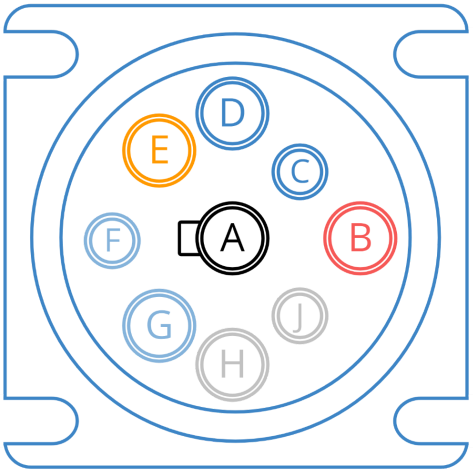
A higher layer protocol enables communication across the large complex networks of e.g. vehicle manufacturers.

For example, the SAE J1939 protocol specifies how to handle "multi-packet messages", i.e. when data larger than 8 bytes needs to be transferred. Similarly, it specifies how data is to be converted into human-readable data.
It does so by providing a family of standards. For example, J1939-71 is a document detailing the information required to convert a large set of cross-manufacturer standardized J1939 messages into human-readable data (more on this below). Many other CAN based higher layer protocols exist, e.g. CANopen, DeviceNet, Unified Diagnostic Services. These typically offer some level of standardization within their respective industries - though all of them can be extended by manufacturers.

In comparison, the aforementioned passenger cars have unique standards per manufacturer. In other words, you can use the same J1939 database file to convert e.g. engine speed across two trucks from different manufacturers - but you cannot e.g. read the data from an Audi A4 using the same IDs & scaling parameters as for a Peugeot 207.

The J1939 connector (9-pin)

The J1939-13 standard specifies the 'off-board diagnostic connector' - also known as the J1939 connector or 9-pin deutsch connector. This is a standardized method for interfacing with the J1939 network of most heavy duty vehicles - see the illustration for the J1939 connector pinout.



A	Ground
B	Battery power
C	CAN 1 H
D	CAN 1 L
E	CAN shield
F	J1708 (+) / CAN 2 H
G	J1708 (-) / CAN 2 L
H	OEM specific
J	OEM specific

Type 1 (black): CAN 1 = 250K
Type 2 (green): CAN 1 = 500K

Black type 1 vs green type 2

Note that the J1939 deutsch connector comes in two variants: The original black connector (type 1) and the newer green connector (type 2), which started getting rolled out around 2013-14.

J1939 type 2 female connectors are physically backwards compatible, while type 1 female connectors only fit type 1 male sockets. The type 2 connector was designed for the SAE J1939-14 standard, which adds support for 500K bit rates. The purpose of "blocking" type 1 connectors is to ensure that older hardware (presumably using 250K bit rates) is not connected to type 2 500K bit rate J1939 networks. Specifically, the physical block is through a smaller hole for pin F in the type 2 male connectors. See also the example of a DB9-J1939 adapter cable (type 2).

Multiple J1939 networks

As evident, the J1939 deutsch connector provides access to the J1939 network through pins C (CAN high) and D (CAN low). This makes it easy to interface with the J1939 network across most heavy duty vehicles.

In some cases, however, you may also be able to access a secondary J1939 network through pins F and G or pins H and J (with H being CAN H and J being CAN L).

Many of today's heavy duty vehicles have 2 or more parallel CAN bus networks and in some cases at least two of these will be available through the same J1939 connector. This also means that you will not necessarily have gained access to all the available J1939 data if you've only attempted to interface through the 'standard' pins C and D.

Other heavy duty connectors

While the J1939 deutsch connector is the most common way to interface with the J1939 network of heavy duty vehicles, other connectors of course exist. Below are some examples:

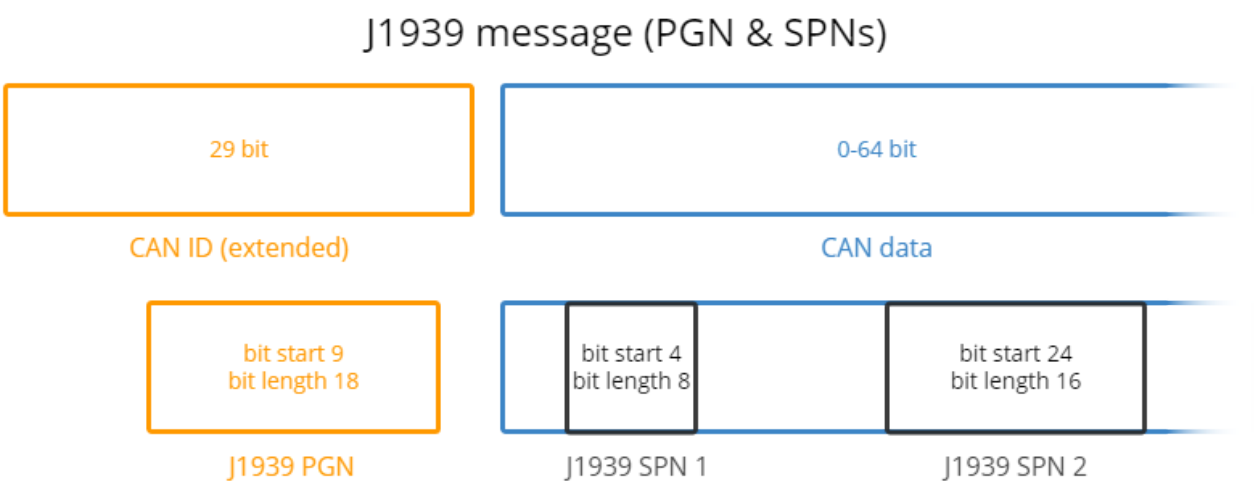
- J1939 Backbone Connector: This 3-pin deutsch connector provides pins for CAN H/L a CAN shield (no power/ground)
- CAT connector: The Caterpillar industrial connector is a grey 9-pin deutsch connector. However, the pin-out differs from the J1939 connector (A: Power, B: Ground, F: CAN L, G: CAN H) and the connector physically blocks access from standard type 1 and 2 J1939 connectors
- OBD2 type B connector: The type B OBD2 connector (SAE J1962) in heavy duty vehicles sometimes provide direct access to the J1939 network through pins 6 and 14
- Volvo 2013 OBD2 connector: This variant matches the type B OBD2 connector, but also adds access to the J1939 high via pin 3 and J1939 low via pin 11

The J1939 PGN and SPN

In the following section we explain the J1939 PGNs and SPNs.

Parameter Group Number (PGN)

The J1939 PGN comprises an 18-bit subset of the 29-bit extended CAN ID. In simple terms, the PGN serves as a unique frame identifier within the J1939 standard. For example, you can look this up in the J1939-71 standard documentation, which lists PGNs/SPNs.



Example: J1939 PGN 61444 (EEC1)

Assume you recorded a J1939 message with HEX ID 0CF00401. Here, the PGN starts at bit 9, with length 18 (indexed from 1). The resulting PGN is 0F004 or in decimal 61444. Looking this up in the SAE J1939-71 documentation, you will find that it is the "Electronic Engine Controller 1 - EEC1". Further, the document will have details on the PGN including priority, transmission rate and a list of the associated SPNs - cf. the illustration. For this PGN, there are seven SPNs (e.g. Engine Speed, RPM), each of which can be looked up in the J1939-71 documentation for further details.

PGN61444 - Electronic Engine Controller 1 - EEC1

Transmission Repetition Rate: Engine speed dependent
Data Length: 8 bytes
Data Page: 0
PDU Format: 240
PDU Specific: 4
Default Priority: 3
Parameter Group Number: 61444 (0x00F004)

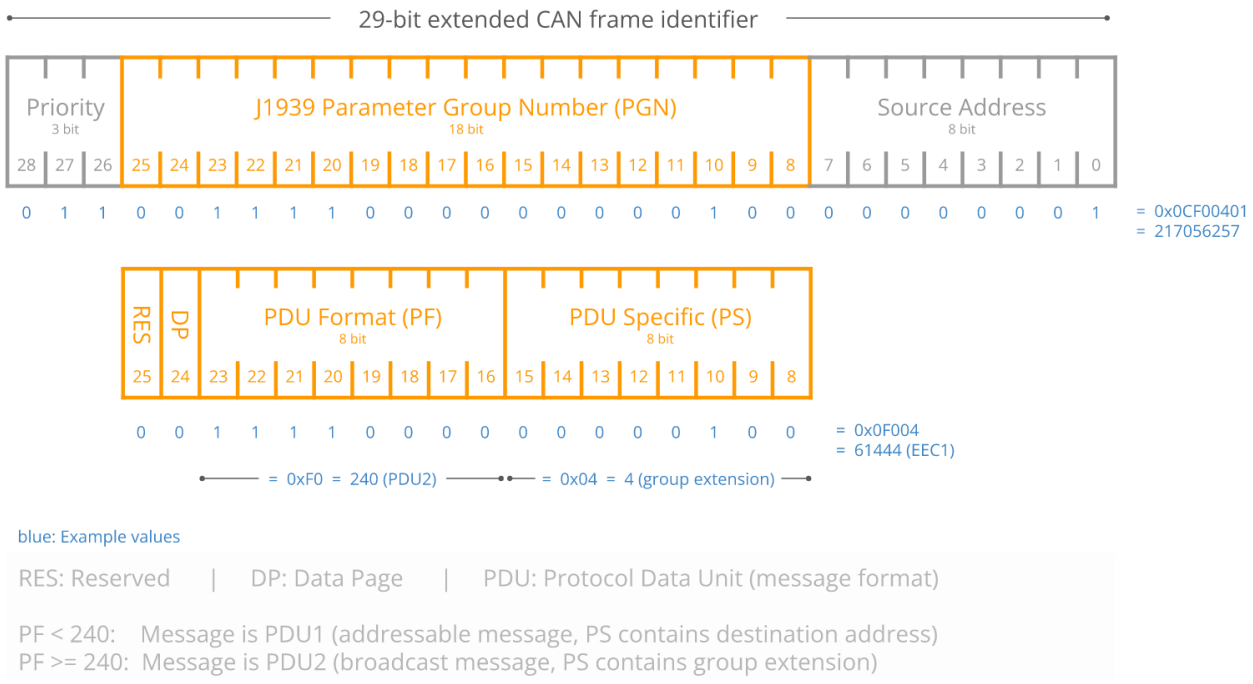
Bit Start/Byte	Length	SPN ID	SPN Description
1.1	4 bits	899	Engine Torque Mode
2	1 byte	512	Driver's Demand Engine - % Torque
3	1 byte	513	Actual Engine - Percent Torque
4-5	2 bytes	190	Engine Speed
6	1 byte	1483	SA of Controlling Device for Engine Control
7.1	4 bits	1675	Engine Starter Mode
8	1 byte	2432	Engine Demand - Percent Torque

Detailed breakdown of the J1939 PGN

Let's look at the CAN ID to PGN transition in detail. Specifically, the 29 bit CAN ID comprises the Priority (3 bits), the J1939 PGN (18 bits) and the Source Address (8 bits). In turn, the PGN can be split into the Reserved Bit (1 bit), Data Page (1 bit), PDU format (8 bit) and PDU Specific (8 bit).

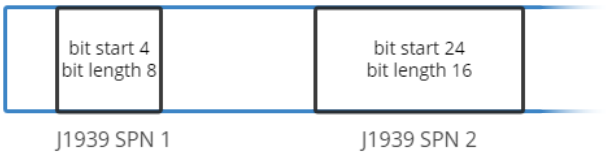
The detailed PGN illustration also includes example values for each field in binary, decimal and hexadecimal form.

To learn more about the transition from 29 bit CAN ID to 18 bit J1939 PGN, see also our online CAN ID to J1939 PGN converter. The converter also includes a full J1939 PGN list for PGNs included in our J1939 DBC file.



Suspect Parameter Number (SPN)

The J1939 SPN serves as the identifier for the CAN signals (parameters) contained in the databytes. SPNs are grouped by PGNs and can be described in terms of their bit start position, bit length, scale, offset and unit - information required to extract and scale the SPN data to physical values.



Example: Extracting J1939 SPN 190 (Engine Speed)

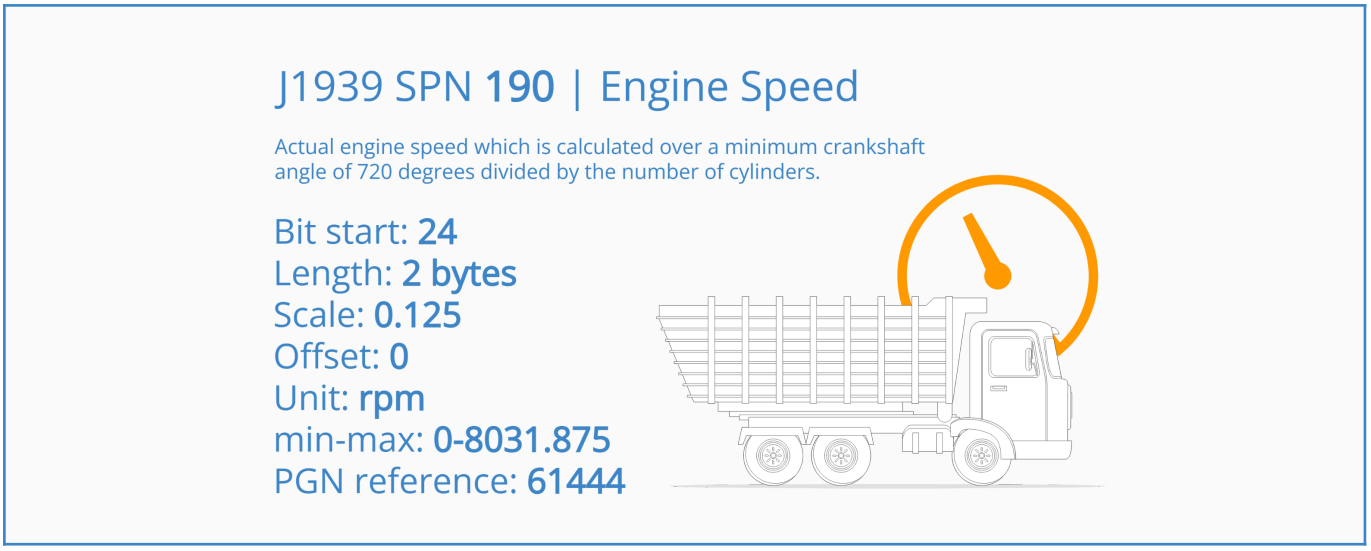
Assume you have recorded a raw J1939 frame as below:

CAN ID	Data bytes
0CF00401	FF FF FF 68 13 FF FF FF

By converting the CAN ID to the J1939 PGN you identify that this is the PGN 61444 from before. From the J1939-71 document, you observe that one of the SPNs within this PGN is Engine Speed (SPN 190) with details as in the illustration below.

Using these details, it is possible to extract the Engine Speed physical value data e.g. for plot purposes. To do so, note from the SPN info that the relevant data is in bytes 4 and 5, i.e. the HEX data bytes 68 and 13. Taking the decimal form of the HEX value 1368 (Intel byte order), we get 4968 in decimal. To arrive at the RPM, we conduct a scaling of this value using the offset 0 and the scale 0.125 RPM/bit. The physical value (aka scaled engineering value) is 621 RPM.

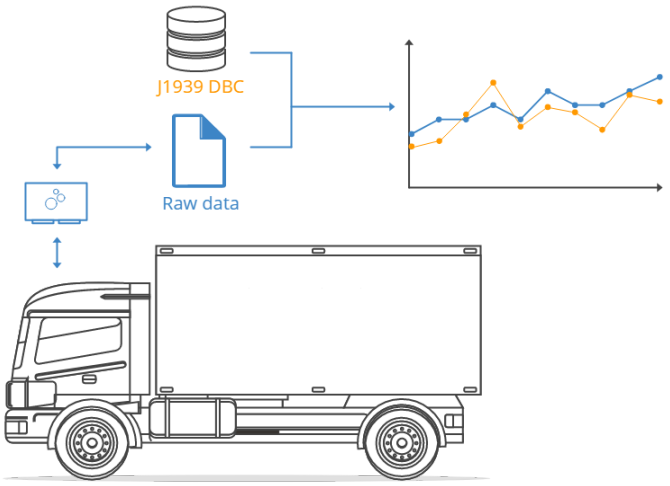
Note how some data bytes in the above are FF or 255 decimal, i.e. not available. While the PGN may theoretically support SPNs in this range, the FF padding means that this particular application does not support these parameters.



In practice, you will not 'PDF-lookup' rules for J1939 data - instead, this info is stored in a CAN database file (DBC).

Example: J1939 DBC file

A J1939 DBC file can be used to decode data across most heavy-duty vehicles. For example, raw J1939 data can be recorded with a CAN bus data logger and analyzed in a CAN software tool that supports DBC conversion (e.g. asammdf). This will typically result in a conversion of 40-60% of the vehicle data - with the rest being OEM specific proprietary data that requires reverse engineering.



J1939 truck sample data: Raw & physical values

Below we illustrate what real J1939 data looks like. The 'raw' J1939 data was recorded from a heavy duty truck using a CANedge2, while the 'physical values' reflect the output after decoding the raw data via the free asammdf software and the J1939 DBC.

Sample: Raw J1939 truck data (CSV)

Data from the CANedge is recorded in a standardized binary format, MDF4, which can be converted to any file format via our MDF4 converters (e.g. to CSV, ASC, TRC, ...). Below is a CSV version of the raw J1939 frames. Notice that the CAN IDs and data bytes are in hexadecimal format:

```
TimestampEpoch;BusChannel;ID;IDE;DLC;DataLength;Dir;EDL;BRS;DataBytes
1578922367.777150;1;14FEF131;1;8;8;0;0;0;CFFFFFF300FFFF30
1578922367.777750;1;10F01A01;1;8;8;0;0;0;2448FFFFFFFFFFF
1578922367.778300;1;CF00400;1;8;8;0;0;0;107D82BD1200F482
1578922367.778900;1;14FF0121;1;8;8;0;0;0;FFFFFFFFFFFFFCFF
1578922367.779500;1;18F0000F;1;8;8;0;0;0;007DFFFF0F7DFFFF
1578922367.780050;1;18FFA03D;1;8;8;0;0;0;2228240019001AFF
1578922367.780600;1;10FCFD01;1;8;8;0;0;0;FFFFFFFF1623FFFF
1578922367.781200;1;18FD9401;1;8;8;0;0;0;A835FFFFA9168F03
1578922367.781750;1;18FDA101;1;8;8;0;0;0;1224FFFFFFFF00FF
1578922367.782350;1;18F00E3D;1;8;8;0;0;0;741DFFFFFFFFFFF
1578922367.782950;1;18F00F3D;1;8;8;0;0;0;B40FFFFFFFFFFFF
1578922367.783500;1;10FDA301;1;8;8;0;0;0;FFFFFFFFFFFFFFFF
...
```

You can optionally download full raw J1939 MDF4 samples from the CANedge2 in our intro docs. The sample data also includes a demo J1939 DBC so that you can replicate the conversion steps via asammdf.

Sample: Decoded physical values J1939 truck data (CSV)

Once the raw J1939 data is decoded and exported, the result is timeseries data with parameters like oil temperature, engine speed, GPS, fuel rate and speed:

```
timestamps,ActualEnginePercentTorque,EngineSpeed,EngineCoolantTemperature,EngineOilTemperature1,EngineFu
elRate,EngineTotalIdleHours,FuelLevel1,Latitude,Longitude,WheelBasedVehicleSpeed
2020-01-13 16:00:13.259449959+01:00,0,1520.13,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.268850088+01:00,0,1522.88,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.270649910+01:00,0,1523.34,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.271549940+01:00,0,1523.58,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.278949976+01:00,0,1525.5,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.289050102+01:00,0,1527.88,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.299000025+01:00,0,1528.13,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.308300018+01:00,0,1526.86,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.309099913+01:00,0,1526.75,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.317199945+01:00,0,1526.45,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
...
```

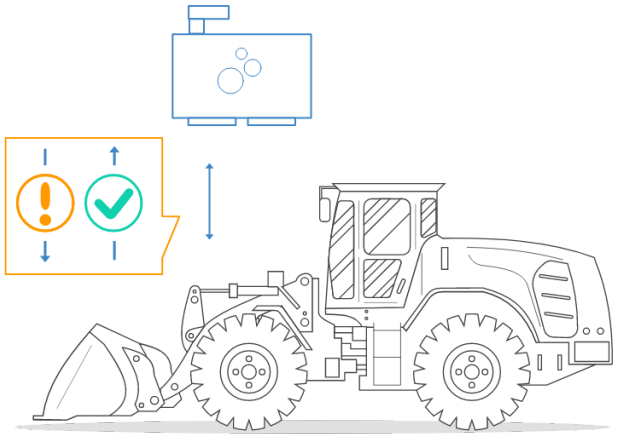
For more on logging J1939 data, see our J1939 data logger and mining telematics articles.

About the CANedge J1939 logger

The CANedge lets you easily record J1939 data to an 8-32 GB SD card. Simply connect it to e.g. a truck to start logging - and decode the data via free software/APIs and our J1939 DBC. Learn more.

J1939 request messages

Most J1939 messages are broadcast via the CAN bus, but some are only sent "on-request" (e.g. when polled by a J1939 data logger). On-request data often includes J1939 diagnostic trouble codes (DTCs), making it important in vehicle diagnostics. Below we briefly outline how it works:



Sending J1939 request messages

To send a J1939 request via the CAN bus, a special 'request message' is used (PGN 59904), which is the only J1939 message with only 3 bytes of data. It has priority 6, a variable transmit rate and can either be sent as a global or specific address request. The data bytes 1-3 should contain the requested PGN (Intel byte order). Examples of requested J1939 messages include the diagnostic messages (e.g. J1939 DM2).

A CAN bus data logger like the CANedge can be set up to send J1939 request messages - see e.g. our CANedge Intro for a detailed step-by-step guide.

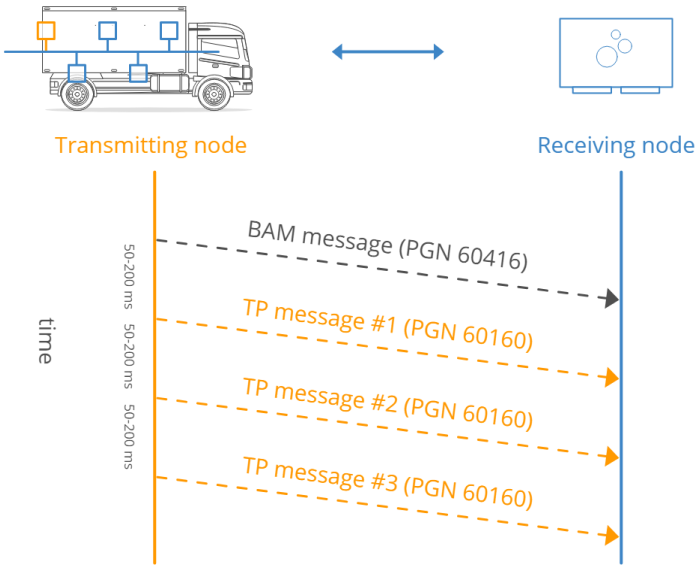
J1939 code requests vs warranty compliance
Sending request messages is typically key to requesting J1939 codes and thus J1939 diagnostics. One challenge, however, is that J1939 loggers are often required to have a contactless connection to the J1939 data link, meaning that they're unable to interact with the CAN bus and transmit J1939 request frames. The restriction is often related to warranty compliance as some vehicle manufacturers do not allow direct access by 3rd party devices via the J1939 connector.

In some cases, it is required that the J1939 analyzer is "physically" contactless, e.g. via a CANcrocodile adapter. In other cases, it is sufficient that the J1939 logger operates in "configurable" silent mode. The latter makes it easier to perform ad hoc requests for J1939 fault codes, either via a manual configuration update or via an over-the-air update for the CANedge2.

J1939 transport protocol (TP)

The previous PGN and SPN examples are based on J1939 messages with 8 data bytes. While these are most common, J1939 multi-frame messages also exist with >8 data bytes - sent via the J1939 transport protocol.

Below we outline how the J1939 transport protocol works, a practical J1939 TP data example and how to decode multi-frame J1939 messages via DBC files:



How does the J1939 transport protocol work?

The J1939 protocol specifies how to deconstruct, transfer and reassemble packets across multiple frames - a process referred to as the J1939 Transport Protocol (see J1939-21). Two types of the J1939 TP exist:

1. The Connection Mode (intended for a specific device)
2. The BAM (Broadcast Announce Message) which is intended for the entire network

For example, a transmitting ECU may send an initial BAM packet to set up a data transfer. The BAM specifies the PGN identifier for the multi-packet message as well as the number of data bytes and packets to be sent. It is then followed by up to 255 packets/frames of data. Each of the 255 packets use the first data byte to specify the sequence number (1 up to 255), followed by 7 bytes of data. The max number of bytes per multi-packet message is therefore 7 bytes x 255 = 1785 bytes.

The final packet contains at least one byte of data, followed by unused bytes set to FF. In the BAM type scenario, the time between messages is 50-200 ms. In post processing, a conversion software tool can reassemble the multiple entries of 7 data bytes into a single payload and handle it according to the multi-packet PGN and SPN specifications as found in e.g. a J1939 DBC file.

A practical J1939 transport protocol example

Decoding J1939 multiframe data is more complex than decoding standard J1939 frames. To understand why, consider the below example of a J1939 transport protocol response, recorded with the CANedge2:

```
TimestampEpoch;BusChannel;ID;IDE;DLC;DataLength;Dir;EDL;BRS;DataBytes
1605078459.438750;1;1CECFF00;1;8;8;0;0;0;20270006FFE3FE00
1605078459.498750;1;1CEBFF00;1;8;8;0;0;0;013011B2A041B240
1605078459.559750;1;1CEBFF00;1;8;8;0;0;0;021FB2102CB2603B
1605078459.618750;1;1CEBFF00;1;8;8;0;0;0;03B230430000D309
1605078459.678750;1;1CEBFF00;1;8;8;0;0;0;04C0441E37967DE1
1605078459.738750;1;1CEBFF00;1;8;8;0;0;0;05E02E7B02FFFF80
1605078459.799850;1;1CEBFF00;1;8;8;0;0;0;06E0FFFFFFFFFFFF
```

The above sequence consists of two J1939 message types:

PGN 60416 - J1939 Transport Protocol BAM (Connection Management)

Data Length: 8 bytes
Default Priority: 7
Parameter Group Number: 60416 (0xEC00)

Byte	Description
1	Fixed at 32
2-3	Message size in bytes
4	Number of packets
5	Reserved (filled with FF)
6-8	PGN

PGN 60160 - J1939 Transport Protocol (Data Transfer)

Data Length: 8 bytes
Default Priority: 7
Parameter Group Number: 60160 (0xEB00)

Byte	Description
1	Sequence number (1 to 255)
2-8	Data payload (unused locations in the last frame are set to FF)

A J1939 BAM message with ID 1CECFF00 (PGN 60416 or EC00), which contains the response data length and J1939 PGN - and a J1939 data transfer messages with ID 1CEBFF00 (PGN 60160 or EB00). These contain the payload across multiple frames.

Below we break down the J1939 transport protocol example with focus on the data byte interpretation:

Timestamp (Epoch)	CAN ID	PGN (HEX)	PGN (DEC)	DataBytes
1605078459.438750	1CECFF00	EC00	60416	2027000 6FFE3FE00
1605078459.498750	1CEBFF00	EB00	60160	013011B2A041B240
1605078459.559750	1CEBFF00	EB00	60160	021FB2102CB2603B
1605078459.618750	1CEBFF00	EB00	60160	03B230430000D309
1605078459.678750	1CEBFF00	EB00	60160	04C0441E37967DE1
1605078459.738750	1CEBFF00	EB00	60160	05E02E7B02FFFF80
1605078459.799850	1CEBFF00	EB00	60160	06E0FFFFFFFFFFFF
1605078459.438750	18FEE3FE	FEE3	65251	3011B2A041B240 ... E0FFFFFF

legend
control byte (0x20=BAM)
#data bytes (0x0027=39)
#packets (0x06=6)
reserved
J1939 PGN (0x00FEE3=65251)
sequence numbers
payload data (39 bytes)
unused (3 bytes)

Generally, a J1939 transport protocol response sequence can be processed as follows:

- Identify the BAM frame, indicating a new sequence being initiated (via the PGN 60416)
- Extract the J1939 PGN from bytes 6-8 of the BAM payload to use as the identifier of the new frame
- Construct the new data payload by concatenating bytes 2-8 of the data transfer frames (i.e. excl. the 1st byte)

Above, the last 3 bytes of the BAM equal E3FE00. When reordered, these equal the PGN FEE3 aka Engine Configuration 1 (EC1). Further, the payload is found by combining the the first 39 bytes across the 6 data transfer packets/frames.
Note: The last 3 data payload bytes in this practical example happen to be FF, yet we still include these in the payload as the BAM message specifies the data length to be 39. The final 3 FF bytes in the 6th packet are unused.

How to decode a J1939 transport protocol message

With the method explained above, we have created a 'constructed' J1939 data frame with a data length exceeding 8 bytes. This frame can be decoded using a J1939 DBC file, just like a regular J1939 data frame. For the PGN EC1, the J1939 DBC specifies a data length of 40 with signals defined for the full payload.

As such, once the J1939 software/API has reconstructed the multiframe response into a single J1939 frame, the DBC decoding can be done as usual. One minor tweak is that most J1939 DBC files expects that the raw log file of J1939 data will contain 29-bit CAN IDs (not 18-bit J1939 PGNs). As such, if the software embeds the reconstructed J1939 TP frame in the original raw data, it may need to convert the extracted J1939 PGN into a 29-bit CAN ID first. You can also see our J1939 google sheet, which breaks down how a J1939 PGN can be converted to a 29-bit CAN ID.

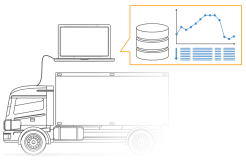
J1939 TP data & Python API decoding
The CANedge lets you request and record J1939 transport protocol data. To decode the TP data, you can either convert the raw log files to another format (like Vector ASC), or you can use our Python API. In our api-examples library on github, we provide a basic example of how J1939 transport protocol data can be reconstructed and DBC decoded, incl.

sample data. Since the CANedge Python API enables decoding of J1939 transport protocol data, J1939 signals from multiframe messages can e.g. be visualized in J1939 telematics dashboards.

Logging J1939 data - example use cases

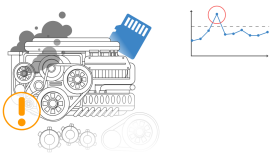
There are several common use cases for recording J1939 data:

Heavy duty fleet telematics
J1939 data from trucks, buses, tractors etc. can be used in fleet management to reduce costs or improve safety
learn more



Live stream diagnostics
By streaming decoded J1939 data to a PC, technicians can perform real-time J1939 diagnostics on vehicles
learn more

Predictive maintenance
Vehicles can be monitored via WiFi CAN loggers in the cloud to predict breakdowns based on the J1939 data
learn more



Heavy-duty vehicle blackbox
A CAN logger can serve as a 'blackbox' for heavy-duty vehicles, providing data for e.g. disputes or J1939 diagnostics
learn more

6 practical tips for J1939 data logging

Many of our end users work with J1939 logging in the field - and below we share 6 practical logging tips:

J1939 logger vs J1939 streaming interface

Standalone J1939 data loggers with SD cards are ideal for logging data from e.g. vehicle fleets over weeks or months. A WiFi J1939 logger also enables telematics use cases. In contrast, a J1939 USB-PC interface requires a PC to stream data from the CAN bus in real-time. This is e.g. useful for diagnostic purposes - or analysing physical events. The CLX000 enables both modes of operation, while the CANedge2 is perfect for telematics.

Direct adapter cable vs contactless reading

To connect your CAN analyzer to a J1939 asset (e.g. a truck) you can typically use the 9-pin J1939 connector. We offer a DB9-J1939 adapter which fits the 9-pin deutsch connector found in many heavy duty vehicles. Alternatively, you may prefer to connect your CAN logger directly to the CAN bus via e.g. a CANCrocodile. This method uses induction to record data silently without cutting any CAN wiring.

WiFi vs. cellular (3G/4G) data upload

For vehicle fleet management & telematics you will typically upload the data via either WiFi or 3G/4G. The CANedge2 lets you transfer data by connecting to a WiFi access point - which can both be a WLAN router or a 3G/4G hotspot. If you need data from a truck on-the-road, you can install the CANedge2 and use it to power a 3G/4G USB hotspot. The benefit to this is that you'll have continuous access to the device - unless it is out-of-coverage. However, in cases where

data only needs to be periodically uploaded an alternative can be to upload data via WLAN routers when the vehicles visit e.g. specific areas (garages, repair shops etc) - letting you reduce data transfer costs.

Software selection & J1939 DBC file

When logging or streaming J1939 data, software for post processing is key. In particular, the software should support DBC-based J1939 conversion to allow easy conversion to human-readable data. The free supporting softwares/APIs for our CAN loggers support this. For USB streaming, our free Wireshark plugin enables live DBC conversion. Further, we offer a digital download J1939 DBC file in collaboration with SAE.

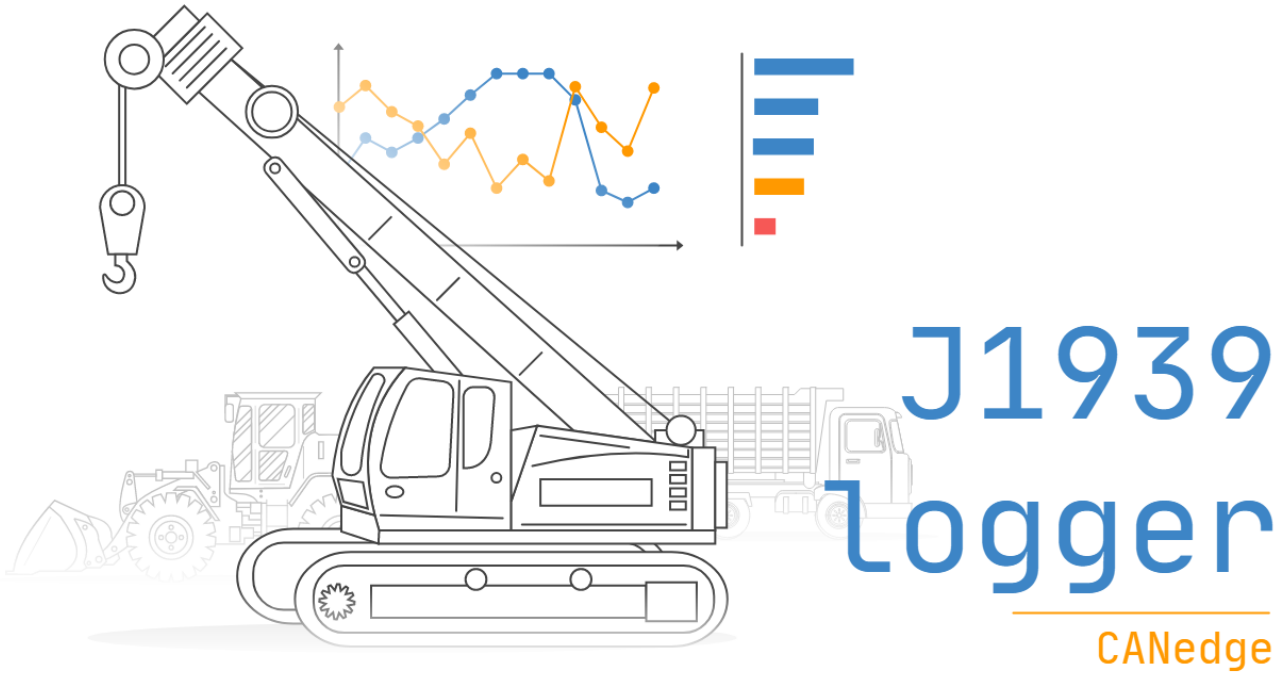
Consider the need for request PGNs

Some J1939 PGNs are only available on-request, meaning that you need to "poll" the CAN bus to log these. The CANedge and CLX000 are able to transmit custom CAN messages, which can be used to send periodic PGN requests. Note that this is not possible in "silent mode" (i.e. it is not possible if the logger is connected via e.g. a CANCrocodile).

Filter, compress and encrypt the data

To optimize your J1939 data logging, a number of advanced configurations can be helpful. In particular, the CANedge advanced filters and sampling rate options help optimize the amount of data logged - key for e.g. minimizing cellular bandwidth usage. Other options include silent mode and cyclical logging, with the latter enabling the logger to always prioritize the latest data (useful in e.g. blackbox logging).

Since J1939 is standardized, it is critical to encrypt your data 'at rest' (e.g. on an SD card) and 'in transit' (during upload). Not doing so exposes your data processing to various security risks, incl. GDPR/CCPA fines and loss of confidentiality and data integrity. For details on securing your J1939 data logging, see our intro to secure CAN logging.



J1939 Data Logger - Easy Truck Fleet Telematics

In this intro you'll learn the top 4 benefits of truck telematics - and how to get started.

You'll also see how the CANedge2 enables modern J1939 telematics - with zero monthly fees and 100% free software/APIs (incl. customizable dashboards).

What is vehicle telematics?

If you're new to vehicle telematics and fleet management, consider below basic definition:

The goal of vehicle telematics is to collect vehicle, driver & environmental data
â€" and use the data to optimize fleet operations

Vehicle telematics is commonly used in heavy-duty fleet management (trucks, buses, construction, mining, ...), prototype field testing and light trucks/service cars (delivery vans, postal services, police cars, taxis, ...). It is used by both vehicle OEMs (Original Equipment Manufacturers) and end users (e.g. construction site managers, fleet managers etc.). In some cases, the telematics system is owned by the OEM and served to an end user in the aftermarket - while in other cases, an aftermarket user may set up their own fleet management system.

What is a telematics control unit (TCU)?

A Telematics Control Unit (TCU) is a telematics device installed in a vehicle, facilitating the tracking of data from the vehicle.

Most telematics control units support the following features:

- GPS tracking: The TCU often records GPS data to track the vehicle position
- WiFi/3G/4G/GSM: The TCU needs to transfer the data to a server for processing

Today, most truck fleet management involves simple TCUs that communicate GPS data to enable route optimization and planning. However, these basic telematics systems typically do not collect data from the vehicle sensors (i.e. J1939 data). Advanced telematics control units like the CANedge2 may support the following:

- CAN bus data logging: Some TCUs support logging of CAN bus data, incl. SAE J1939 and FMS
- Edge processing: A TCU may need to perform 'edge processing' of the data (filtering, compression, triggers, ...)
- Memory: Some TCUs have memory buffers (e.g. an SD card) to buffer data in zones with no connectivity
- OTA updates: To enable fleet management at scale, over-the-air updates of firmware/configs are required
- Data requests: It may be key to enable requesting of data via CAN, e.g. to log diagnostic trouble codes (DTC)
- Modularity: Often it is necessary to add external sensors (e.g. GPS/IMU, ...) to the TCU
- Security: Modern TCUs should support secure telematics, incl. TLS data transfer, data encryption etc

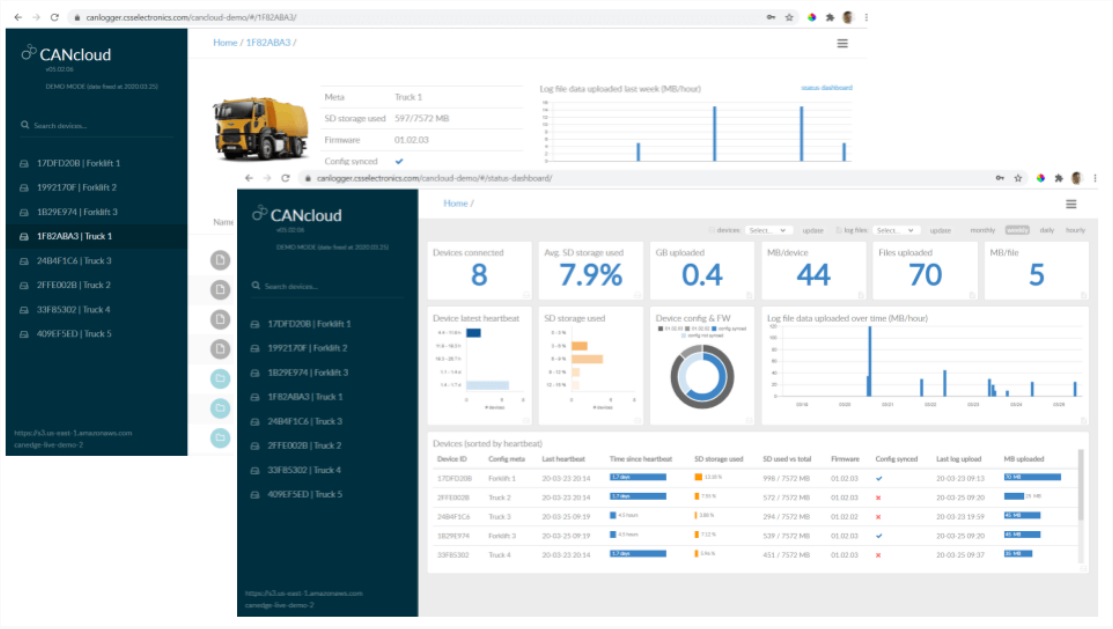
What is a telematics platform/cockpit?

If you have e.g. 200+ telematics control units deployed in the field, you'll need a way to monitor, track and manage all of them.

A telematics platform should support the following features:

- Enable users to login to the server endpoint where the telematics data is stored
- Enable easy access to the data uploaded by each telematics control unit
- Provide an interface for updating the configuration/firmware of each TCU
- Provide a status dashboard to monitor devices, SD storage%, data uploads etc

CANcloud is a 100% free browser based open source telematics platform designed for use with the CANedge2. It enables the above features, while also offering full customization (incl. easy company branding).



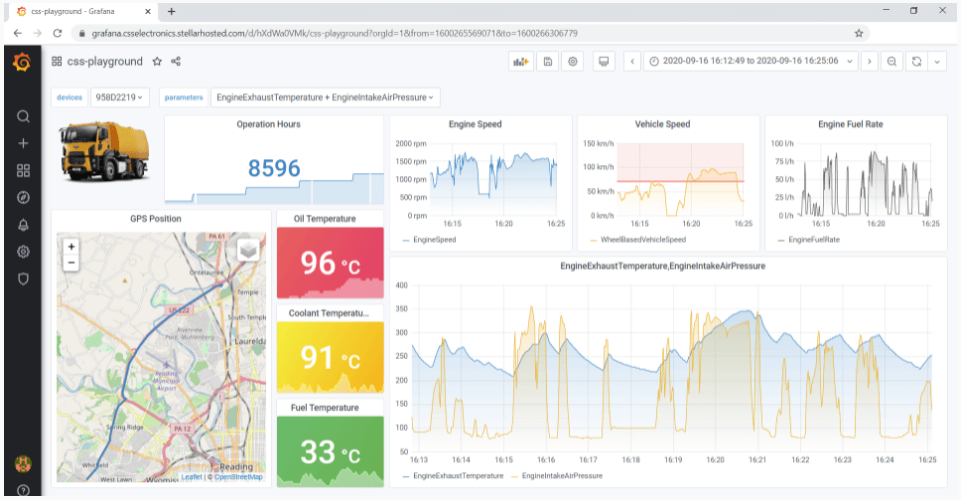
What is a telematics dashboard?

There are different end users of vehicle telematics data, including below examples:

- Fleet managers: Plan routes, reduce truck fuel costs, schedule maintenance etc.
- OEM engineers: Use high-frequency CAN/J1939 data for e.g. R&D and diagnostics
- Researchers: Collect data on e.g. driving behavior, costs, emissions etc.

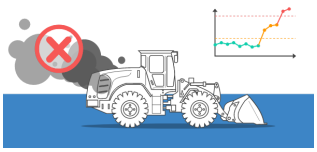
A common way to present data is through browser based telematics dashboards. Here, the data is typically processed on a backend server and pushed to a database. As an example, the CANedge2 data can be easily integrated with the open source Grafana dashboard tool. This enables 100% free and customizable telematics dashboards to visualize e.g. truck fleet telematics data.

For more in-depth data analysis (e.g. diagnostics), specialized CAN software/APIs may be required. Here, the CANedge2 enables easy analysis via free software like the asammdf GUI - or via popular tools like Vector CANalyzer and PCAN Explorer using simple data converters. Further, the Python API enables large-scale automated data processing (incl. e.g. CAN bus DBC decoding via J1939 DBC files).

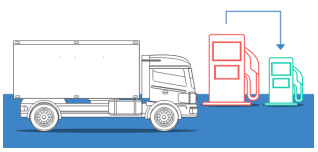


Top 4 benefits of J1939 fleet telematics

Truck telematics & vehicle fleet management is increasingly used by e.g. heavy-duty vehicle OEMs - below we list the top benefits.

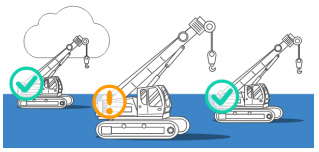


Fewer breakdowns & better diagnostics
J1939 data can help minimize breakdowns and downtime via predictive maintenance - as well as diagnose rare issues, on-site or remotely



Reduced fuel & maintenance costs
Fuel cost can comprise up to 50% of heavy duty fleet costs. Here, J1939 data can be used in optimizing routes, idling, driver behavior and fuel theft

Simpler compliance & dispute handling
By auto-collecting your vehicle data you'll simplify compliance (e.g. emissions, ELD/HOS). For OEMs, this also enables data-based dispute processing



Faster time-to-market
Real field usage data is extremely valuable to OEM development teams - e.g. for equipment debugging or fleet prototype testing

Why use the CANedge J1939 logger?

The CANedge CAN bus data logger offers optional GPS/IMU, WiFi and/or 3G/4G - ideal for J1939 fleet telematics:

PLUG & PLAY
Log data out-the-box. Standalone. Link your vehicle to your server in <2 minutes

PRO SPECS
Extractable 8-32 GB SD. 2xCAN/LIN. CAN FD. Zero data loss. 50 Âµs RTC. Error frames. MF4

SO SMALL
Only 8 x 5 x 2 CM. 100G. Robust alu enclosure. 5+ LEDs. Configurable 5V power out (CH2)

WIFI/LTE
Push data via WiFi or 3G/4G to your server. E2E security. OTA updates

GNSS + 3D IMU
Built-in GPS/IMU. 3x accuracy via sensor fusion. Position, speed, distance & more

OPEN SOURCE
Free open source software/APIs. MF4 to ASC/CSV. DBC support. Python. Dashboards

[Check out the 5 min CANedge intro video]

How does vehicle telematics work with the CANedge?

With the CANedge, vehicle telematics involves below steps:

#1 Setup your server: Set up your local/dedicated/cloud server via our guide in <5 min

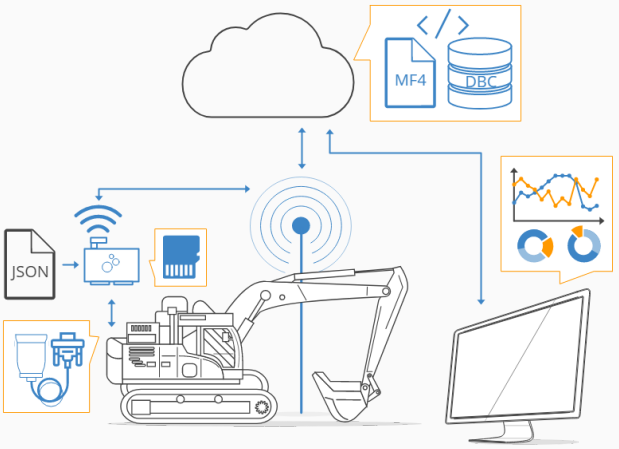
#2 Configure: Configure your device with your server and WiFi/SIM details and e.g. filters, triggers or transmit lists

#3 Record: Connect the device to your vehicle J1939 connector (e.g. using a DB9-J1939 adapter) to log raw J1939 data

#4 Transfer: The device connects to your WiFi access point(s) or 3G/4G to auto-push log files to your server

#5 Process: Use the free software/APIs to process your data - e.g. converting it to other formats (ASC, TRC, CSV, ...) or DBC decoding it to physical values via asammdf or the Python API

#6 Visualize: Finally, you can also visualize your data and set alerts via free, customizable browser dashboards



What makes the CANedge2 different?

Traditional telematics may be a good fit for some - but it comes with a number of downsides:

- Low specs: Most telematics dongles do not enable the type of pro spec logging required by vehicle OEMs
- High costs: You pay a monthly fee which quickly adds up - incl. potential vendor lock-ins
- Complex: One-size-fits-all platforms typically entail significant complexity - and may not fit your use case
- No data ownership: Your sensitive data is sent to the telematics provider's server, not your own
- Poor security: Many telematics solutions struggle with security issues, leaks and GDPR/CCPA challenges

The CANedge2 has been designed to offer a different solution:

- Pro specs: The CANedge2 is a unique combo of pro specs CAN logging - with the convenience of telematics
- Zero fees: You only pay for the hardware - all software/APIs are 100% free and open source
- Simple-to-use: The device is simple-to-use and lets you start small, gradually scaling to more advanced setups
- Zero lock-in: Data is uploaded to your own server via your WiFi access points or 3G/4G - with no vendor lock-in
- Interoperable: data & config files use standard formats (MDF4, JSON), easing integration
- Secure: The device can encrypt data & passwords on the SD (e.g. for GDPR compliance) - and upload via HTTPS

WiFi telematics vs 3G/4G cellular transfer

The CANedge2 is a WiFi CAN logger, with a number of benefits:

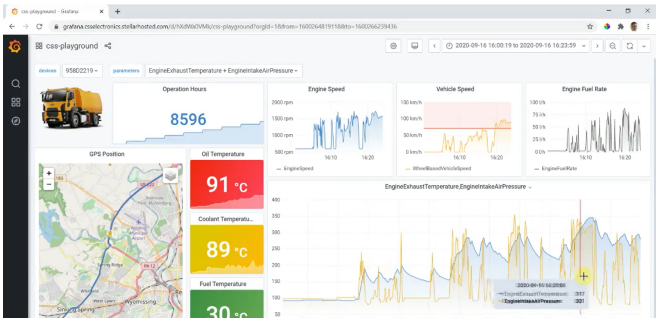
Save costs	**Existing 3G/4G**	**Mobile assets**	**Local server**
The CANedge2 can log data to an SD card for	In e.g. transit buses there are often existing WiFi	In many use cases, however, on WiFi is	In some use cases, you may want to transfer data

extended periods without WiFi access. Once it returns to a garage with a WiFi router, it can automatically upload the data (and free up the SD) - without 3G/4G transfer costs	access points available, which the CANedge2 can utilize - thus avoiding the need for a separate SIM card. This is also often the case in truck fleet telematics	available - and here the CANedge3 is the best solution. Simply insert your own SIM card and the device will push data to your server automatically	to a local server. The CANedge2 can connect to a local WiFi router and push data to a local server, e.g. on a PC or Pi. This is often used for warehouse and mining telematics

How to add GPS and 3D inertial data?

Some J1939 applications have built-in GPS data that is communicated via the CAN bus - and which can be decoded using a standard J1939 DBC. However, other vehicles do not broadcast this information by default. If you need to add GPS and/or 3D inertial sensor data to your J1939 telematics data, you can use our CANmod.gps GPS-to-CAN module with a 3D IMU. This module can be used as a plug & play add-on for the CANedge.

Software example: J1939 telematics dashboards



With the CANedge, you can easily set up free, custom browser dashboards for visualizing your J1939 data and setting alerts.

Check out the online playground - or learn more in our intro!

Playground Learn more Other software

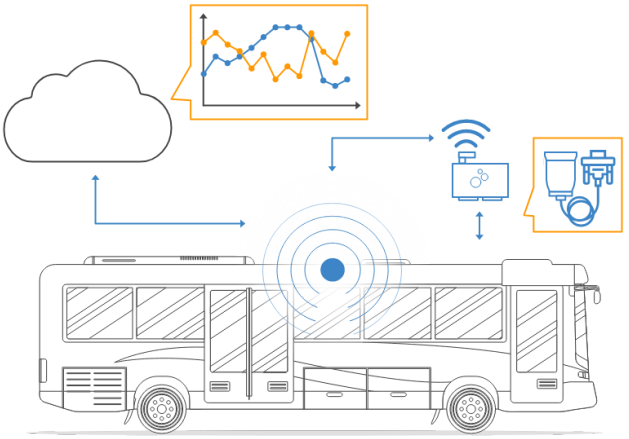
J1939 telematics use case examples

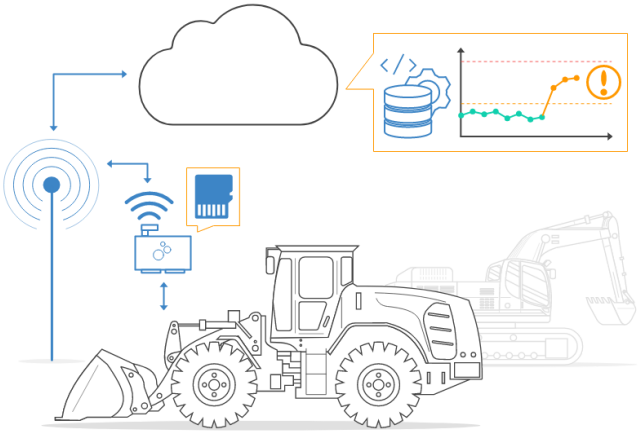
Below we provide example use cases to illustrate how the CANedge2 can be used in practice.

On-road transit bus data via in-vehicle WiFi

Need to monitor operational data across a fleet of transit buses?

In most buses, the CANedge2 can be connected easily via a DB9-J1939 adapter - and will start logging raw J1939 data to the SD card. If the transit bus has an in-vehicle WiFi access point, the CANedge2 can use this to get online and auto-push data to a dedicated server or cloud in near real-time. Here, data can be analyzed e.g. ad hoc via your favorite CAN tools or in the free asammdf GUI. You can also set up dashboards to enable low cost WiFi fleet management.





Predictive maintenance via periodic uploads

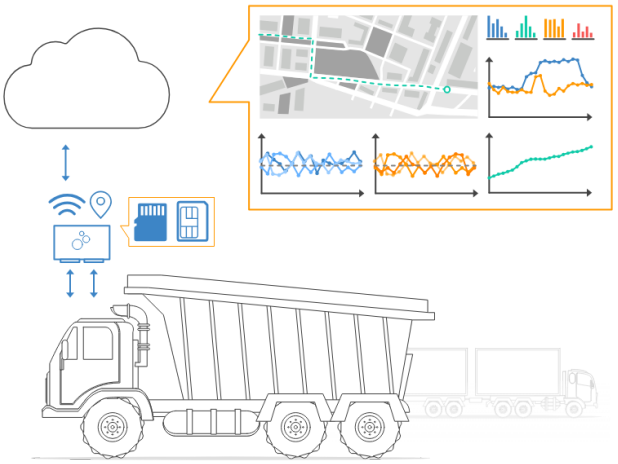
Need to predict breakdowns across construction or mining vehicles?

Some heavy-duty vehicles operate e.g. in mines with no 3G/4G connectivity (see e.g. our mining telematics intro). Others operate in specific areas like construction yards or warehouses. In these cases, the CANedge2 can be set up to connect via 1-5 WiFi access points in the area. When a vehicle gets in range of a WiFi access point, data recorded on the SD card is auto-pushed to your server (local, dedicated or cloud). Here, it can be converted to physical values using a J1939 DBC file and the free APIs - e.g. to spot irregular patterns that indicate issues.

Near real-time fleet monitoring via 3G/4G

Need to get data from a fleet of vehicles via cellular uploads?

As an OEM, collecting quality field data from J1939 mobile assets is vital to many use cases - e.g. to provide insurance, handle disputes, for development or diagnostics. With the CANedge3, data can be uploaded via 3G/4G to your own server, using your own SIM card. Further, devices can be easily configured/updated over-the-air with a few clicks. If devices go out-of-coverage, data is simply buffered on the SD card until 3G/4G coverage is back. Further, the CANedge3 has an internal GPS/IMU, adding information like position, speed, trip distance, acceleration and more to your log files.



Case study: J1939 telematics

Learn how Antu Energia uses the CANedge2 to remotely collect CAN/J1939 from heavy duty trucks and buses. Data is uploaded via a 3G/4G access point to their own AWS S3 cloud server for use in e.g. analyses, dashboards and script automation.

"CSS Electronics devices/software have been an absolute cornerstone to the success of many of our projects - and the customer service is one of the best I've ever had"

Read case study
40+ other case studies

J1939 data from a truck

Below you can download CANedge J1939 sample data.

You can also download the free software and try out the process of decoding the raw J1939 data with our J1939 DBC demo.

raw J1939 & dbc decoded J1939 data software

FAQ

How do you collect & convert J1939 data?

What vehicle data can be logged via J1939?

Most commercial trucks and heavy duty vehicles use the SAE J1939 protocol (or the derived FMS standard) for the vehicle CAN bus sensor network. A CAN logger can therefore serve as a truck data logger, assuming it can record the raw CAN bus data (which will in this case be equivalent to J1939 data).

In contrast to e.g. passenger cars, this means that the method for converting data to physical values (aka human-readable form) is standardized across many brands, years and manufacturers. From a vehicle data logging perspective, this allows end users (e.g. fleet managers or operators) to get data for their fleet management systems, without having access to the proprietary conversion rules owned by the OEMs (original equipment manufacturers).

To decode raw J1939 data from e.g. a truck or tractor, you'll need a database of conversion rules. Typically, a J1939 DBC file is used for this purpose. A J1939 DBC file takes outset in the official SAE J1939 standards, which provide conversion info for a large share of the data that is potentially available in a given vehicle.

However, most vehicles also use a number of proprietary parameters - this data still conforms to the J1939 standard, but you'll need to reverse engineer the CAN data to identify how to interpret these specific parameters.

With that in mind, we've listed below some examples of data parameters that may be available in J1939 heavy duty vehicles:

- Vehicle Speed
- Engine RPM
- Tachograph Data
- Fuel Consumption (Per Km, Per Hour)
- Tire Pressure
- Temperatures (Oil, ...)
- Throttle Position
- Battery Voltage
- GPS Position (if built-in)
- Gas Mass Flow Rate
- Brake Position
- Engine Avg. Fuel Economy
- ABS Active
- Engine Exhaust
- Engine Torque
- Steering Wheel Angle
- Gear Position
- Door Lock Status
- Vehicle Make/Model/Serial/VIN
- Hydraulic Pressure
- Emergency Braking Active
- Pitch/Roll Angle
- Oil Level & Pressure
- Fuel Level
- Trip Counter
- Coolant Temperature
- Cargo/Trailer Weight
- Diagnostic Trouble Codes (DTCs)

If you've logged a range of CAN IDs from your truck, bus, harvester or other J1939/FMS vehicle, you can identify what J1939 PGNs are contained in your data. To do so, use our CAN ID to PGN converter or learn more on our J1939 DBC page.

Can I also use the CANedge2 as a car telematics device?

Yes, the CANedge2 is basically a CAN bus data logger - meaning that it can record raw CAN data from any high speed CAN bus (ISO 11898-2) application. This includes practically all vehicles, including car fleets like taxis, police cars, ambulances etc.

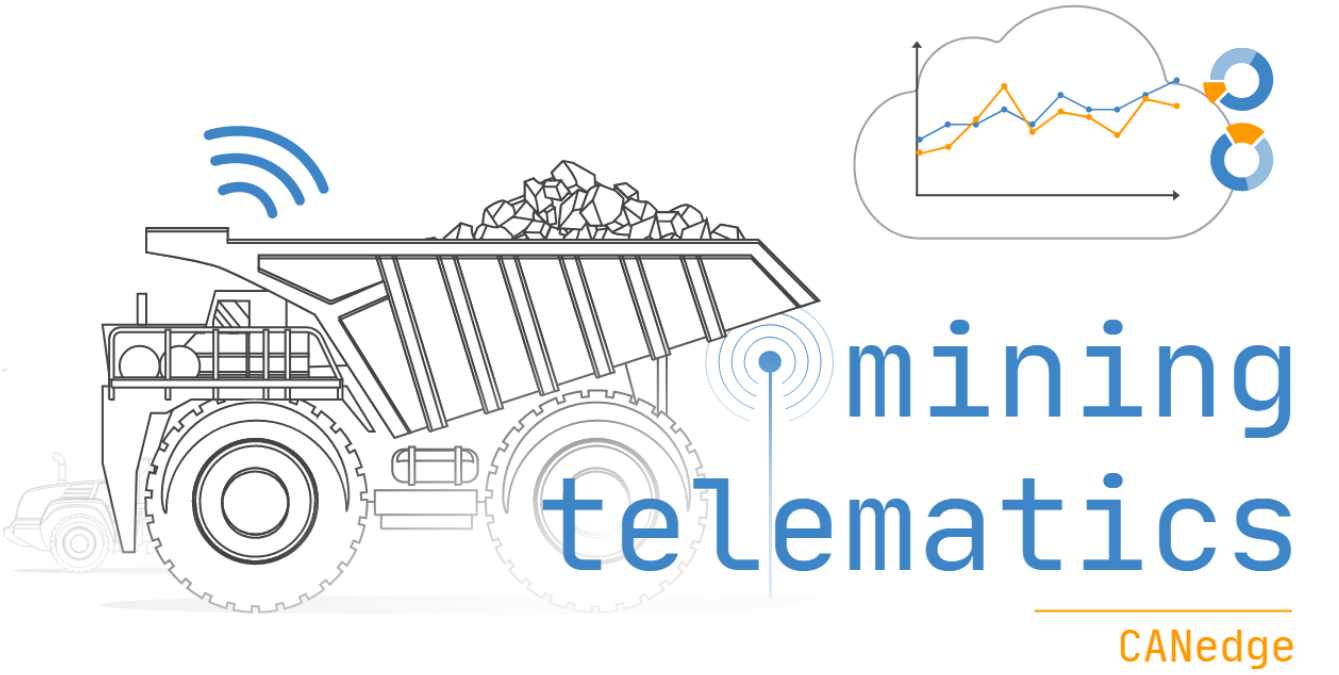
Typically you'll then log OBD2 data, which can be seen as the equivalent to J1939 data - but for passenger cars.

If you're interested in car fleet management, check out our OBD2 data logger article.

What is ELD compliance?

The ELD mandate is a US regulation that specifies that operators of certain commercial vehicles will need to use electronic logging devices (ELD) to monitor driver activity. This can include operational data like hours of service (HOS). It basically replaces the requirement for paper log books.

The CANedge can be a simple low cost method for storing these logs by recording the relevant J1939 parameters - either to the SD card, or alternatively transferring the data to your own server for automated compliance. To learn more, contact us.

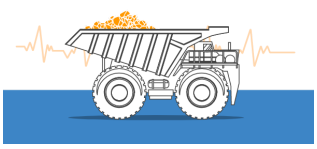


Mining Telematics - J1939 IIoT Logger

In this practical step-by-step guide you'll learn how to use the CANedge2 IIoT J1939 logger in underground mining telematics - from raw data to mining dashboards.

Top 4 benefits of mining telematics

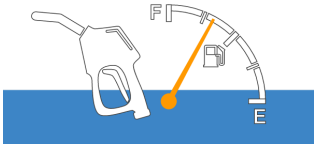
Our users within mining (contractors/OEMs) generally quote 4 key productivity benefits of mining telematics. You probably know these, so we'll twist it by linking these with examples of relevant J1939 data parameters:



Utilization & payload

How many tons are moved? By which vehicles? When? How many hours did a vehicle run/idle last month? Knowing the "pulse" of your machines and the usage efficiency is vital to a variety of use cases

`Engine Hours, Engine Speed, Payload Weight Sensor Status`



Fuel cost reductions

How much fuel is used per vehicle? By driver? By period? By correlated parameter? Monitoring fuel at scale can vastly improve your TCO (total cost of ownership) - and help prevent e.g. fuel theft

`Fuel Rate, Fuel Level, Tire Pressure, Vehicle Speed`

Predictive maintenance

Is a vehicle going to break down next week? Mining vehicle downtime is extremely costly - and even very basic predictive maintenance can make a huge difference to ensuring your asset health

`Diagnostic Trouble Codes (DTCs), Temperatures`

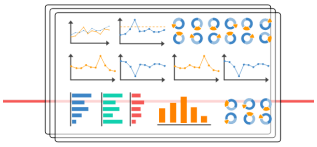
Blackbox & diagnostics

Does a vehicle exhibit a rare issue? Is monitoring required for insurance / compliance? By logging data continuously, you'll be able to review past events and diagnose issues much faster

`Multiple J1939 parameters with focus on e.g. outliers`

The challenge with traditional mining telematics

Most companies attempt to harvest these benefits - but many fail due to below challenges.



Too complex

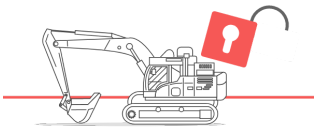
Signed up for an expensive E2E telematics platform - but you only use 10% of the features? Your solution scope must match internal capabilities. If not, costs will drastically outweigh the value

No 3G/4G coverage

Do you have good cellular connectivity in your underground mine? If not, this disqualifies most traditional telematics dongles as they rely on consistent 3G/4G coverage to push data

No data ownership

Tired of generic PDF reports? What about vendor lock-in? As an end user, you often get one-size-fits-all dashboards or aggregated low resolution data - blocking opportunities for tailored use cases



Security risks

Are you OK sending your CAN bus data to a server that you do not own? Is it encrypted? What if a dongle is compromised? Many telematics providers are not addressing cybersecurity adequately

How does the CANedge2 solve the mining telematics challenges?

The CANedge2 provides an alternative mining fleet management system:

Start simple

It is extremely simple to get started. Add to that the benefit of zero software/API costs, zero monthly fees and zero vendor lock-in. Together, these factors let you scale up both your volume and solution scope at your own pace. At the same time, the versatile configuration and open software/API suite allows the device to scale with your ambitions.

Flexible WiFi

The SD card & WiFi combo is ideal for underground mining. The device logs data to the extractable SD card when offline, ensuring zero data loss. When in range of one of 1-5 WiFi access points (WLAN routers or 3G/4G hotspots), the device will auto-push the data to your server. This allows for effective telematics - even with sporadic coverage.

You own the data

You have 100% control of your data: The device lets you customize your data flow - e.g. message filters, data compression, triggers and transmit lists. Further, you can easily process incoming data via open APIs - e.g. to create mining fleet dashboards or transform it into relational databases (SQL, Historian, PostScript) for your BI system. All data is owned by you.

Top security

Finally, the security is designed to be best-in-class: Data can be encrypted on the SD and sent securely via HTTPS to your server. Your WiFi/server details can be encrypted on the SD card. Further, configuraton & digitally signed firmware can be securely updated over-the-air across your fleet.

The CANedge mining telematics data logger

The CANedge CAN bus data logger offers optional GPS/IMU, WiFi and/or 3G/4G - ideal for mining fleet telematics:

PLUG & PLAY
Log data out-the-box. Standalone. Link your vehicle to your server in <2 minutes

PRO SPECS
Extractable 8-32 GB SD. 2xCAN/LIN. CAN FD. Zero data loss. 50 Âµs RTC. Error frames. MF4

SO SMALL
Only 8 x 5 x 2 CM. 100G. Robust alu enclosure. 5+ LEDs. Configurable 5V power out (CH2)

WIFI/LTE
Push data via WiFi or 3G/4G to your server. E2E security. OTA updates

GNSS + 3D IMU
Built-in GPS/IMU. 3x accuracy via sensor fusion. Position, speed, distance & more

OPEN SOURCE
Free open source software/APIs. MF4 to ASC/CSV. DBC support. Python. Dashboards

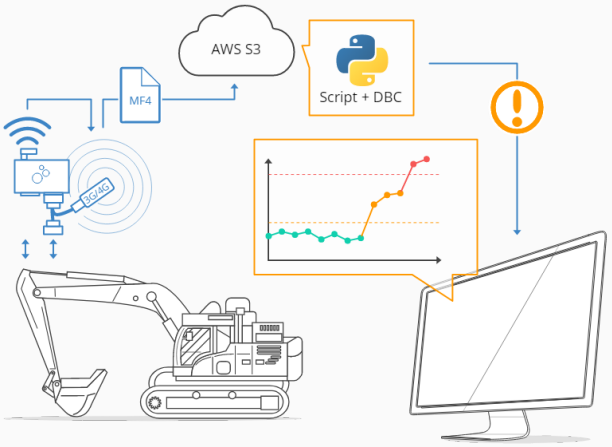
[Check out the 5 min CANedge intro video]

Case study: J1939 telematics

Learn how one of our larger OEM customers use the CANedge2 to collect CAN bus data from heavy machinery in the field - and use it to perform simple automated predictive maintenance.

"Both the hardware and support from CSS Electronics have been outstanding!"

Read case study
40+ other case studies

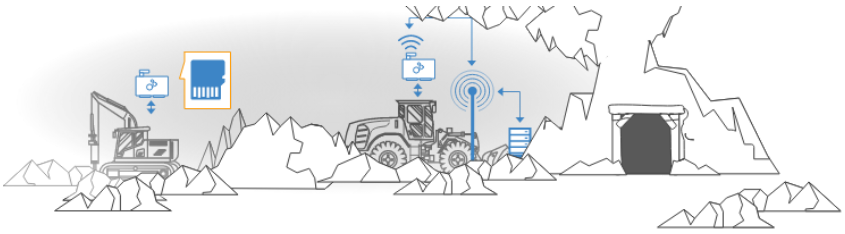


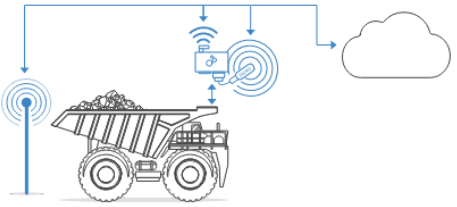
Mining telematics: Step-by-step guide

In the following section, we assume that you're an engineer at a mining OEM/contractor. You've been tasked with setting up automated data acquisition across a range of vehicles - and below we help you get started.

#1 Consider your data flow

It's a good idea to consider your data flow up front - below we outline three popular setups:





Underground SD + WiFi
Here, the CANedge2 connects via underground WiFi access points, sending logged data from the SD card when in WiFi range. Ideal for mining vehicles that primarily operate below ground.

Above ground 3G/4G/WiFi
Alternatively, your CANedge2 can upload data via WiFi when arriving at a maintenance site. Or, you can use the CANedge3 to upload via 3G/4G when above ground. Useful if you have no underground WiFi network.

Mix
The device supports 1-4 WiFi access points. You can thus configure a CANedge2 to both upload via an underground WiFi network and via a 3G/4G USB router for maximum coverage.

#2 Select your hardware

Next, you need to select a hardware bundle for your initial pilot test.

What is the recommended starter kit?

To get started, we recommend getting a CANedge2 or CANedge3 along with a set of adapter cables. The most common mining vehicle adapters are our J1939-DB9 and CAT-DB9 adapters. If you have experience recording CAN data directly from the wiring harness, you can also consider our contactless CAN readers. For details, see our products page.

Do you need WiFi, LTE or just SD logging?

For some use cases you may simply need to log CAN data to an SD card. For example, if your aim is to only analyze data if an issue occurs or as part of dispute handling, a CANedge1 dual CAN logger will suffice. The CANedge1 is identical to the CANedge2/CANedge3, except that it does not support WiFi/LTE. If you're in doubt which option is best for your use case, contact us.

What type of connectors are in your vehicles?

J1939 deutsch 9-pin | **Caterpillar 9-pin** | **Other types** | **Raw wiring harness**

In most heavy duty vehicles, you'll find a J1939 deutsch 9-pin connector in the cabin (sometimes behind a panel). If you see this, check if the relevant pins are available (power, ground, CAN low, CAN high). If so, you'll be able to use a DB9-J1939 adapter.

In CAT engines, you'll often see a 9-pin connector with a different pin-out. In this case, contact us as we may be able to supply a custom DB9-Deutsch-9-pin (CAT) cable for this. You will not be able to use a DB9-J1939 adapter in a CAT engine.

Other connectors exist - here we recommend to identify the pin-out (e.g. from a spec PDF or via an oscilloscope). After that, you can find an open-wire matching adapter and create your own custom cable. Optionally use our DB9-generic adapter.

In some cases, you may prefer to log data directly from the CAN wiring harness. In this case you can use a CANCrocodile adapter to snap onto the CAN L/H.

An example of a Sandvik mining truck connector interface.

Do you need to log multiple CAN buses?

In some heavy duty mining vehicles like dump trucks, excavators, drillers etc you may have two separate CAN buses, each containing separate data parameters. The CANedge2 lets you record data from both CAN buses with synced timestamps into the same log file. In this case, you'll of course need a suitable adapter cable for both CAN buses.

Note that often one CAN bus will be J1939 based, while the other may e.g. be CANopen based - with the latter being predominantly proprietary in nature. In such cases, we would recommend to start with the J1939 data and review whether data from the 2nd CAN bus is needed - and if so, reverse engineer the relevant parameters.

Another typical use case for multiple CAN buses is logging J1939 vehicle data via one channel - while using the 2nd channel of the CANedge for additional sensor modules. Examples include tire pressure sensors, payload sensors, hydraulic sensors, analogue/digital inputs, GPS & accelerometer, fuel sensors etc. Assuming you have sensor modules that broadcast their data via CAN, you can combine one or more of these into a single CAN bus and feed the data into the CANedge2 channel 2 port.

Will you upload data via WLAN or 3G/4G?

The CANedge2 can connect to your existing WiFi access points in e.g. an underground mine to upload data.

However, it is also possible for the device to upload data via a cellular hotspot like a USB 3G/4G WiFi router. This can be useful if you e.g. wish to upload data from a truck when it resurfaces from your mine and regains cellular connectivity.

However, if your aim is to purely upload data via 3G/4G, we instead recommend using our CANedge3, which is superior for this use case.

Do you need GNSS/IMU tracking?

In some cases, you may need to add GPS tracking as part of your mining fleet management system and mining dashboards. There are different ways of achieving this. Some mining trucks have a built-in GPS module that broadcasts GPS data via the CAN bus. If the data is encoded as per the J1939 protocol standard, you can easily decode this GPS data to human-readable form and use it along with other CAN signals.

If no built-in GPS is available, you can use a CANedge incl. GNSS/IMU.

#3 Log your first data

To log raw vehicle data to the SD card, simply connect the CANedge to your vehicle and later disconnect it.

If you convert the data into CSV using our MDF4 converters the result may look as below raw J1939 truck data:

```
TimestampEpoch;BusChannel;ID;IDE;DLC;DataLength;Dir;EDL;BRS;DataBytes
1578922367.777150;1;14FEF131;1;8;8;0;0;0;CFFFFFF300FFFF30
1578922367.777750;1;10F01A01;1;8;8;0;0;0;2448FFFFFFFFFFF
1578922367.778300;1;CF00400;1;8;8;0;0;0;107D82BD1200F482
1578922367.778900;1;14FF0121;1;8;8;0;0;0;FFFFFFFFFFFFFCFF
...
```

#4 Transfer data to your server

Once you've confirmed that you can log data as expected, you can set up the WiFi and server flow.

Once set up, you can test if your device correctly uploads data by logging into your server via our free open source CANcloud tool. CANcloud is basically a 'cockpit' for your deployed devices.

Don't worry if you've never set up a server before.

We provide step-by-step guidance in our get started docs. For example, you can set up an AWS S3 cloud server in <5 min. You then simply add your WiFi/server details to your configuration file - and you're ready.

#5 Process your data

Once you've verified that your data is coming into your server as expected, you can start processing the data.

For ad hoc analysis (e.g. diagnostics), you can load the log files in e.g. asammdf to DBC convert and plot it. Or, you can convert the data to e.g. Vector ASC and analyze it via CANalyzer.

You can also automate your processing via the free CAN bus Python API.

Regarding DBC files

A DBC file is a database of conversion rules - it's a text file that details how to go from raw CAN data to human-readable data.

Most mining vehicles are heavy duty and communicate data via the J1939 protocol. As such, you can use a J1939 DBC file to convert the raw data into human-readable form. It differs by vehicle model/year to what extent the data can be converted - we typically see 30-60% conversion in most cases. As such, we also offer that you can send us a raw log file that we can convert on your behalf to serve as a "demo" of what the DBC would yield in your use case before you purchase it. Of course, if you're the OEM of the vehicle in question, you'll most often have access to the full DBC file covering 100% of the data.

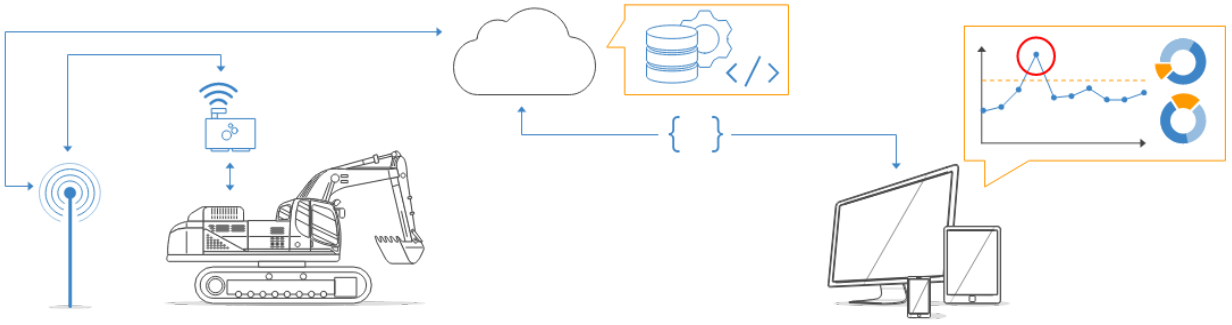
Below is a sample of DBC converted J1939 truck data:

```
timestamps,ActualEnginePercentTorque,EngineSpeed,EngineCoolantTemperature,EngineOilTemperature1,EngineFuelRate,EngineTota
lIdleHours,FuelLevel1,Latitude,Longitude,WheelBasedVehicleSpeed
2020-01-13 16:00:13.259449959+01:00,0,1520.13,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.268850088+01:00,0,1522.88,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.270649910+01:00,0,1523.34,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.271549940+01:00,0,1523.58,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.278949976+01:00,0,1525.5,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.289050102+01:00,0,1527.88,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.299000025+01:00,0,1528.13,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.308300018+01:00,0,1526.86,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.309099913+01:00,0,1526.75,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.317199945+01:00,0,1526.45,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.319149971+01:00,0,1526.38,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.320349932+01:00,0,1526.15,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.23
2020-01-13 16:00:13.326800108+01:00,0,1524.93,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.25
2020-01-13 16:00:13.329050064+01:00,0,1524.5,92,106,3.8,1868.3,52,40.6440124,-76.1223603,86.26
```

#6 Visualize your data

A final step may be to visualize your data in mining dashboards.

You can see our dashboard intro on how to set up free, customizable Grafana dashboards. You can also set up your own BI integrations using the API or e.g. use other dashboard tools like Dash by Plotly - you decide.



Thank you for reading our guide - we hope you found it useful!

CSS Electronics | DK36711949 | Soeren Frichs Vej 38K, 8230 Aabyhoej, Denmark

www.csselectronics.com | contact@csselectronics.com | +45 91 25 25 63 | LinkedIn

Products | Software | Guides | Case studies