

USER GUIDE

DbEdit User Guide

Contents

1. Copyright Notice . . . 5
2. Introduction . . . 6
    2.1. The figures . . . 6
    2.2. Symbols Description . . . 6
    2.3. Contact Information . . . 6
3. Getting Started . . . 7
    3.1. System requirements . . . 7
    3.2. Installing . . . 7
    3.3. Updates . . . 7
    3.4. The main window . . . 8
    3.5. The activity bar . . . 8
    3.6. The status bar . . . 9
    3.7. Settings . . . 10
    3.8. Keyboard shortcuts . . . 10
4. The Explorer . . . 11
    4.1. The row menu . . . 11
    4.2. Files changed on disk . . . 12
5. Editing . . . 13
    5.1. Tables . . . 13
    5.2. The inspector . . . 13
    5.3. Undo and redo . . . 13
    5.4. Saving . . . 13
    5.5. Copy and paste . . . 14
    5.6. Search . . . 14
    5.7. Problems . . . 14
6. CAN Databases (DBC) . . . 17
    6.1. The database overview . . . 17
    6.2. ECUs . . . 18
    6.3. Messages and signals . . . 18
    6.4. The bit layout . . . 20
    6.5. Multiplexing . . . 20
    6.6. J1939 . . . 20
    6.7. The network matrix . . . 20
    6.8. Value tables and value descriptions . . . 21
    6.9. Attribute definitions . . . 21
7. LIN Description Files (LDF) . . . 22
    7.1. Nodes . . . 22
    7.2. Frames and signals . . . 22
    7.3. Schedule tables . . . 24
    7.4. Encoding types . . . 25
8. A2L Files . . . 26

8.1. The module in the explorer . . . 26
8.2. Object tables . . . 26
8.3. Block pages . . . 27
8.4. IF_DATA . . . 28
8.5. Large files . . . 28
9. Licence Agreement . . . 29
1. Licence . . . 29
2. What you may share . . . 29
3. Restrictions . . . 29
4. Ownership . . . 29
5. Third-party components . . . 29
6. Your files and the update check . . . 30
7. Check your results . . . 30
8. No warranty . . . 30
9. Limitation of liability . . . 30
10. Updates and availability . . . 30
11. Termination . . . 30
12. Governing law and disputes . . . 30
13. Contact . . . 31

Document Information

Document Number	100-0-0005
Created By	Alexander Olsson
Date	2026-11-10
Revision	Rev. 00.02
Classification	EXTERNAL
Applies to	Altech DbEdit (CAN, LIN and ASAP2 database editor)

Revision History

Revision	Date	Author	Change
00.01	2026-10-26	Alexander Olsson	Initial release: installing and updating, the explorer, editing, and the DBC, LDF and A2L editors.
00.02	2026-11-10	Alexander Olsson	Licence agreement added (Chapter 9). Figures renewed; the explorer's named buttons and the bit layout of multiplexed messages.

© 2026 Altech Electronics Nordic AB. All rights reserved.

1. Copyright Notice

This document is the property of Altech Electronics Nordic AB. The document may not be edited, utilized, reproduced or otherwise exploited, in full or in part, without the written permission of Altech Electronics Nordic AB.

2. Introduction

Altech DbEdit is a Windows application for editing the description files that CAN and LIN tools and calibration tools read:

- **DBC** — CAN databases, including J1939 and CAN FD: ECUs, messages, signals, multiplexing, value tables, attributes and the network matrix.
- **LDF** — LIN description files (LIN 2.0 to 2.2A, ISO 17987): nodes and their attributes, frames, signals, schedule tables and encoding types.
- **A2L** — ASAM MCD-2 MC (ASAP2) descriptions: measurements, characteristics, axes, conversion methods, record layouts, functions, groups and the module's blocks, including IF_DATA.

Open whole folders of files, edit in tables with an inspector beside them, see every edit in the file's bit layout or time line, undo anything, and let each format's validator list what is wrong. Copy and paste moves objects between files with everything they depend on.

Hover over a control for a tooltip, and right-click almost anything for the options that apply to it. The same look, themes and keys as Altech CalTool.

2.1. The figures

The figures were made with three small demonstration files — a CAN database, a LIN description and an A2L for an engine controller — so every step can be repeated. They contain no customer data. Where a figure shows a value, a name or a description, it is an example.

2.2. Symbols Description

The following symbols are used in this user guide to highlight certain information.



Critical information — the requirements must be followed in order to not affect the performance of the system or product.



Recommended information — the requirements should be followed in order to not affect the performance of the system or product.



Additional information — further information can be found in a referenced Altech guideline or datasheet. It's recommended to study all referenced documents before installing any Altech system or product.

2.3. Contact Information

For information or sales inquiries, please contact info@altechelectronics.io.

© 2026 Altech Electronics Nordic AB. All rights reserved.

3. Getting Started

3.1. System requirements

Item	Requirement
Operating system	Windows 10 or 11, 64-bit
Disk	About 120 MB
Network	Only for the update check (Section 3.3). DbEdit works offline.

3.2. Installing

Run `DbEdit-Setup-<version>.exe`. DbEdit installs under **Altech** in the Start menu, beside CalTool if that is installed; the user guide sits there too. Nothing else needs to be installed: the program carries everything it needs.

The installer first shows the licence agreement (Chapter 9). DbEdit is free of charge.

The installer offers to open `.dbc`, `.ldf` and `.a2l` files with DbEdit, one tick box per kind. Ticked, a double-click on such a file in Windows Explorer starts DbEdit with the file added to its explorer and open. Untick a kind that should stay with another tool, such as a `.dbc` that CANdb++ owns; the other tool is still offered under **Open with**. Uninstalling DbEdit removes the associations.

Installing a newer version over an older one upgrades it in place. The installer removes the old version's files first, so nothing from an earlier build stays behind. Settings and the explorer's files and folders are kept.

When DbEdit starts, it brings back the files and folders that were in the explorer when it was last closed.

3.3. Updates

Each time it starts, DbEdit asks Altech's release page whether a newer version has been published. When one has, a blue **Update available** notice appears at the right end of the status bar:



Figure 1. The update notice at the right end of the status bar

Click it to download the new installer. A small window shows the progress; when the download is done, DbEdit starts the installer and closes. The installer upgrades DbEdit and starts it again.

If the computer is offline, or the release page cannot be reached, nothing is shown and DbEdit simply works as it is. The check never sends anything about you or your files.

© 2026 Altech Electronics Nordic AB. All rights reserved.

3.4. The main window

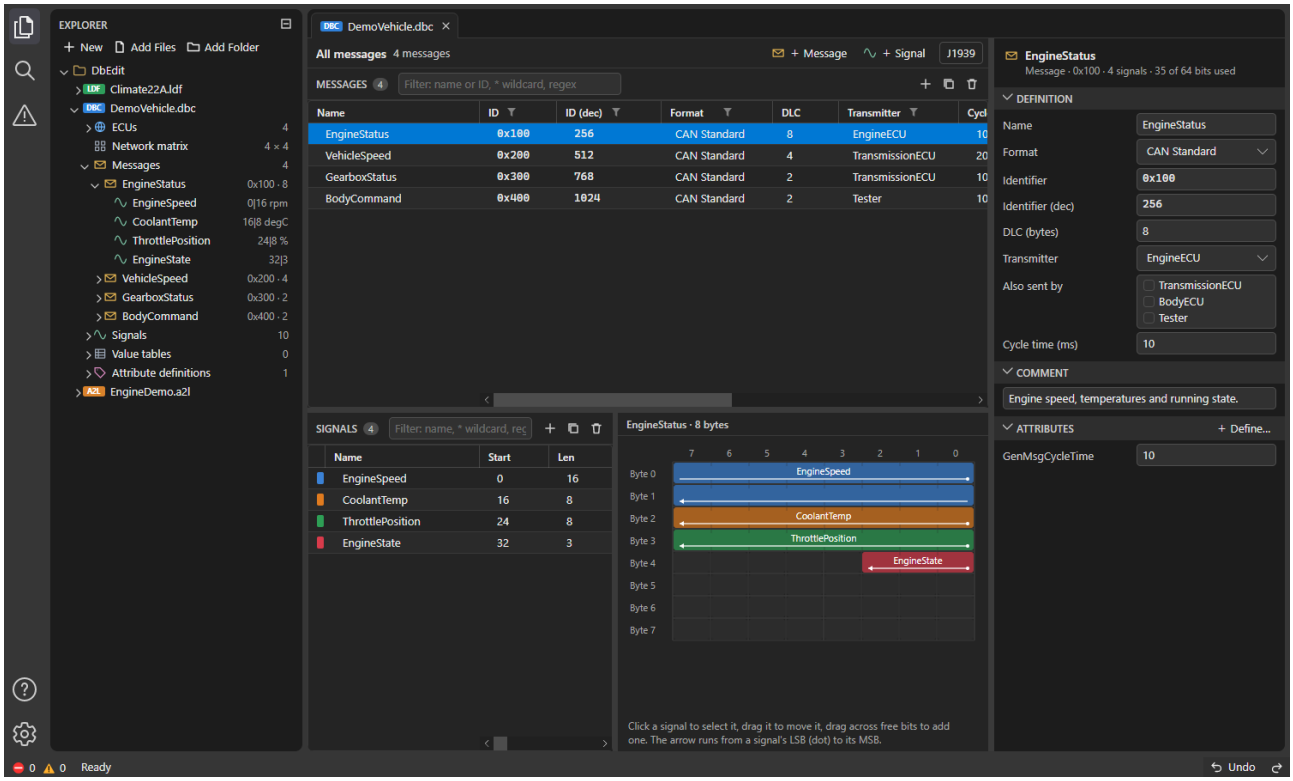

Figure 2. The main window: a CAN message selected

From left to right:

- **Activity bar** — the explorer, search, problems, the user guide and settings (Section 3.5).
- **Side bar** — the explorer with every open file and its contents (Chapter 4). Drag its edge to resize it; `Ctrl` + `B` hides and shows it.
- **Editor** — one tab per open file. What the middle shows depends on what is picked in the explorer: a list of messages with the selected message's signals and bit layout below, a schedule table's time line, a list of characteristics, a block's fields.
- **Inspector** — every property of the selected object, grouped in sections. Type in a field and leave it to apply the change.
- **Status bar** — problem counts, the last action's result, undo and redo (Section 3.6).

3.5. The activity bar

© 2026 Altech Electronics Nordic AB. All rights reserved.



From the top:

- **Explorer** (`Ctrl` + `Shift` + `E`) — files and folders, and everything in them.
- **Search** (`Ctrl` + `Shift` + `F`) — finds objects by name in every open file (Section 5.6).
- **Problems** (`Ctrl` + `Shift` + `M`) — what each file's validator found (Section 5.7).

At the bottom:

- **User guide** — opens this document.
- **Settings** — theme and text size (Section 3.7), and the version you are running.

Clicking the icon of the pane that is already showing folds the side bar away; clicking it again brings it back.

3.6. The status bar



Figure 3. The left end of the status bar

- **Problem counts** — errors and warnings across every open file. Click to open the Problems pane.
- **Status message** — what the last action did, or why it did not.

At the right end:

- **File changed** — an open file was changed by another program. It blinks until clicked; the click asks which files to reload (Section 4.2).
- **Undo** and **Redo** — the last change in the active file, named in the tooltip.
- **Update available** — a newer version can be installed (Section 3.3).

© 2026 Altech Electronics Nordic AB. All rights reserved.

3.7. Settings

Click the gear at the bottom of the activity bar to choose the **theme** (dark or light; the light theme suits a bright room) and the **text size** (small, medium or large). Both apply at once to every window and are remembered. The version of DbEdit you are running is shown at the bottom of the menu, above the **Licence** and **Third-party** buttons, which open the licence agreement and the notices for the open-source components DbEdit includes, in a window of their own.

3.8. Keyboard shortcuts

Key	Action
`Ctrl` + `N`	New database file (DBC, LDF or A2L)
`Ctrl` + `O`	Add files to the explorer
`Ctrl` + `Shift` + `O`	Add a folder to the explorer
`Ctrl` + `S`	Save the active file
`Ctrl` + `Shift` + `S`	Save the active file under another name
`Ctrl` + `W`	Close the active tab
`Ctrl` + `Z` / `Ctrl` + `Y`	Undo / redo in the active file
`Ctrl` + `C` / `Ctrl` + `V`	Copy / paste the selected objects (Section 5.5)
`Delete`	Delete the selected objects
`Ctrl` + `B`	Hide or show the side bar
`Ctrl` + `Shift` + `E` / `F` / `M`	Explorer / Search / Problems
`F2` or double-click	Edit a table cell
`Escape`	Leave a field without applying what was typed

While typing in a text field, `Ctrl` + `Z` undoes the typing; everywhere else it undoes the last change to the file.

© 2026 Altech Electronics Nordic AB. All rights reserved.

4. The Explorer

The explorer lists the files and folders you have added and everything inside each file. Its entries are remembered between sessions.

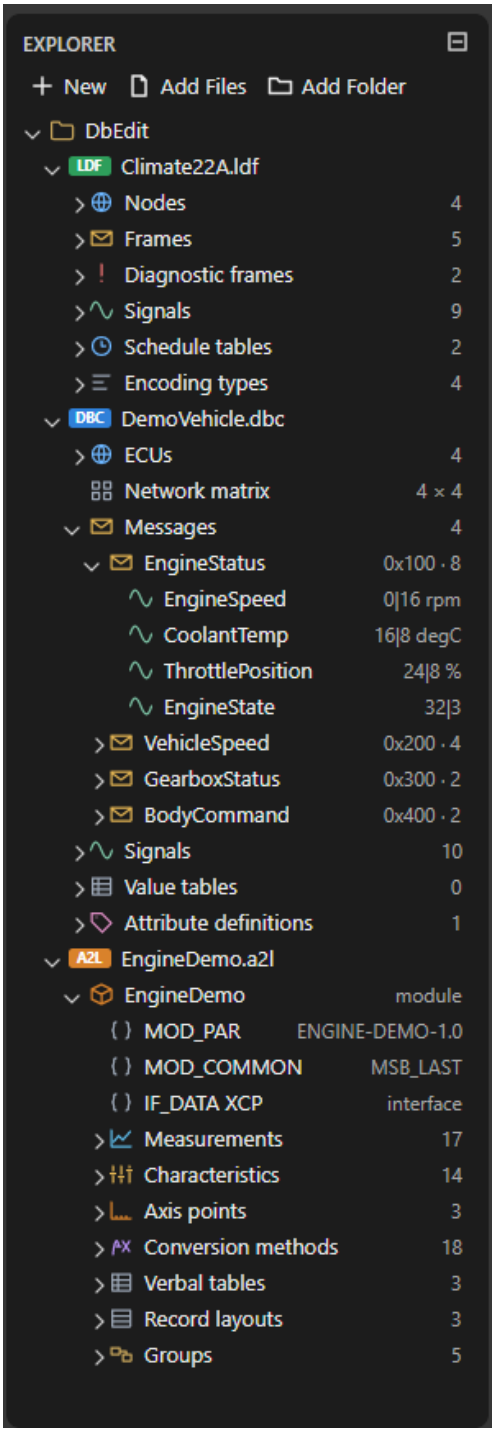

Figure 4. Three files in one folder, each expanded

1. Adding files and folders

Use **Add Files** and **Add Folder** at the top of the explorer, the keys (`Ctrl` + `O` for files, `Ctrl` + `Shift` + `O` for a folder), the buttons on the welcome page, or **drop** files or folders anywhere in the window.

A folder shows every `.dbc`, `.ldf` and `.a2l` in it and its subfolders, and follows the folder: a file that appears, disappears or is renamed on disk is reflected without doing anything.

Right-click a top entry to **remove** it from the explorer. The files on disk are not touched.

2. New files

New at the top of the explorer (or `Ctrl` + `N`) creates an empty CAN database, LIN description or A2L file where you choose, adds it to the explorer and opens it.

3. Inside a file

Expand a file to see its structure: a DBC into its ECUs (each with Tx and Rx messages), the network matrix, messages with their signals, all signals, value tables and attribute definitions; an LDF into nodes, frames, diagnostic frames, signals, schedule tables and encoding types; an A2L into its module's blocks and one list per kind of object.

The grey text at the right of a row is its most useful fact: a message's identifier and length, a signal's start bit, length and unit, a count for a list.

Click a row to open it in the editor: a list row shows the list, an object row the list it belongs to with that object selected. The file's own row opens its overview.

4.1. The row menu

Right-click a row for what applies to it:

- **Properties** — opens the object in the editor and shows it in the inspector.

© 2026 Altech Electronics Nordic AB. All rights reserved.

- **Copy** and **Paste** — the object and everything it depends on (Section 5.5). The entry names what the clipboard holds. A paste under an ECU's **Tx messages** makes that ECU the sender.
- **Duplicate** — a copy beside the original, with a free name and identifier.
- **Delete** — removes the object and every reference to it. Also `Delete`.
- **Add …** — a new object of the kind the row lists: an ECU, a message, a signal, a slave, a frame, a schedule table, a value table, an encoding type.

4.2. Files changed on disk

DbEdit watches every file it has read. When another program changes one — a build regenerates a DBC, a colleague saves an LDF — the status bar shows a blinking **file changed** notice. Click it to choose:

- **Reload selected** — read the ticked files again. A file with unsaved edits in DbEdit starts unticked, because reloading would discard them.
- **Reload all**.
- **Ignore all** — keep what DbEdit has. The next save overwrites the other program's change.

A reload keeps your place: the same object stays selected, found again by name.

© 2026 Altech Electronics Nordic AB. All rights reserved.

5. Editing

The three editors share the same way of working.

5.1. Tables

Each list — messages, signals, frames, characteristics — is a table.

- **Edit a cell** by double-clicking it or pressing `F2`, then `Enter` to apply or `Escape` to leave it. A value the file cannot hold is refused and the cell keeps its old value; the status bar says why.
- **Filter** with the box above the table: plain text matches anywhere in a name, * is a wildcard and a regular expression works too. Some columns have a funnel in their header that filters by a ticked list of values — a message's format or transmitter, for example.
- **Sort** by clicking a header: ascending, descending, then the file's own order again. Numbers sort by value, hexadecimal included.
- **Select** several rows with `Shift` and `Ctrl` for a copy or a delete.
- The + button above a table adds an object of its kind; the copy button duplicates the selected rows; the bin deletes them.

A column that is too narrow for its header can be widened by dragging the header's edge.

5.2. The inspector

The right pane shows every property of the selected object in titled sections: its definition, its comment, its attributes, what it is used by. Type in a field and leave it (`Tab`, a click elsewhere, `Enter`) to apply the change. A refused value stays in the field, marked, and is not applied.

A multi-line field shows **Apply** and **Revert** while it has unapplied text. A list of names — a message's receivers, a signal's subscribers — has a box to add one, with suggestions; * adds every match. **Edit as text** pastes a whole list at once.

Hover over a field's label for a description of what it is.

5.3. Undo and redo

Every change can be undone, with `Ctrl` + `Z` and `Ctrl` + `Y` or the status bar's arrows, one file at a time. The tooltip names what will be undone. Undoing back to the saved state clears the tab's unsaved mark.

5.4. Saving

`Ctrl` + `S` saves the active file; `Ctrl` + `Shift` + `S` saves it under another name. A value you have typed but not yet left is saved too. A tab with unsaved changes shows a dot; closing it, or DbEdit, asks whether to save.

© 2026 Altech Electronics Nordic AB. All rights reserved.

A file DbEdit has not changed is written back exactly as it was read. A changed file is written in the format's standard form, with its comments and attributes kept.

5.5. Copy and paste

`Ctrl` + `C` copies the selected rows, or the right-clicked explorer row, and `Ctrl` + `V` pastes into whichever file is active — the same file, another file of the same format, or another file open in a second DbEdit window. The names are placed on the Windows clipboard as text as well, for pasting into a document.

A paste brings what the objects need:

- A **DBC** message brings its signals, the attribute definitions they use, the sending and receiving ECUs with their attributes, value tables and signal types. A taken identifier or name is renumbered.
- An **LDF** frame brings its signals, their publisher and subscribers as slaves with their node attributes, and their encodings. A schedule table loses the slots the target file cannot send.
- An **A2L** object brings nothing else: its conversion method, record layout and axes are looked up by name in the target file.

The pasted object is selected in the explorer and the editor when the paste is done.

5.6. Search

`Ctrl` + `Shift` + `F` opens the search pane. Type a name, or part of one, and every matching object in every open file is listed with its file and kind. Click a hit to go to it. Wildcards and regular expressions work as in the table filters.

5.7. Problems

© 2026 Altech Electronics Nordic AB. All rights reserved.

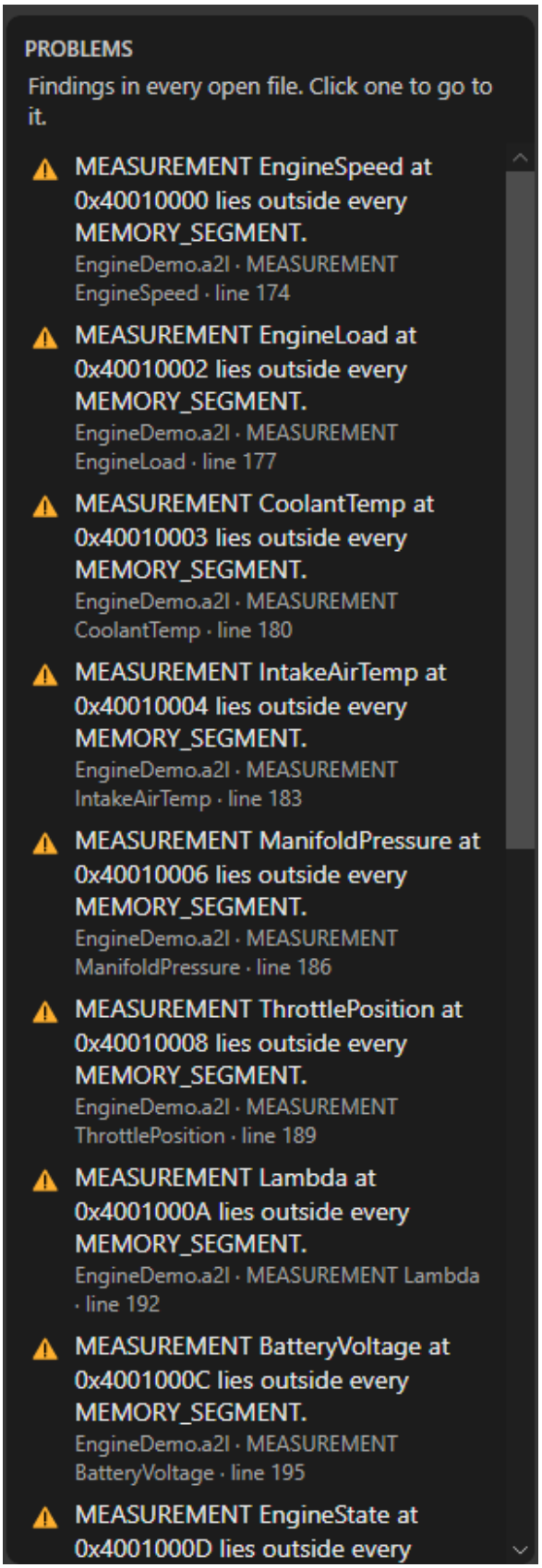

Figure 5. The Problems pane

Each file is checked when it is read and after every change: the parser's own messages first (a line it could not read), then the format's validator — a signal that leaves its message, two messages with one identifier, a frame without a publisher, a measurement outside every memory segment. The status

© 2026 Altech Electronics Nordic AB. All rights reserved.

bar counts errors and warnings across every open file; the pane lists them with the file, the object and the line. Click one to go to the object.

A file that could not be read at all shows the reason in its tab instead of an editor; fix the file and it is reloaded (Section 4.2).

© 2026 Altech Electronics Nordic AB. All rights reserved.

6. CAN Databases (DBC)

The DBC editor covers classic CAN, CAN FD and J1939 databases as CANdb++ and the Kvaser Database Editor do: a scope in the explorer, the messages of that scope in a table, the selected message's signals and bit layout below, and the inspector at the right.

6.1. The database overview

Click the file's own row in the explorer to see the database as a whole.

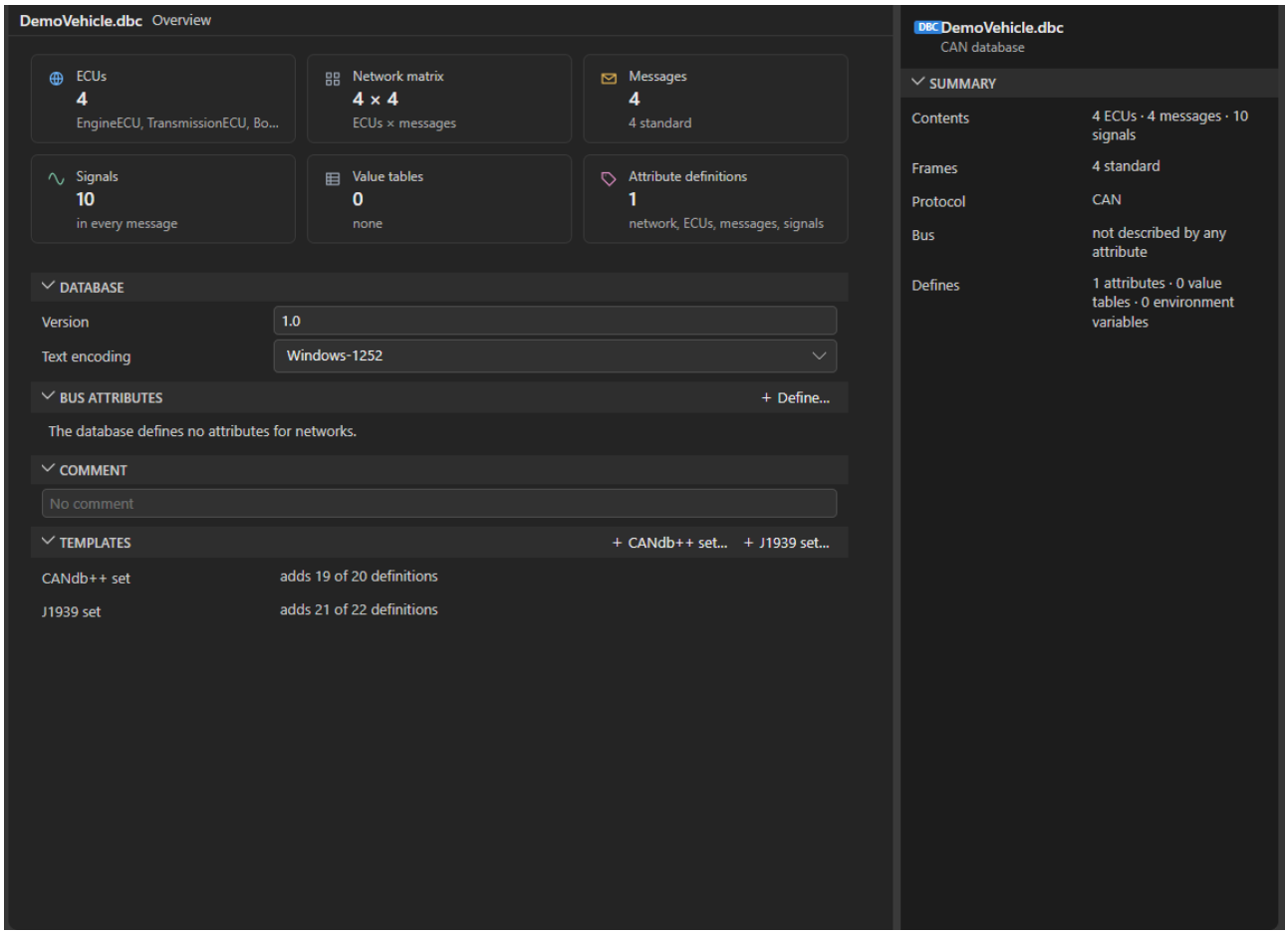

Figure 6. The overview of a CAN database

- **Tiles** — how many ECUs, messages, signals, value tables and attribute definitions the file holds. Click a tile to open that list.
- **Database** — the version string and the text encoding the file is written in.
- **Bus attributes** — every attribute the database defines for the network, such as `BusType` and `Baudrate`. **Define…** adds a new attribute definition.
- **Comment** — the database's comment.
- **Templates** — adds the CANdb++ attribute set (`GenMsgCycleTime`, `GenSigStartValue`, …) or the J1939 set in one step. The dialog lists every definition of the set; one the database already has is greyed. Untick what the database should not get, then **Add**.

© 2026 Altech Electronics Nordic AB. All rights reserved.

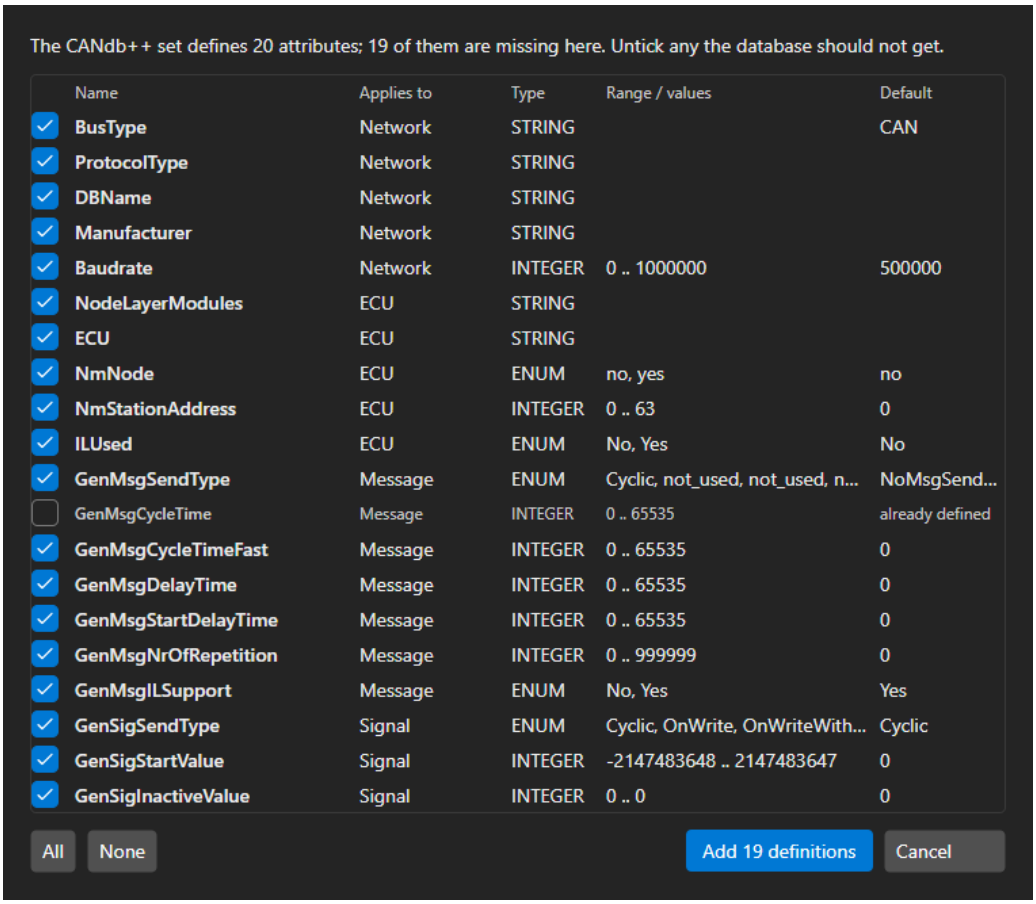

Figure 7. The CANdb++ template

The right pane summarises the network: frames by kind, the protocol and the bus speed, read from the attributes.

6.2. ECUs

The **ECUs** list shows every node with how many messages it sends and receives. Expand an ECU in the explorer for its **Tx messages** and **Rx messages**; picking one of those scopes the message table to it. An ECU's inspector holds its name, comment and attributes.

Add an ECU with the list's + button or the explorer's **Add ECU**; delete one and every message it sent and every receiver entry it had go with it.

6.3. Messages and signals

© 2026 Altech Electronics Nordic AB. All rights reserved.

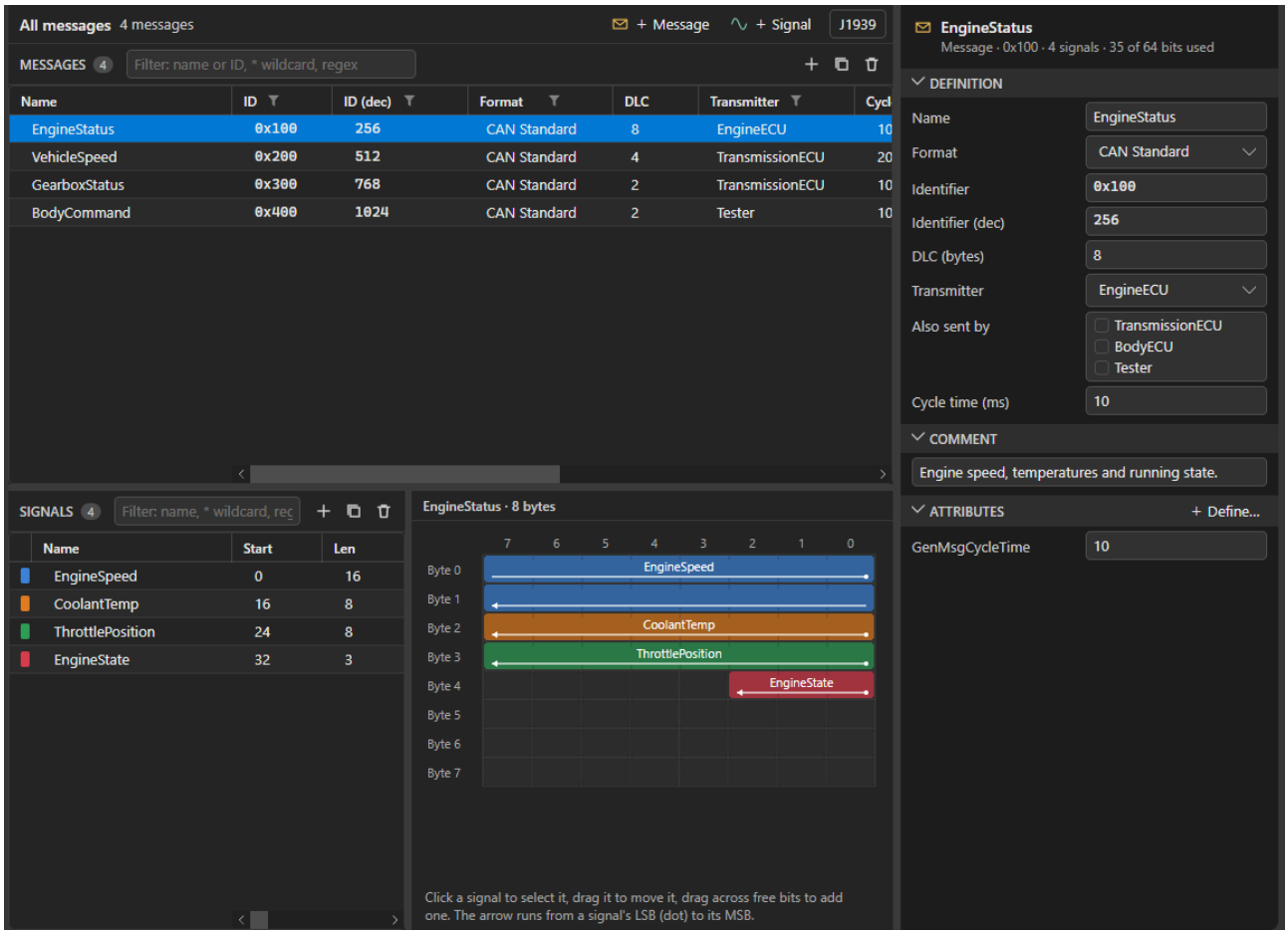

Figure 8. A message, its signals and their layout

The message table shows the name, identifier (hexadecimal and decimal), format, length, transmitter and cycle time of every message in the scope. The **ID**, **Format** and **Transmitter** headers filter by ticked values.

The inspector edits the message:

- **Format** — CAN Standard (11-bit), CAN Extended (29-bit), CAN FD Standard or CAN FD Extended. The format is stored as CANdb++ stores it, in the identifier's extended bit and the `VFrameFormat` attribute, so other tools read it the same way.
- **Identifier** — in hexadecimal or decimal; both fields show the same value.
- **DLC** — 0 to 8 bytes for classic CAN, up to 64 for CAN FD. The bit layout grows with it.
- **Transmitter** and **Also sent by** — the sender, and further senders for a message several ECUs send.
- **Cycle time** — the `GenMsgCycleTime` attribute, when the database defines it.
- **Attributes** — every attribute the database defines for messages. An unset attribute shows its default greyed; clearing a field returns it to the default.

Select a message to see its signals in the lower table: name, start bit, length, byte order, sign, factor, offset, minimum, maximum, unit and receivers. A signal's inspector adds the value type (unsigned,

© 2026 Altech Electronics Nordic AB. All rights reserved.

signed, IEEE float or double), the multiplexing role, value descriptions and the signal's attributes.



The **start bit** is the signal's least significant bit, as CANdb++ and the Kvaser editor show it, for Intel and Motorola signals alike. Changing a Motorola signal's length or byte order keeps that bit where it is.

6.4. The bit layout

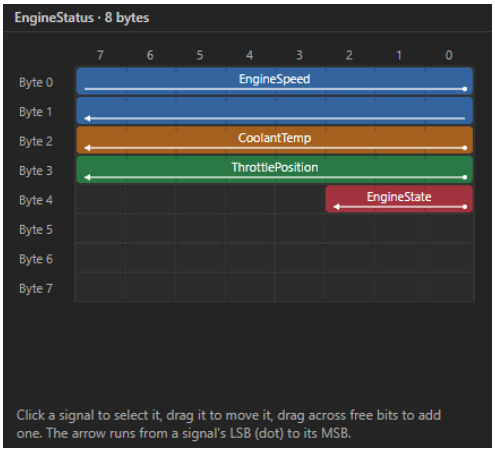

Figure 9. The bit layout of an 8-byte message

The layout draws the message's bytes, one row each, bits 7 to 0 from left to right. Each signal is a block in its own colour; the arrow along its bottom runs from the least significant bit (the dot) to the most significant (the head), so an Intel signal's arrow runs down into the following byte and a Motorola signal's up into the previous one. Two signals that share a bit are drawn red, unless multiplexing keeps them out of the same frame.

- **Click** a signal to select it in the table and the inspector.
- **Drag** a signal to move it. An Intel signal moves along the message's bit numbering, a Motorola signal along its own.
- **Drag across free bits** to add a signal there: a new signal of that length is created and selected, ready to be named.

6.5. Multiplexing

A signal's inspector sets its **multiplexing** role: the message's multiplexor, a signal present only for one multiplexor value, or an ordinary signal. Extended multiplexing — a signal that depends on several values or on another multiplexed signal — is edited as the list of value ranges it is present for.

The bit layout of a multiplexed message has a chooser above it: **All groups**, or one multiplexor value. With **All groups**, a byte used by several groups is drawn as one lane per group, each tagged with its value (`m115`). Choosing a value shows that group and the signals that are always present.

6.6. J1939

For an extended message in a J1939 database the identifier is also edited as its parts: **priority**, **PGN**, **source address** and, for a destination-specific PGN, the **destination address**. Changing one rewrites the 29-bit identifier; typing an identifier updates the parts. The J1939 template (Section 6.1) adds the attributes J1939 tools expect.

6.7. The network matrix

© 2026 Altech Electronics Nordic AB. All rights reserved.

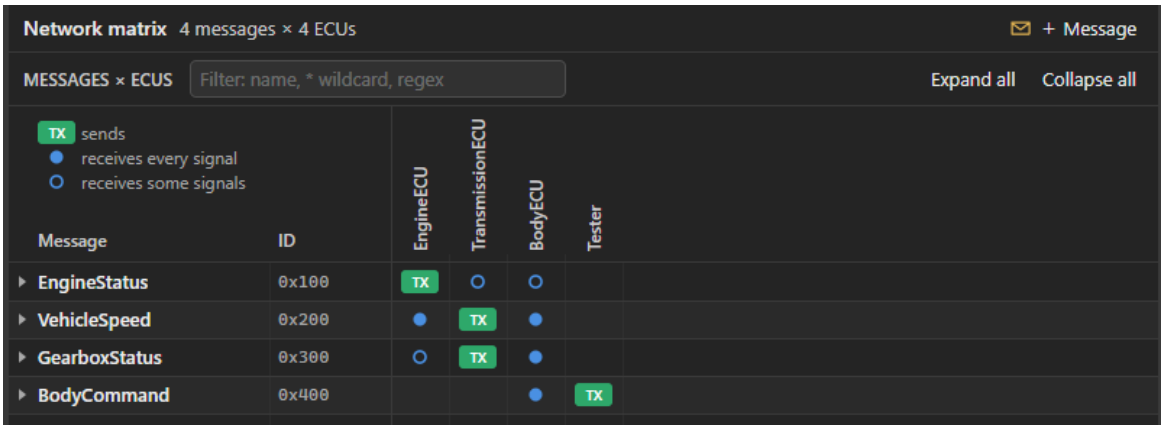

Figure 10. The network matrix

Network matrix in the explorer shows the messages against the ECUs. A green **TX** marks the sender; a filled dot means the ECU receives every signal of the message, a hollow one some of them. Open a message with its chevron (or the `Right` key, or **Expand all**) to see one row per signal with a dot for each receiving ECU.

- **Click** an empty cell to make the ECU receive the message (every signal) or the one signal; click a marked cell to stop.
- **Right-click** a cell to make the ECU the transmitter, or to receive all or nothing.
- Click an ECU header to see the ECU in the inspector; click a row to see the message or signal.

6.8. Value tables and value descriptions

Value tables are named lists of value descriptions a database defines once and signals refer to. Each is edited as its list of value / text pairs. A signal's own value descriptions (the `VAL_` lines) are edited in the signal's inspector in the same way, and a value table can be applied to a signal from there.

6.9. Attribute definitions

Attribute definitions lists every attribute the database defines: its name, what it applies to (network, ECU, message, signal, environment variable, or a relation such as ECU × message), its type (integer, hexadecimal, float, string or enumeration), its range or values and its default. Define a new one with **Define...**, here or in any object's **Attributes** section; delete one and every value set for it goes too.

Relation attributes (CANdb++'s `BA_REL_`), such as an ECU's own cycle time for a message it receives, are shown in the inspector of the message, signal or environment variable they relate to, one row per ECU.

© 2026 Altech Electronics Nordic AB. All rights reserved.

7. LIN Description Files (LDF)

The LDF editor covers LIN 2.0, 2.1 and 2.2A and ISO 17987 description files: the master and its slaves, unconditional, event-triggered, sporadic and diagnostic frames, signals, schedule tables and encoding types. The buttons above the editor add a frame, a slave, a schedule table or an encoding type to the open file.

7.1. Nodes

Nodes lists the master with its timing (time base and jitter) and every slave. A slave's inspector holds its node attributes: protocol version, configured and initial NAD, product identifier (supplier, function, variant), response error signal, fault state signals, the P2 / ST / N_As / N_Cr timings, and the slave's configurable frames with their message identifiers. A slave the file lists without node attributes gets them from the inspector with one click.

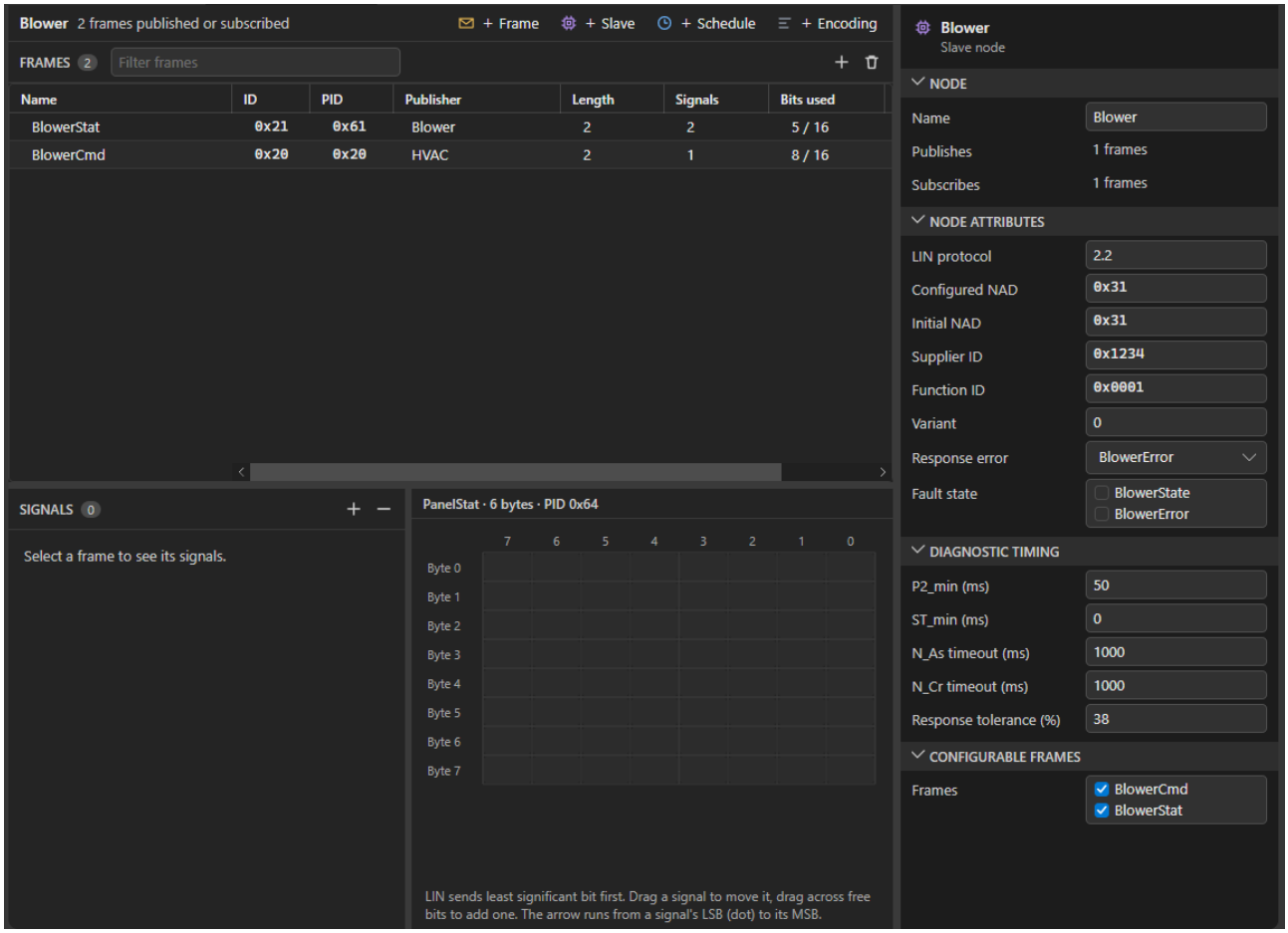

Figure 11. A slave node and its attributes

Expanding a node in the explorer shows the frames it publishes and subscribes to.

7.2. Frames and signals

© 2026 Altech Electronics Nordic AB. All rights reserved.

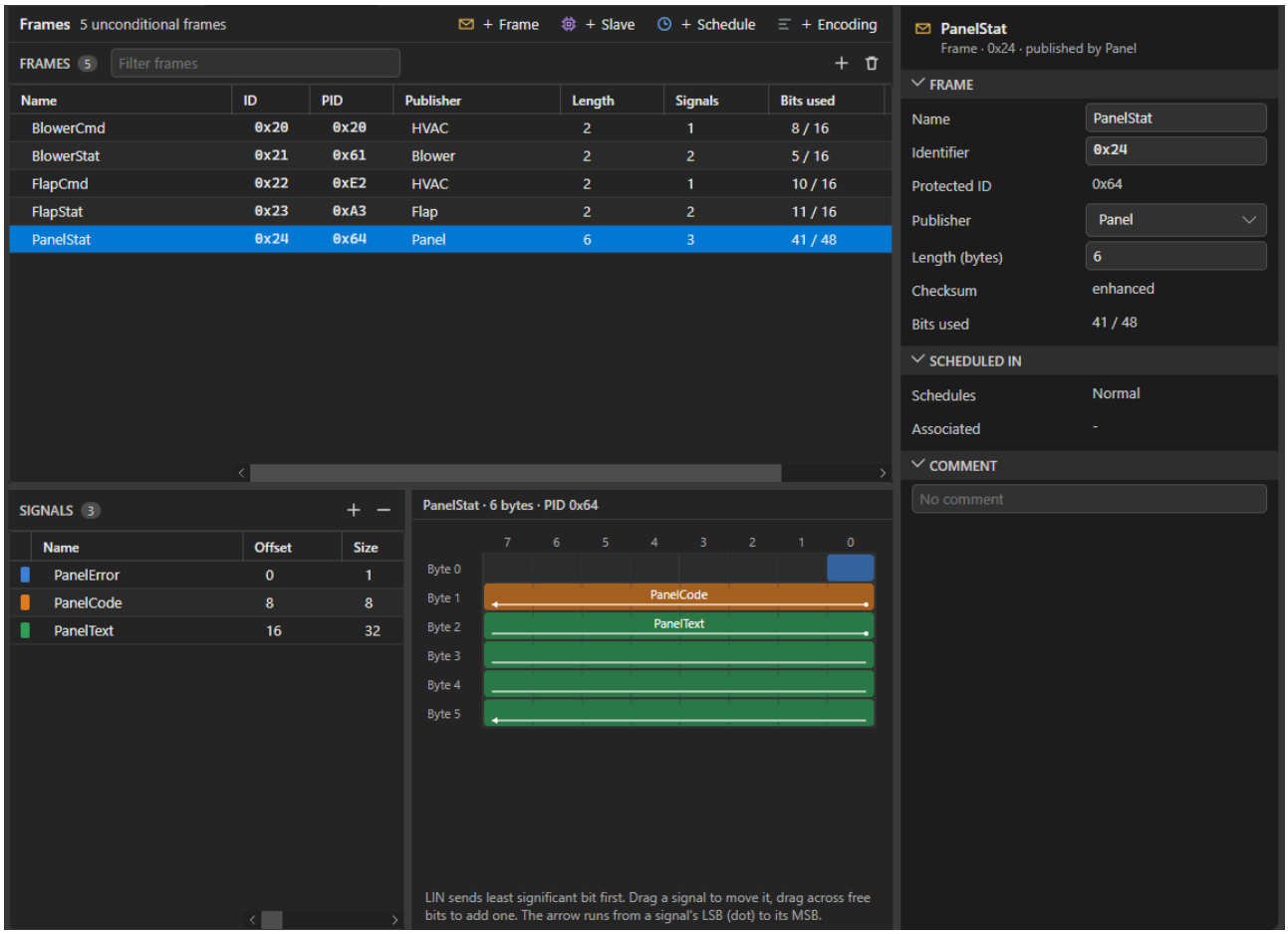

Figure 12. A frame, its signals and their layout

The frame table shows every frame's identifier, publisher and length, with the frame's kind noted for event-triggered and sporadic frames. The inspector edits the identifier (the protected identifier is shown beside it), the publisher and the length; for an event-triggered frame the collision-resolving schedule table and the frames it carries; for a sporadic frame the frames it may send.

Select a frame for its signals: name, offset in the frame, size, initial value, publisher and subscribers. The bit layout below works as for CAN (Section 6.4): click to select, drag to move, drag across free bits to add a signal. LIN signals are little-endian, so every arrow runs towards the following byte.

© 2026 Altech Electronics Nordic AB. All rights reserved.

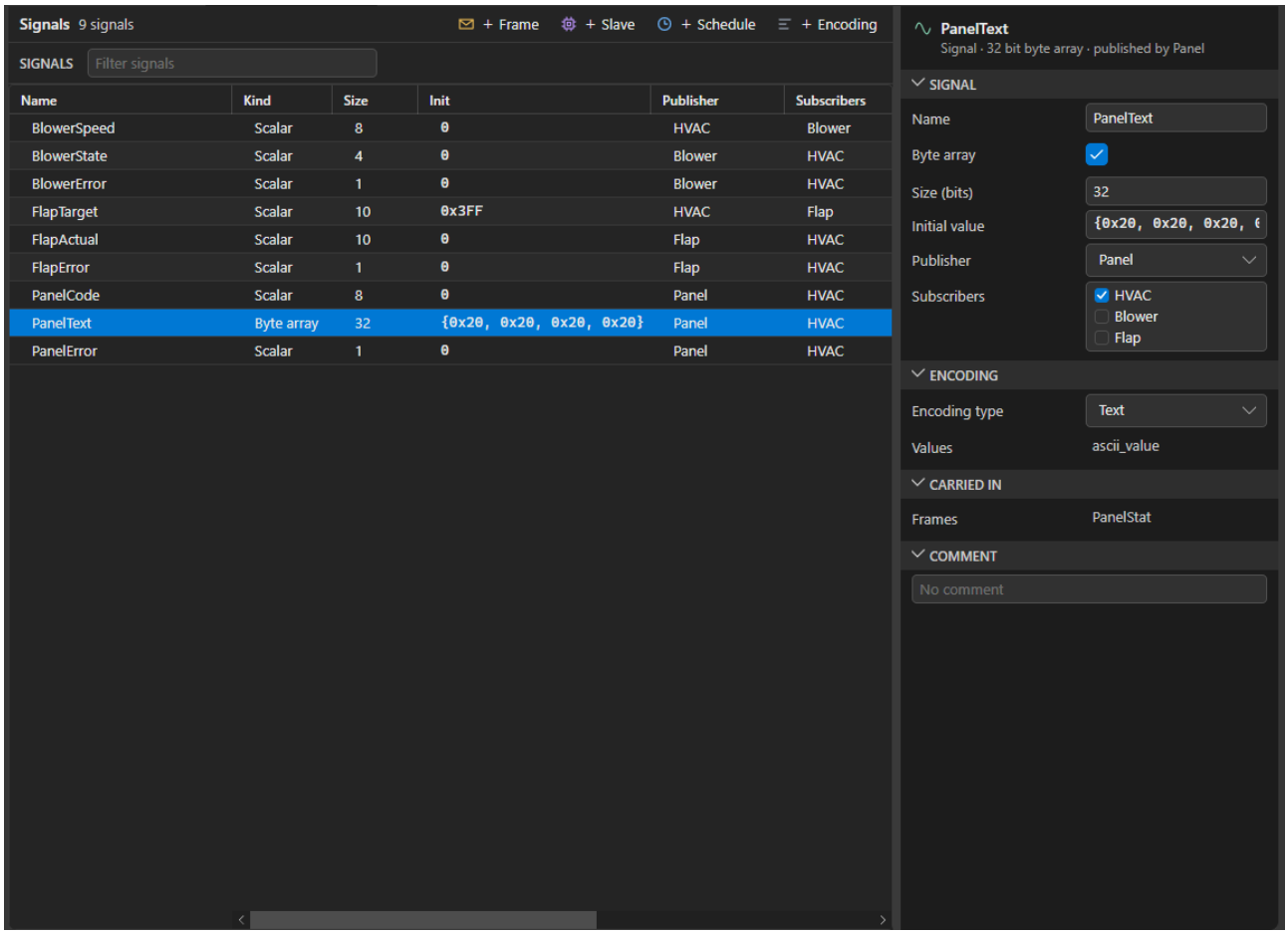

Figure 13. A byte-array signal

A signal's inspector covers its kind (**scalar** of 1 to 16 bits, or a **byte array** of 1 to 8 bytes), size, initial value, publisher, subscribers, the frames that carry it and its encoding type with the encoding's values.

7.3. Schedule tables

© 2026 Altech Electronics Nordic AB. All rights reserved.

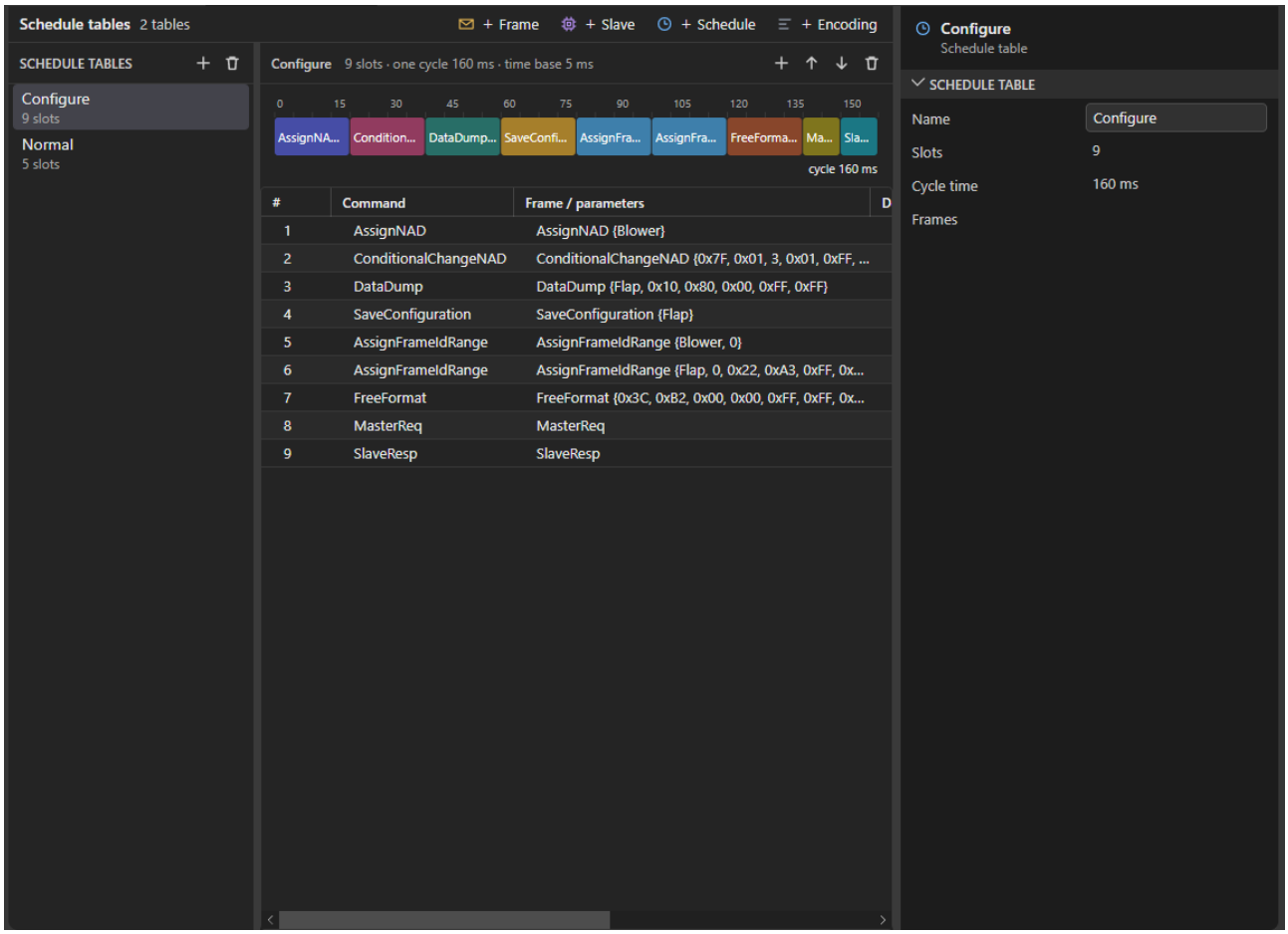

Figure 14. A schedule table and its one-cycle time line

Schedule tables lists the file's tables at the left; the selected table's slots are shown as a numbered list and as a time line of one cycle. Each slot names its command — a frame, or one of the configuration and diagnostic commands (`AssignNAD`, `ConditionalChangeNAD`, `DataDump`, `SaveConfiguration`, `AssignFrameIdRange`, `FreeFormat`, `MasterReq`, `SlaveResp`) with its parameters — and its delay. The table's inspector shows the slot count, the cycle time and the frames it schedules.

- + adds a slot after the selected one; the arrows move it up and down; the bin removes it.
- A slot's delay that is not a multiple of the master's time base is drawn hatched on the time line, since the master cannot schedule it as written.

7.4. Encoding types

Encoding types lists the file's signal encodings. Each is edited as its LDF lines: a `logical_value` per named state, a `physical_value` range with scale, offset and unit, `bcd_value` or `ascii_value`. The signals that use an encoding are listed in its inspector.

© 2026 Altech Electronics Nordic AB. All rights reserved.

8. A2L Files

The A2L editor is driven by the ASAM MCD-2 MC keyword specification rather than by fixed pages, so every keyword, parameter and optional block the standard defines can be edited, and a file of tens of megabytes opens in a second.

8.1. The module in the explorer

A file expands into its header and its module. Under the module come the single blocks — **MOD_PAR**, **MOD_COMMON**, the **IF_DATA** of each interface, **A2ML** — and then one list per kind of object: measurements, characteristics (grouped by type: VALUE, CURVE, MAP, VAL_BLK, …), axis points, conversion methods, verbal tables, record layouts, functions and groups (as the hierarchy their `SUB_FUNCTION` and `SUB_GROUP` lists describe). The count at the right is the list's size.

8.2. Object tables

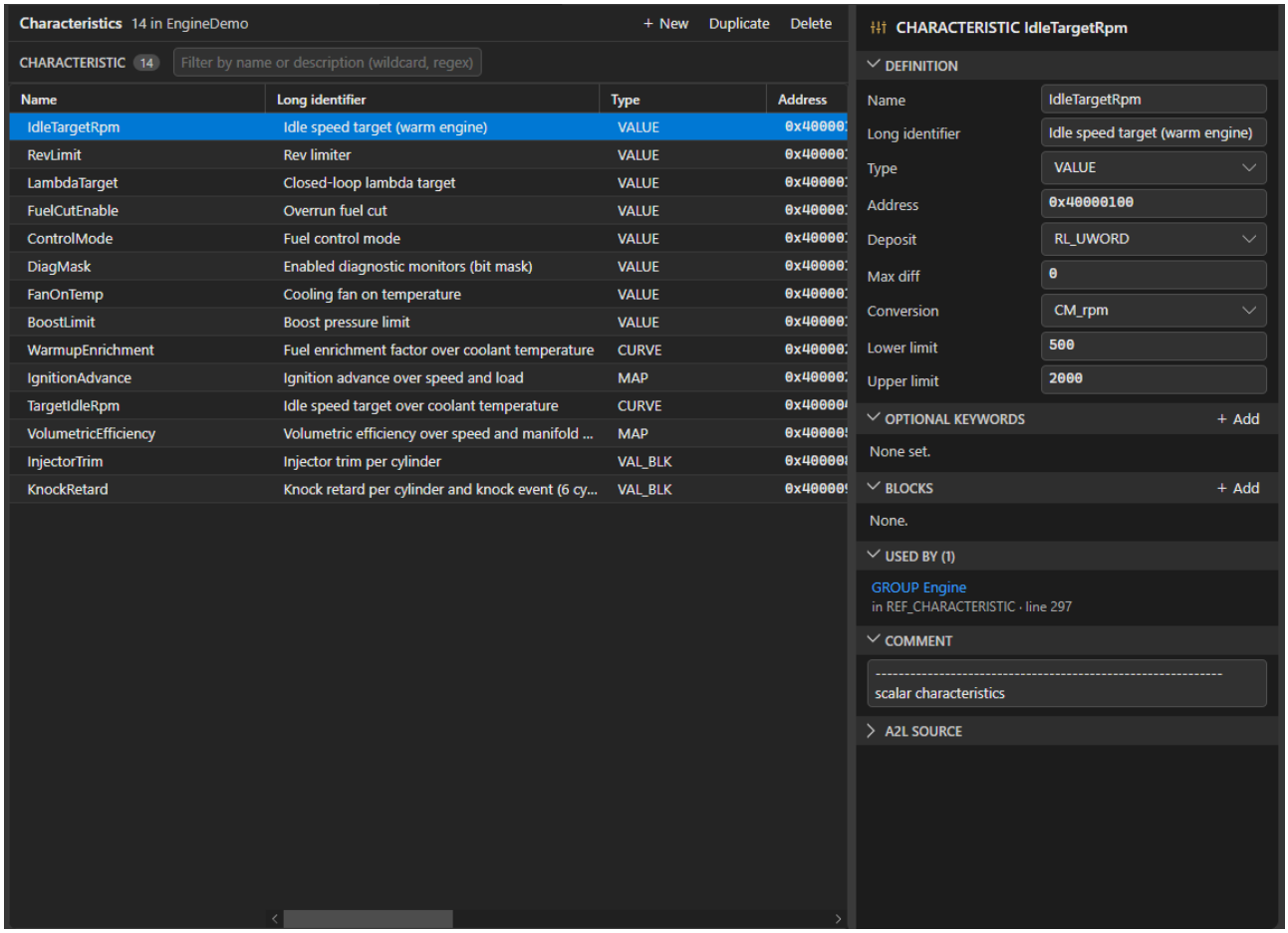

Figure 15. The characteristics of a module

Each list is a table whose columns are the keyword's own parameters: a characteristic's name, long identifier, type, address, record layout, maximum difference, conversion and limits. Every column sorts (Section 5.1); the filter matches the name or the description. **New**, **Duplicate** and **Delete** act on the selected rows.

© 2026 Altech Electronics Nordic AB. All rights reserved.

The inspector shows the selected object:

- **Definition** — the keyword's positional parameters, typed: a reference (a conversion method, a record layout, an axis) is picked from the file's own objects, an enumeration from its values. Renaming an object renames every reference to it.
- **Optional keywords** — the optional entries the standard allows (`FORMAT`, `PHYS_UNIT`, `BIT_MASK`, `DISPLAY_IDENTIFIER`, …). **Add** offers those not yet set; each row has a remove.
- **Blocks** — nested blocks such as a characteristic's `AXIS_DESCR`s, or a function's `DEF_CHARACTERISTIC` list, opened in place with a step back.
- **Used by** — every object that refers to this one, each a link.
- **Comment** — the comment written before the block in the file.
- **A2L source** — the block's text as written, editable as a last resort for something the fields do not reach. A change is parsed back and checked before it is applied.

8.3. Block pages

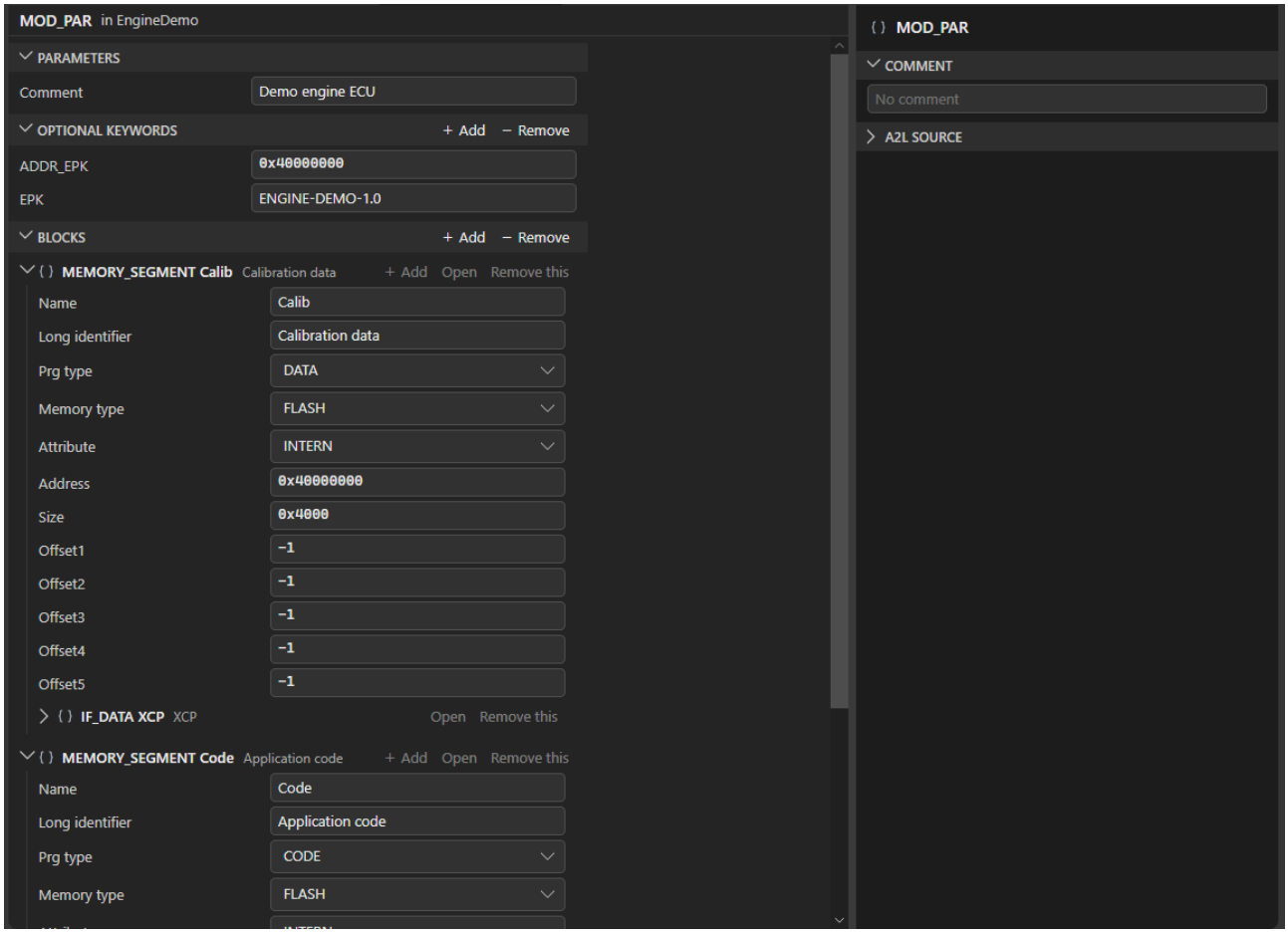

Figure 16. MOD_PAR as a page

A single block picked in the explorer — MOD_PAR, MOD_COMMON, an IF_DATA, the A2ML — fills the middle of the window as a page, with its comment and source at the right. The page lists the block's parameters, its optional keywords and its blocks (a MOD_PAR's memory segments, for example) as

© 2026 Altech Electronics Nordic AB. All rights reserved.

nested groups that fold; a folded group shows a summary of what it holds.

8.4. IF_DATA

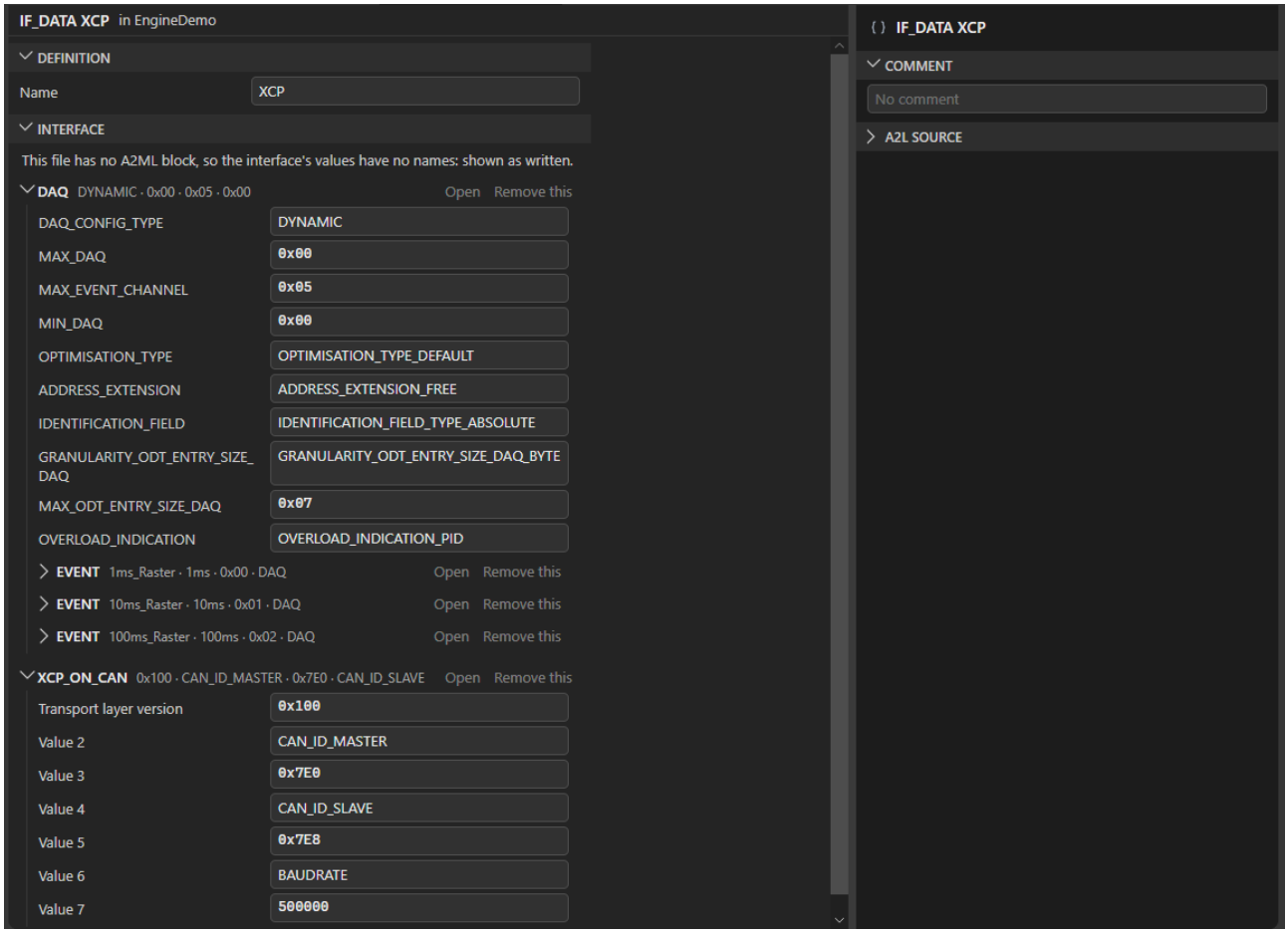

Figure 17. An XCP IF_DATA read through the file's A2ML

An interface's IF_DATA is read through the A2ML the file carries, so every value has its place and name: the XCP protocol layer's timeouts and commands, the DAQ lists and events, the transport layer's identifiers. A repeated flag (the optional commands) is a checklist; an item the A2ML allows but the file lacks is offered under a group's **Add**. Where the ASAM standard names a place, that name is used; a file with no A2ML shows the block as written, value by value.

A module with several IF_DATAs has an **interfaces** page: a tile per interface, each opening that interface's page.

8.5. Large files

Tables of tens of thousands of objects filter and sort without delay, and an edit that changes a value takes a fraction of a second. The validator runs after edits settle and lists an object at an address outside every memory segment, a reference to an object that does not exist, and the other checks the standard allows.

A file DbEdit has not changed is saved byte for byte.

© 2026 Altech Electronics Nordic AB. All rights reserved.

9. Licence Agreement

This is the agreement that comes with DbEdit, version 1.0, 9 November 2026. The installer shows it, and the buttons at the foot of the settings popup open it and the third-party notices.

This agreement is between you and Altech Electronics Nordic AB, Swedish company registration number 559438-6491 ("Altech"). If you use DbEdit for an organisation, you accept it on behalf of that organisation. By installing or using DbEdit you accept this agreement. If you do not accept it, do not install or use DbEdit.

1. Licence

Altech gives you a free, non-exclusive, non-transferable licence to install and use DbEdit, on any number of computers, for private and commercial purposes. No fee, licence key or account is required.

2. What you may share

Within your own organisation, you may copy the unmodified installer and deploy it to your organisation’s computers. To give DbEdit to anyone else, point them to Altech’s official download page.

3. Restrictions

You may not:

- sell, rent, lease or charge for DbEdit;
- publish or distribute DbEdit other than as section 2 allows;
- modify DbEdit, or remove its copyright or licence notices;
- decompile or reverse engineer DbEdit, except where mandatory law gives you the right to do so.

4. Ownership

DbEdit is protected by copyright and is licensed, not sold. Altech and its licensors keep all rights not expressly granted here. Your description files, and the files you create or edit with DbEdit, remain yours. Altech claims no rights in them.

5. Third-party components

DbEdit includes open-source components under their own licences. They are listed in the file DbEdit-Third-Party-Notices.txt, installed with DbEdit. Those licences apply to those components and take precedence over this agreement for them.

© 2026 Altech Electronics Nordic AB. All rights reserved.

6. Your files and the update check

DbEdit does not send your files or their contents anywhere. When it starts, DbEdit asks GitHub whether a newer version has been published, and it downloads an installer from GitHub only if you choose to update. GitHub receives your IP address, as with any web request, and handles it under its own terms.

7. Check your results

DBC, LDF and A2L files can define how vehicles and control units behave. You are responsible for reviewing and testing every file you create or change before you use it, particularly in a vehicle, on a test bench or in any safety-related system. DbEdit is an editing tool, is not qualified or certified for functional safety, and does not replace your own verification.

8. No warranty

DbEdit is provided free of charge and "as is". To the extent the law allows, Altech gives no warranty of any kind, express or implied, including that DbEdit is free of errors, that it suits a particular purpose, or that it will correctly read, edit or save any file. Keep backups of your files.

9. Limitation of liability

To the extent the law allows, Altech is not liable for loss of data, loss of profit, production stoppage, damage to vehicles or equipment, or any indirect or consequential loss arising from DbEdit. Altech’s total liability under this agreement is limited to SEK 1,000. Nothing in this agreement limits liability for intent or gross negligence, or liability that cannot be limited by law, and nothing in it affects rights you have as a consumer under mandatory law.

10. Updates and availability

Altech may change DbEdit, publish new versions with new licence terms, or stop offering DbEdit at any time, and has no obligation to provide support or updates. The terms that came with a version continue to apply to that version.

11. Termination

This licence ends automatically if you breach this agreement. When it ends, you must stop using DbEdit and delete your copies. You may end it at any time by doing the same. Sections 4, 7, 8, 9 and 12 continue to apply.

12. Governing law and disputes

Swedish law governs this agreement, without regard to its conflict-of-law rules. Disputes are decided by the courts of Sweden, except where mandatory consumer law gives you the right to bring a claim elsewhere.

© 2026 Altech Electronics Nordic AB. All rights reserved.

13. Contact

Altech Electronics Nordic AB, info@altechelectronics.io

© 2026 Altech Electronics Nordic AB. All rights reserved.