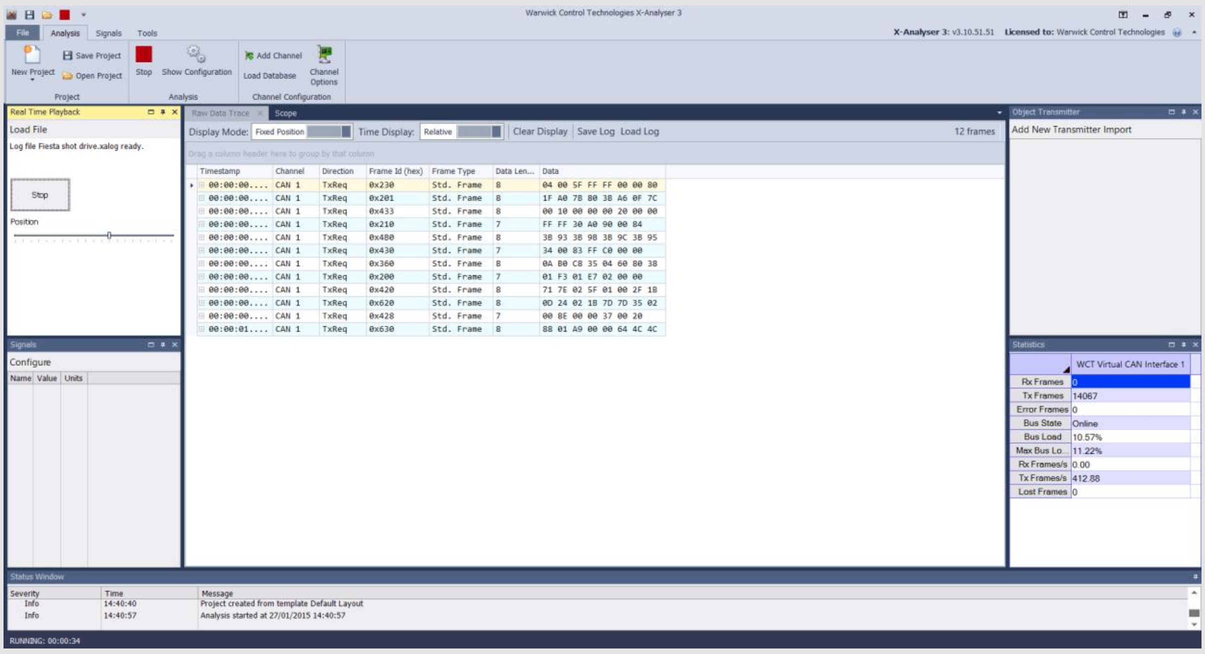
Extracting Signals data from raw CAN data

Richard T. McLaughlin – Warwick Control

In the last article, we explained the content of the CAN frame, and showed how to visualise it on a PC based analysis tool. The CAN frame consists of the Identifier (ID), Control (Data Length), Data (up to 8 Bytes) and error checking (CRC) portion. As shown in Figure 1, this can be displayed in an analyser where can see this information on the CAN bus. Now the question is – what is this telling us?

Here you can see the CAN IDs, their Data Length (DLC) and the raw data in Hexadecimal format, along with the timestamp for each message. Also seen here is the Frame Rate, which interpreted into Bus Load. Here we see a Bus Load of 10.57 % (Max 11.2%), which is quite a low Bus Load compared to higher spec vehicles today.



Timestamp	Channel	Direction	Frame Id (hex)	Frame Type	Data Len...	Data
00:00:00....	CAN 1	TxReq	0x230	Std. Frame	8	04 00 5F FF FF 00 00 80
00:00:00....	CAN 1	TxReq	0x201	Std. Frame	8	1F A0 7B 80 3B A6 0F 7C
00:00:00....	CAN 1	TxReq	0x433	Std. Frame	8	00 10 00 00 00 20 00 00
00:00:00....	CAN 1	TxReq	0x210	Std. Frame	7	FF FF 30 A0 90 00 84
00:00:00....	CAN 1	TxReq	0x4B0	Std. Frame	8	3B 93 3B 9B 3B 9C 3B 95
00:00:00....	CAN 1	TxReq	0x430	Std. Frame	7	34 00 83 FF C0 00 00
00:00:00....	CAN 1	TxReq	0x360	Std. Frame	8	0A B0 C8 35 04 60 B0 38
00:00:00....	CAN 1	TxReq	0x200	Std. Frame	7	01 F3 01 E7 02 00 00
00:00:00....	CAN 1	TxReq	0x420	Std. Frame	8	71 7E 02 5F 01 00 2F 1B
00:00:00....	CAN 1	TxReq	0x620	Std. Frame	8	0D 24 02 1B 7D 7D 35 02
00:00:00....	CAN 1	TxReq	0x428	Std. Frame	7	00 8E 00 00 37 00 20
00:00:01....	CAN 1	TxReq	0x630	Std. Frame	8	88 01 A9 00 00 64 4C 4C

	WCT Virtual CAN Interface 1
Rx Frames	0
Tx Frames	14067
Error Frames	0
Bus State	Online
Bus Load	10.57%
Max Bus Lo...	11.22%
Rx Frames/s	0.00
Tx Frames/s	412.88
Lost Frames	0

Severity	Time	Message
Info	14:40:40	Project created from template Default Layout
Info	14:40:57	Analysis started at 27/01/2015 14:40:57

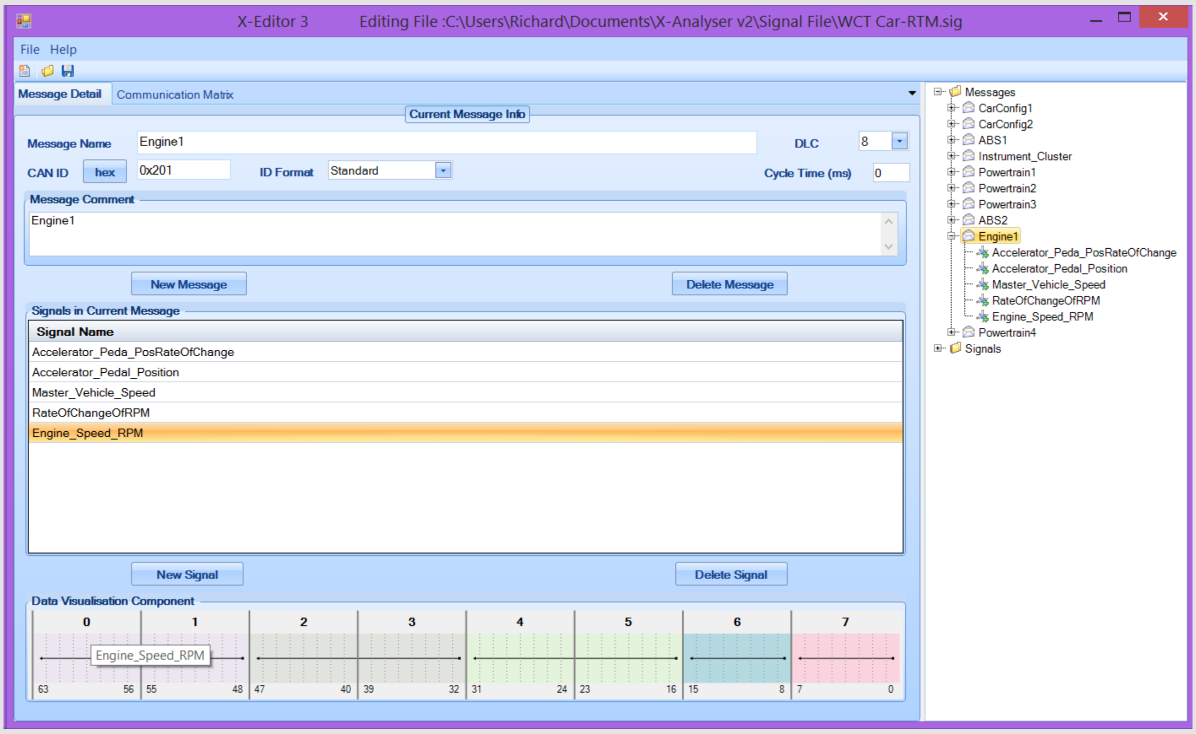
Figure 1. CAN analyser raw data display

In this article, we will explore how we can interpret this information utilising what is known as the CAN Signal Database. Here we will utilise the CAN Signal Database interpretation capabilities of the X-Analyser with its built in CAN Signal Database editor – the X-Editor. We will also show you basic ideas on how to identify data for reverse engineering purposes.

The CAN Signal Database file gives a breakdown of the CAN frame information - what the CAN frame Identifier represents, and how the data in the CAN frame is interpreted to give engineering values such as engine speed, road speed, etc. Once the CAN frame is passed from the CAN interface to the microcontroller of the ECU, the CAN Signal Database sets the rules on how the data is stored in the controller’s register, how each signal is extracted from the register, and how the data is scaled to give the engineering values.

This vehicle had few ECUs, as it was basic small coupe from the mid-2000s. Looking at the X-Editor in Figure 2, it can be seen that there are 10 CAN messages designated. Here it can be seen that the messages include info from the Engine Control, Transmission Control, ABS, Powertrain and Instrument Pack modules. Note the Message/Signals tree on the right side of the display.

As an example, here we have the Engine 1 Message highlighted. In the main window, all the Signals within this message are outlined. The CAN ID is 201 Hex, and this message has 8 Bytes of data. The ID Format is Standard, which means the ID is 11 bits – Extended Format allows for a 29-bit ID. Also note that the bit representation of all the signals is shown at the bottom of the display, where you can see how the 8 Bytes of data are distributed.


X-Editor 3 Editing File :C:\Users\Richard\Documents\X-Analyser v2\Signal File\WCT Car-RTM.sig

File Help

Message Detail | Communication Matrix

Current Message Info

Message Name: Engine1
DLC: 8
CAN ID: hex 0x201
ID Format: Standard
Cycle Time (ms): 0

Message Comment: Engine1

New Message | Delete Message

Signals in Current Message

Signal Name
Accelerator_Peda_PosRateOfChange
Accelerator_Pedal_Position
Master_Vehicle_Speed
RateOfChangeOfRPM
Engine_Speed_RPM

New Signal | Delete Signal

Data Visualisation Component

0	1	2	3	4	5	6	7
63 … 56	55 … 48	47 … 40	39 … 32	31 … 24	23 … 16	15 … 8	7 … 0

Engine_Speed_RPM

- Messages
  - CarConfig1
  - CarConfig2
  - ABS1
  - Instrument_Cluster
  - Powertrain1
  - Powertrain2
  - Powertrain3
  - ABS2
  - Engine1
    - Accelerator_Peda_PosRateOfChange
    - Accelerator_Pedal_Position
    - Master_Vehicle_Speed
    - RateOfChangeOfRPM
    - Engine_Speed_RPM
  - Powertrain4
- Signals

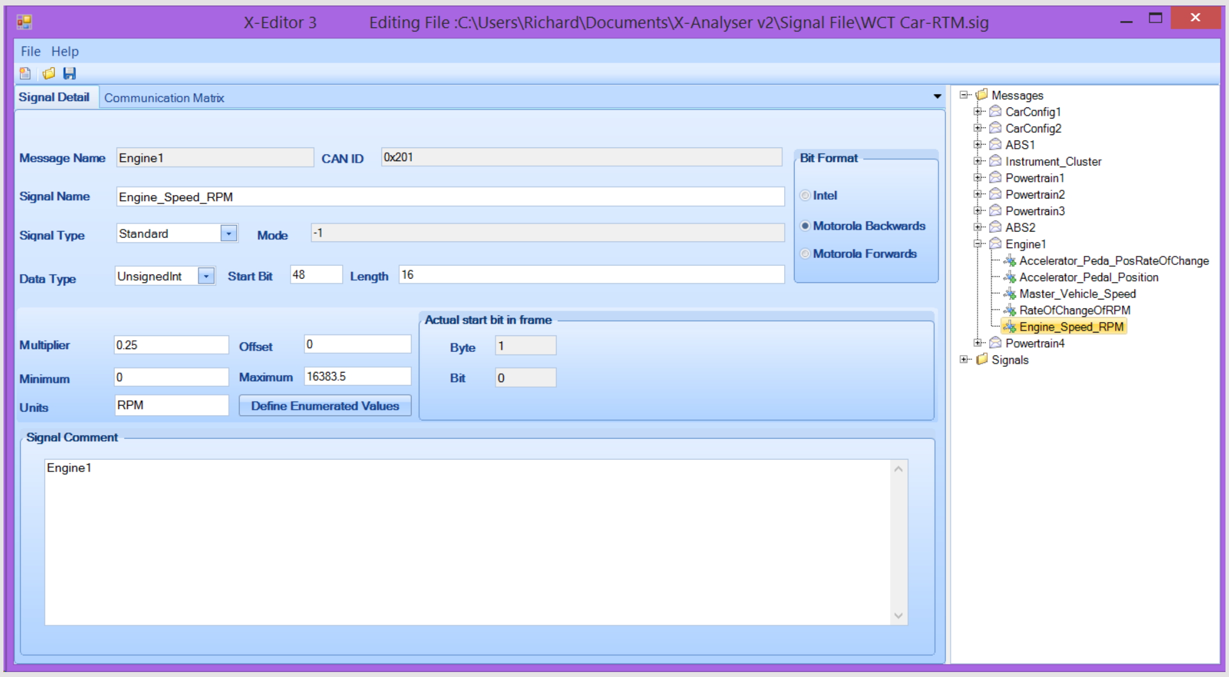
Figure 2. CAN Signal Database Editor Message Display (The X-Editor)

In this example, the Engine_Speed_RPM signal is outlined. Figure 3 shows the details of the Engine_Speed_RPM signal. This shows how the data is extracted from the CAN message, and how it is processed by the ECU software. Here you can see that the Engine_Speed_RPM signal within the Engine1

CAN frame (ID 201 Hex) is allocated 16 bits, and it is extracted from the start bit of 7. The Bit Format is Motorola Backwards.

In the CAN specification, there are three standard ways of allocating signals within a CAN frame – Intel, Motorola Backwards and Motorola Forwards. Most vehicles today use either Intel or Motorola Backwards. These formats set the boundaries and positions of the data. Referring back to Figure 2, it can be seen in the Data Visualisation Component at the bottom shows where the Engine_Speed_RPM signal is placed in the data area of the CAN frame. Note that the Start bit is 48, and the length is 16.

To understand the bit placement in the data field of a CAN frame, we shall have a review of the two most popular formats – Intel and Motorola Backwards. Consider our example of Start bit of 48 and bit length of 16. Looking at the Intel format in Figure 4, you can see the Byte assignment is left to right, and the bit assignment is right to left. This figure also shows that the Motorola Backward format has the Byte assignment right to left, as is the bit assignment. Using this example, you can see the difference between the two formats, where the bit placings are quite different. Intel format is widely used in J1939 and NMEA 2000 for the truck, bus, construction, agricultural and marine industries. Motorola Backward is widely used in many automotive industries. Motorola Forward is still used in some automotive application, but it is not as popular as Motorola Backward. You can see in Figure 4 the bit assignment of Motorola Forward is quite unorthodox as compared to the other two.


X-Editor 3
Editing File :C:\Users\Richard\Documents\X-Analyser v2\Signal File\WCT Car-RTM.sig
File
Help
Signal Detail
Communication Matrix
Define Enumerated Values

Message Name	Engine1	CAN ID	0x201		
Signal Name	Engine_Speed_RPM				
Signal Type	Standard	Mode	-1		
Data Type	UnsignedInt	Start Bit	48	Length	16
Multiplier	0.25	Offset	0		
Minimum	0	Maximum	16383.5		
Units	RPM				

Bit Format: Intel [✓ Motorola Backwards] Motorola Forwards

Actual start bit in frame: Byte 1, Bit 0

Signal Comment: Engine1

- Messages
  - CarConfig1
  - CarConfig2
  - ABS1
  - Instrument_Cluster
  - Powertrain1
  - Powertrain2
  - Powertrain3
  - ABS2
  - Engine1
    - Accelerator_Peda_PosRateOfChange
    - Accelerator_Pedal_Position
    - Master_Vehicle_Speed
    - RateOfChangeOfRPM
    - Engine_Speed_RPM
  - Powertrain4
- Signals

Figure 3. X-Editor CAN Signal Database Editor Signal Display

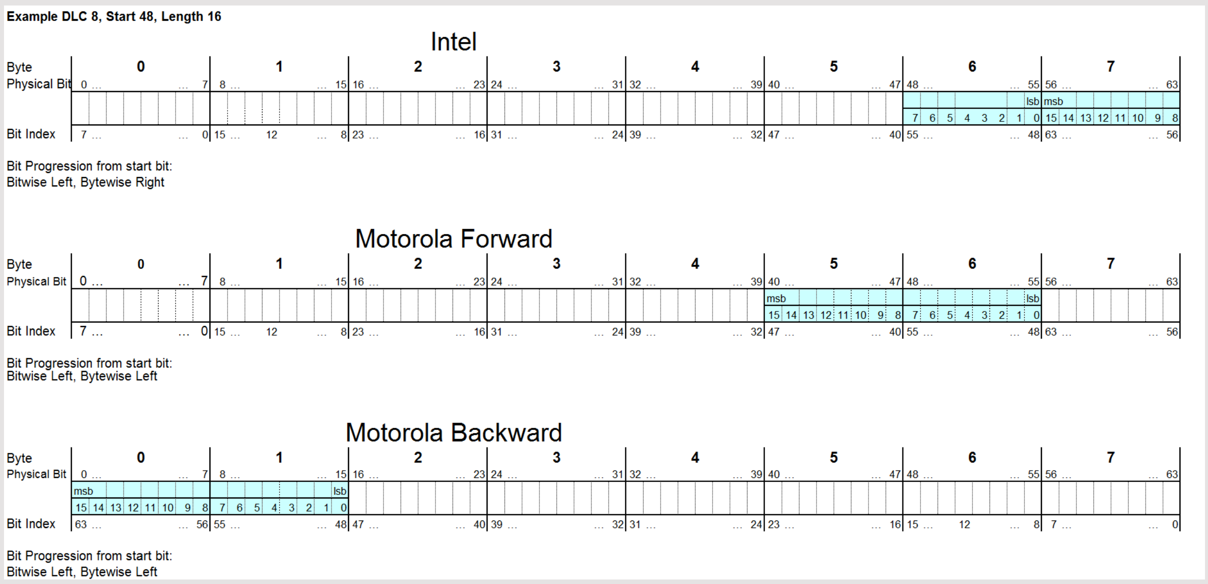

Example DLC 8, Start 48, Length 16
Intel
Byte
Physical Bit
Bit Index
lsb
msb
Bit Progression from start bit:
Bitwise Left, Bytewise Right
Motorola Forward
Bit Progression from start bit:
Bitwise Left, Bytewise Left
Motorola Backward
Bit Progression from start bit:
Bitwise Left, Bytewise Left

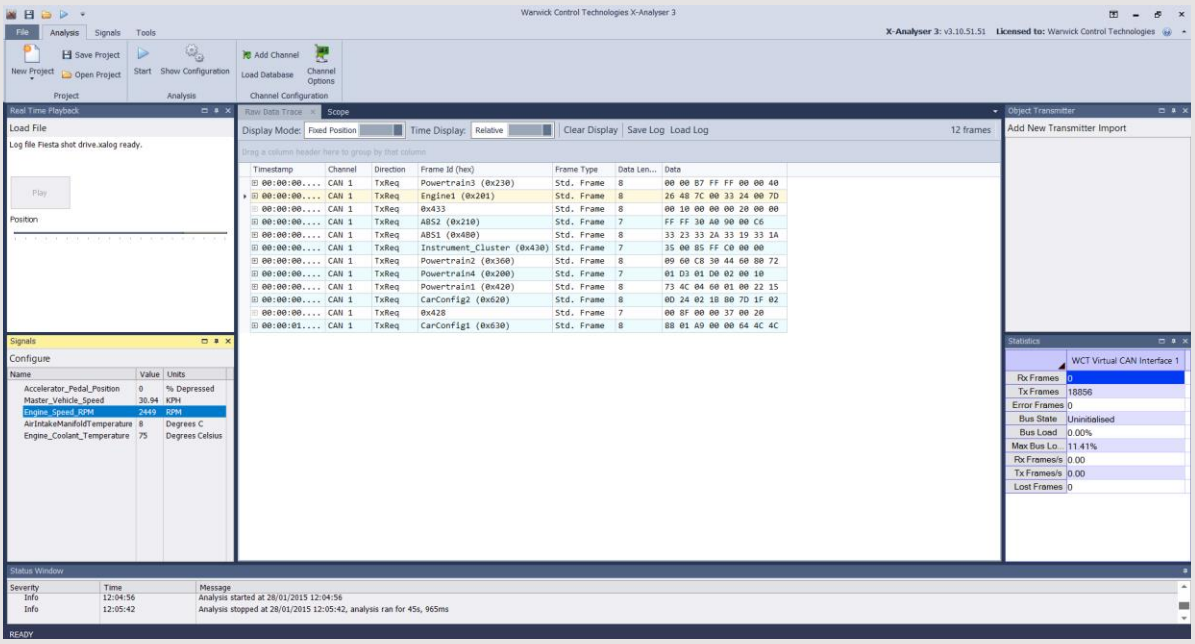
Figure 4. Intel and Motorola CAN data field bit formats

In Figure 3, you can see that the Engine_Speed_RPM signal is given a Multiplier of 0.25, and an Offset of 0. Sometimes there is a negative offset for a signal such as Temperature. The minimum of 0 and maximum of 16,383.5 RPM is set. This is the result of dividing the raw values of the 16-bit signal by ¼. The maximum raw value of 16-bit parameter would be 65,534 before it is reduced by ¼.

Once the data is allocated to an ECU, the ECU then processes it according to the specifications of this particular vehicle model. Going back to our CAN analyser, we can load a CAN Signal Database into the analyser to interpret that CAN ID and raw data seen in Figure 1. In Figure 5, it can be seen that once a CAN Signal Database file is loaded, the CAN IDs can be displayed by their allocated message name, e.g. Engine1, Powertrain2, ABS1, etc. The display still shows the raw data, so the next step is to go to a Signals display to show how the raw data is processed to give engineering values.

In Figure 5 the Signals display on the left of the analyser display shows the engineering values of the CAN data after it has been processed via the CAN Signal Database. Here we show several Signals selected in a configuration process. Once a CAN Signal Database file is loaded into the X-Analyser, you can select relevant signals to display.

Here it can be seen the instantaneous values of each of the Signals designated in the CAN Signal Database file loaded. This is done by selecting Configure in the Signals display, and you will see a list of all the Signals available from the CAN Signal Database file, which message they are embedded in. You may select as many Signals as you like for display.



Log file Fiesta shot drive.xalog ready.

Display Mode: Fixed Position | Time Display: Relative | 12 frames

Timestamp	Channel	Direction	Frame Id (hex)	Frame Type	Data Len...	Data
00:00:00....	CAN 1	TxReq	Powertrain3 (0x230)	Std. Frame	8	00 00 B7 FF FF 00 00 40
00:00:00....	CAN 1	TxReq	Engine1 (0x201)	Std. Frame	8	26 48 7C 00 33 24 00 7D
00:00:00....	CAN 1	TxReq	0x433	Std. Frame	8	00 10 00 00 00 20 00 00
00:00:00....	CAN 1	TxReq	ABS2 (0x210)	Std. Frame	7	FF FF 30 A0 90 00 C6
00:00:00....	CAN 1	TxReq	ABS1 (0x4B0)	Std. Frame	8	33 23 33 2A 33 19 33 1A
00:00:00....	CAN 1	TxReq	Instrument_Cluster (0x430)	Std. Frame	7	35 00 85 FF C0 00 00
00:00:00....	CAN 1	TxReq	Powertrain2 (0x360)	Std. Frame	8	09 60 C8 30 44 60 80 72
00:00:00....	CAN 1	TxReq	Powertrain4 (0x200)	Std. Frame	7	01 D3 01 D0 02 00 10
00:00:00....	CAN 1	TxReq	Powertrain1 (0x420)	Std. Frame	8	73 4C 04 60 01 00 22 15
00:00:00....	CAN 1	TxReq	CarConfig2 (0x620)	Std. Frame	8	0D 24 02 1B 80 7D 1F 02
00:00:00....	CAN 1	TxReq	0x428	Std. Frame	7	00 8F 00 00 37 00 20
00:00:01....	CAN 1	TxReq	CarConfig1 (0x630)	Std. Frame	8	88 01 A9 00 00 64 4C 4C

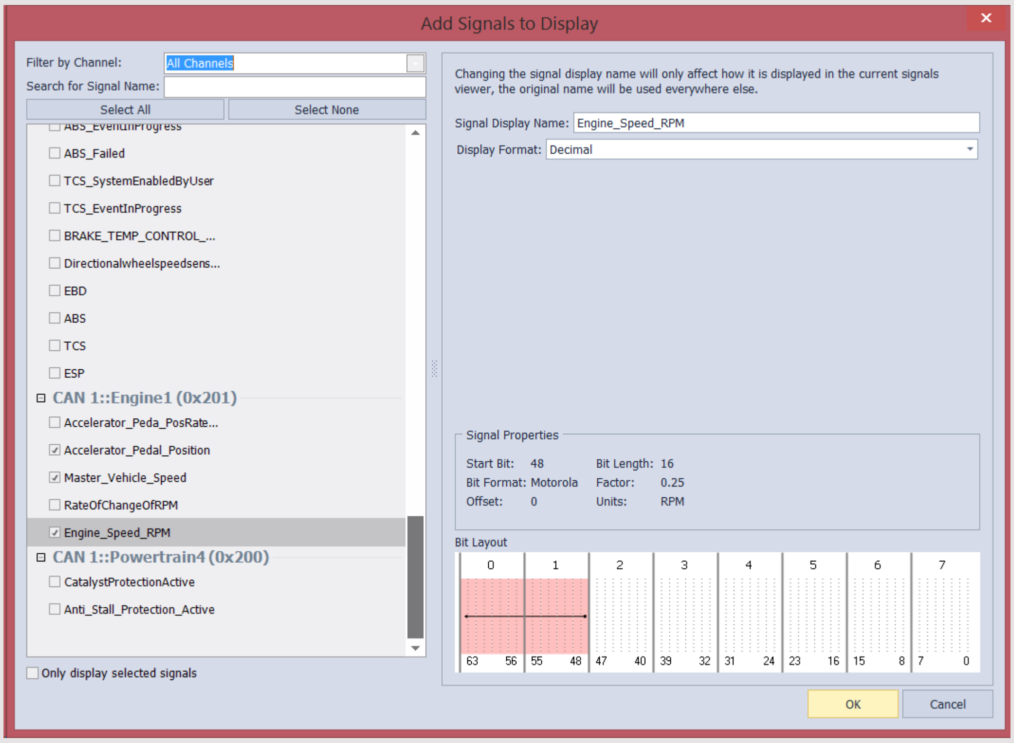
Name	Value	Units
Accelerator_Pedal_Position	0	% Depressed
Master_Vehicle_Speed	30.94	KPH
Engine_Speed_RPM	2449	RPM
AirIntakeManifoldTemperature	8	Degrees C
Engine_Coolant_Temperature	75	Degrees Celsius

	WCT Virtual CAN Interface 1
Rx Frames	0
Tx Frames	18856
Error Frames	0
Bus State	Uninitialised
Bus Load	0.00%
Max Bus Lo...	11.41%
Rx Frames/s	0.00
Tx Frames/s	0.00
Lost Frames	0

Severity	Time	Message
Info	12:04:56	Analysis started at 28/01/2015 12:04:56
Info	12:05:42	Analysis stopped at 28/01/2015 12:05:42, analysis ran for 45s, 965ms

Figure 5. CAN analyser raw data display, showing Message names.

You can review each parameter in the Signals display by right clicking on the signal of interest and selecting Properties. You will see a display as shown in Figure 6, where it reviews the message it came from (e.g. Engine 1, ID 201) along with the bit placement and multiplier info. In this example, we have selected Engine_Speed_RPM signal. You can see in Figure 6 that it is contained in the Engine 1 (ID 201) CAN message. The bit placing is shown where the start bit is 48, and the length is 16. Also, the multiplier (factor) is 0.25.


Add Signals to Display

Filter by Channel: All Channels
Search for Signal Name:

Select All | Select None

- ☐ ABS_EventInProgress
- ☐ ABS_Failed
- ☐ TCS_SystemEnabledByUser
- ☐ TCS_EventInProgress
- ☐ BRAKE_TEMP_CONTROL_...
- ☐ Directionalwheelspeedsens...
- ☐ EBD
- ☐ ABS
- ☐ TCS
- ☐ ESP

CAN 1::Engine1 (0x201)
- ☐ Accelerator_Peda_PosRate...
- ☑ Accelerator_Pedal_Position
- ☑ Master_Vehicle_Speed
- ☐ RateOfChangeOfRPM
- ☑ Engine_Speed_RPM

CAN 1::Powertrain4 (0x200)
- ☐ CatalystProtectionActive
- ☐ Anti_Stall_Protection_Active

☐ Only display selected signals

Changing the signal display name will only affect how it is displayed in the current signals viewer, the original name will be used everywhere else.

Signal Display Name: Engine_Speed_RPM
Display Format: Decimal

Signal Properties

Start Bit:	48	Bit Length:	16
Bit Format:	Motorola	Factor:	0.25
Offset:	0	Units:	RPM

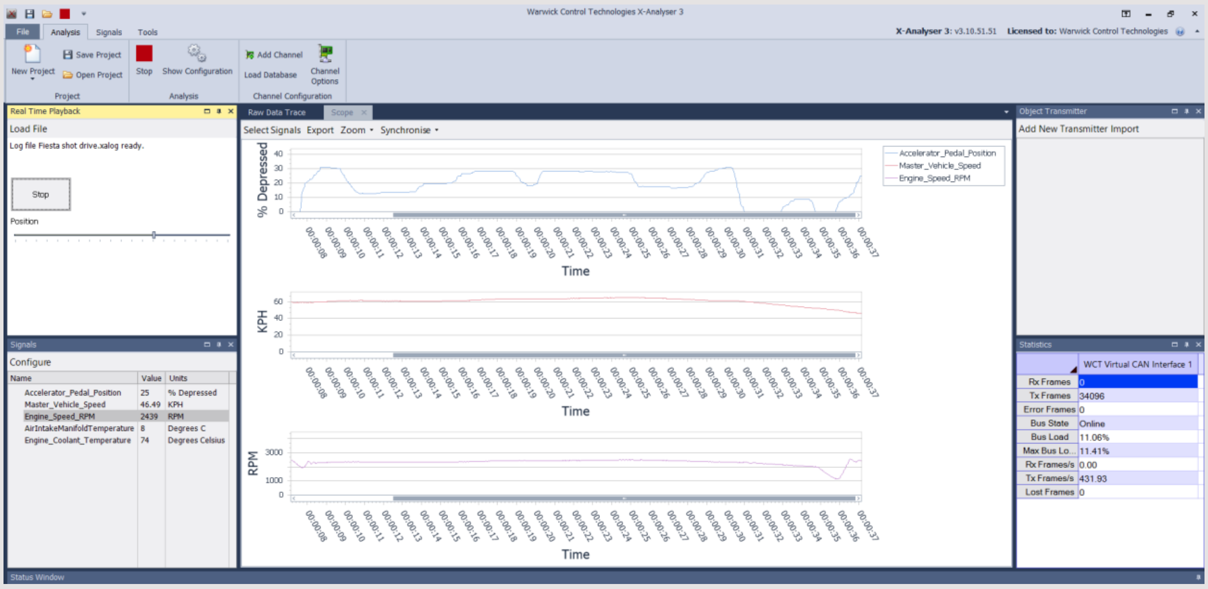
Bit Layout

0	1	2	3	4	5	6	7
63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0

OK | Cancel

Figure 6. X-Analyser Signal Properties Display

Figure 7 shows the CAN analyser's Scope display. Here we can view the trend of selected signals over a period of time. These Signals are mapped in from the loaded CAN Signal Database file the same way they are in the Signals display. Here you can see that the signals are updated in a scope format to show the trend of each of the selected signals during the data collection session. For example, the relationship between the throttle position, the vehicle speed and engine speed is illustrated here. It is possible to save this session in .csv format for later trend analysis in Excel. Many analysers provide other means of displaying signals, such as dial displays, bar codes, etc.



Log file Fiesta shot drive.xalog ready.

Signals

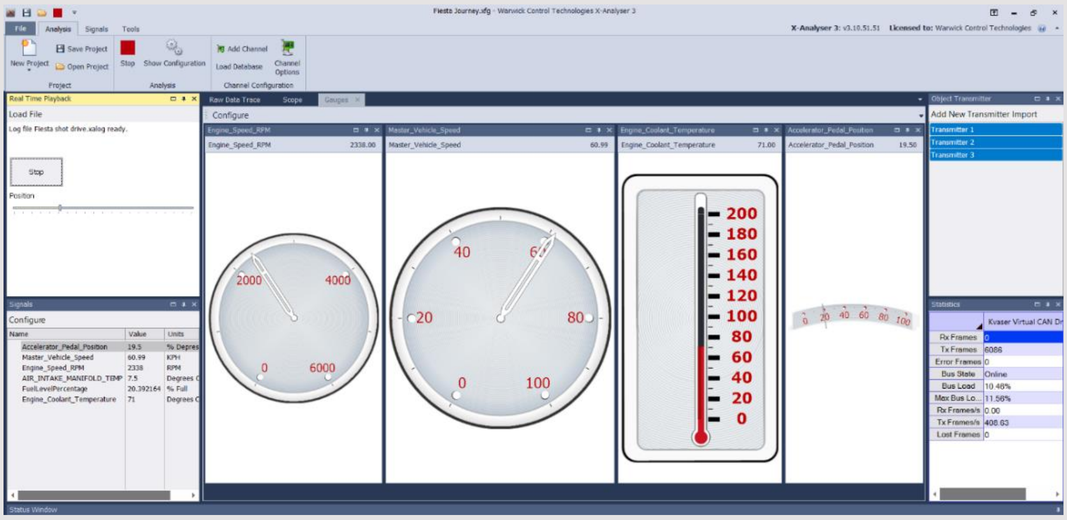
Name	Value	Units
Accelerator_Pedal_Position	25	% Depressed
Master_Vehicle_Speed	46.49	KPH
Engine_Speed_RPM	2439	RPM
AirIntakeManifoldTemperature	8	Degrees C
Engine_Coolant_Temperature	74	Degrees Celsius

Statistics

	WCT Virtual CAN Interface 1
Rx Frames	0
Tx Frames	34096
Error Frames	0
Bus State	Online
Bus Load	11.06%
Max Bus Lo...	11.41%
Rx Frames/s	0.00
Tx Frames/s	431.93
Lost Frames	0

Figure 7. X-Analyser Scope Display

It is also possible to map these signals into gauge displays in X-Analyser. The same way you map signals in the Signals and Scope display, you can map signals into your choice of Gauge display. Figure 8 shows and example of this, where four parameters are displayed as gauges – Engine Speed RPM, Master Vehicle Speed, Engine Coolant Temperature and Accelerator Pedal Position.



Log file Fiesta shot drive.xalog ready.

Signals

Name	Value	Units
Accelerator_Pedal_Position	19.5	% Depres
Master_Vehicle_Speed	60.99	KPH
Engine_Speed_RPM	2338	RPM
AIR_INTAKE_MANIFOLD_TEMP	7.5	Degrees C
FuelLevelPercentage	20.392164	% Full
Engine_Coolant_Temperature	71	Degrees C

Statistics

	Kvaser Virtual CAN Dr
Rx Frames	0
Tx Frames	6086
Error Frames	0
Bus State	Online
Bus Load	10.46%
Max Bus Lo...	11.56%
Rx Frames/s	0.00
Tx Frames/s	408.63
Lost Frames	0

Figure 8. X-Analyser Gauges Display

As discussed here, the CAN Signals Database is a file that is generated based on the specification of your particular vehicle. The CAN db will tell you what information is in each CAN ID. It also establishes how the raw digital data is extracted from the CAN frame and scaled before it is passed on to the microcontroller in the ECU for processing.

Most car companies have their own methods of interpreting CAN data, i.e. specifying the CAN Signals Database. Unfortunately, these car companies treat this propriety information as confidential, and they are not readily willing to share this information to the Aftermarket industry. There have been methods for ascertaining the CAN data information for developing CAN databases from past knowledge and reverse engineering. This is a difficult area that requires either a good relationship with the OEM or some clever reverse engineering methods to obtain relevant Signals information. We will devote a special article to reverse engineering methods utilising some advance features of the X-Analyser in the near future.

For those in low quantity production vehicle industries (truck, bus, construction, agricultural, etc.), life is a bet easier. There are standards utilised in these industries that designate standard CAN message IDs and the format of the signals data in the messages. The two popular CAN higher layer standards are CANopen and SAE J1939. For those in these industries, this is information that will be interesting in a future article.