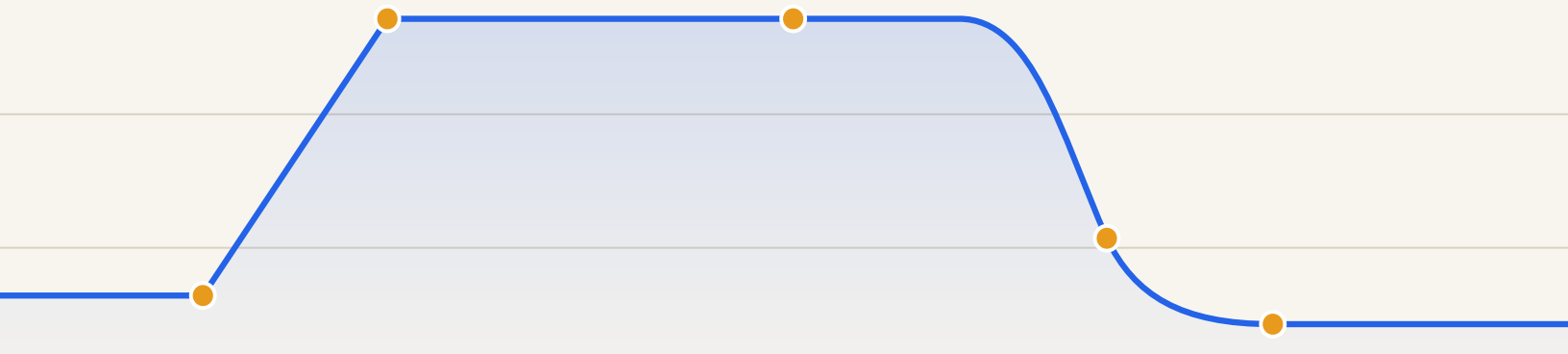
CAN SIGNAL SIMULATOR · WINDOWS

simCAN

simulated signals. real confidence.

User Guide

The complete guide to building, validating, transmitting, recording, and replaying CAN signal scenarios — so they're **proven at the bench before you reach the proving grounds**.



EDITION
Launch / Prerelease

REVISED
May 28, 2026

PLATFORM
Windows

DOWNLOAD & SUPPORT
simulatedsignals.com

About this guide

simCAN — CAN Signal Simulator for Windows · Launch / prerelease edition

simCAN is a Windows CAN signal workbench for exercising CAN signal behavior at the bench before the proving grounds — so scheduled vehicle and track time is spent running the test, not debugging signal generation. This guide walks through everything from installation and licensing to authoring scenarios, transmitting to live hardware, recording real traffic, and replaying it — with the same caution simCAN is built around: **simulate first, validate, then go live.**

SAFETY FIRST — READ BEFORE CONNECTING HARDWARE

simCAN is a bench-validation, simulation, and development tool. It is **not** a safety-certification tool and does not replace professional engineering review, road-test safety practices, vehicle-integration review, regulatory compliance, or independent validation. Use it on an isolated bench or controlled test bus unless you have explicit approval to connect to a vehicle network. **Dry run is on by default** — keep it on until your bus settings and scenario are correct.

How to read this guide

Interface elements are styled so you can scan instructions quickly:

Buttons & fields appear like this.

Menu paths: › e.g. File › Export Excel Scenario

`Ctrl` + `Z` keyboard shortcuts.

`COM5@2000000` literal values you type.

NOTE

Useful context or behavior worth knowing.

TIP

A faster or safer way to do something.

CAUTION

Something that can affect hardware or test validity.

KEY IDEA

A concept that recurs throughout simCAN.

Contents

PART ONE

Get started

What simCAN does, the vocabulary it uses, and how to install, license, and open the workbench for the first time.

01 What simCAN is

simCAN turns a DBC (or a handful of manually defined signals) into time-based signal profiles, plays them onto a bench CAN bus as scheduled frames, and measures how well your laptop, adapter, and bus actually keep up.

In practical terms, simCAN lets an engineer or technician do all of the following from one window:

- Load a **DBC** file and see the messages and signals it defines.
- Build time-based **signal profiles** for those signals and export them to an **Excel scenario** workbook.
- Use Excel — or an LLM — to author realistic scenarios, then import the finished workbook back in.
- Validate a scenario before anything reaches hardware.
- Transmit enabled CAN frames to one or more **isolated** CAN buses.
- Record real incoming CAN traffic to a timestamped log, and decode it into editable profiles when a DBC is available.
- Replay recorded logs back onto a bench bus at their original timing.
- Benchmark the adapter / laptop / driver path at fixed output rates.

Typical uses

simCAN is built for bench validation, simulation, development, and pre-proving-ground confidence building — rehearsing and proving a scenario in the lab so limited, costly proving-ground time isn't lost to signal or hardware problems. For example, exercising ADAS and vehicle-network signal behavior at the bench (AEB, LKAS, ACC, FCW, sensor dropout, lane departure, cut-in and cut-out), driving a downstream logger, HIL setup, ECU, display, gateway, or measurement system with repeatable frames, converting DBC channels into an editable scenario, capturing and replaying real traffic, and checking whether a specific laptop, adapter, driver, USB path, and bus load can sustain a target transmit rate.

> **WHAT SIMCAN IS NOT**
>
> simCAN is not a safety-certification tool. It does not replace professional engineering review, road-test safety practices, vehicle-integration review, regulatory compliance, or independent validation. Connect generated traffic to a real vehicle network only when the IDs, scaling, rates, and bus topology are controlled and approved.

02 Core concepts

A short glossary. These terms recur throughout simCAN and the rest of this guide.

CAN · CAN frame · CAN signal

CAN (Controller Area Network) is the vehicle/bench bus where frames are sent by arbitration ID. A *frame* is a packet with an arbitration ID, data bytes, flags, and timing. A *signal* is a decoded engineering value packed inside a frame — vehicle speed, yaw rate, longitudinal acceleration, brake command, steering angle, object range, lane offset, status flags, and so on.

DBC

A CAN database file. It defines message names, arbitration IDs, signals, bit positions, scaling, offsets, units, signedness, and limits. Loading a DBC is the safest way to ensure simCAN packs signal values correctly.

Scenario & signal profile

A scenario is a planned set of signal values over time. Each signal has a *profile* — a time/value curve with a duration, output frequency, Y-axis range, and points. simCAN interpolates between points to produce sampled output values.

Output Hz

How often simCAN schedules a signal/frame for output. If several signals share the same arbitration ID, they are packed into the same frame and the frame follows the highest needed rate for that group.

> **KEY IDEA — DRY RUN VS. LIVE CAN**
>
> **Dry run** is a safe simulation mode and is **enabled by default**. In dry run, simCAN calculates frames and updates the UI and console but never opens the real adapter or puts frames on the bus. **Live CAN** is dry run unchecked: simCAN opens the configured adapter and sends or receives real frames.

Record Mode

The mode for capturing real incoming frames instead of transmitting a planned scenario.

Playback

Reads a recorded JSONL log and sends those frames back onto a real bus at original relative timing.

Isolated CAN bus

A physically separate CAN network. Two isolated buses require two adapter channels — either two USB-to-CAN adapters or one multi-channel adapter. A USB hub does not merge CAN networks; it only provides more USB ports.

Machine-bound license

A signed license valid for one Windows machine, installed and checked before the workbench opens. A free 7-day trial license can be started from the splash; a purchased key installs a permanent license for that machine.

03 Package & system requirements

What's in the Windows package

The customer download is `simCAN_windows.zip`. It normally includes:

File	Purpose
`simCAN.exe`	The Windows executable.
`simCAN_User_Guide.pdf`	This guide.
`README-FIRST.txt`	Short installation and verification notes.
`SHA256SUMS.txt`	Hashes for verifying downloads against the published build.

> **CAUTION**
> Do not run `simCAN.exe` from inside the ZIP viewer. Extract the ZIP first.

System requirements

Area	Requirement
Operating system	Windows is the primary supported target for the packaged EXE. A Python source version exists, but normal customers should use the packaged Windows ZIP.
PC	A Windows laptop or bench PC.
Adapters	One or more USB-to-CAN adapters. For multiple isolated buses, one CAN channel per bus.
Bus	A properly wired and correctly terminated CAN bench bus.

Supported adapters & backends

Backend	Channel form	Notes
slcan	`COM5@2000000`	Serial CANable-style adapters that appear as a COM port.
gs_usb	`0`, `1`, `2` …	candleLight / GS-USB / CANable 2.0 WinUSB devices. These do not create COM ports unless flashed to slcan firmware.
socketcan	`can0`	Mainly Linux / source workflows.
pcan, vector, kvaser	vendor-specific	Available if the matching drivers and python-can support are present.

> **DSD TECH SH-C31A / CANABLE 2.0**
> If the adapter appears as a COM port, start with `slcan`. If it appears as `canable2 gs_usb` / candleLight / GS-USB, use `gs_usb` with channel `0` for the first adapter, `1` for the second, and so on. If Device Manager shows the device under *Other devices* or with a warning icon, use the `Driver Setup` helper for WinUSB/Zadig guidance (§25).

Bus requirements

- The bitrate in simCAN must match the physical bus.
- Classic CAN devices should leave `CAN FD` unchecked; enable it only when the bus and adapter actually support CAN FD.
- Each physical CAN segment needs correct termination.

- High output rates depend on host timing, adapter throughput, driver behavior, USB latency, CAN arbitration, and bus load.

04 Install & verify

Download from the official site, extract, and — for the launch build — verify the file hash before running.

Install from the customer ZIP

1. Open the official download link from your purchase email or the checkout-completion page.
2. Download `simCAN_windows.zip` and save it to a normal local folder (Documents, Desktop, or Downloads).
3. Right-click the ZIP and choose **Extract All**; pick a location and finish extraction.
4. Open the extracted simCAN folder and confirm `simCAN.exe` is visible.
5. Optional but recommended: open `README-FIRST.txt`, and compare hashes using the download page or `SHA256SUMS.txt`.

> **CAUTION**
> Do not run `simCAN.exe` directly from inside the compressed ZIP window. If Windows Explorer still shows a *Compressed Folder* path, go back and extract first.

Windows security warning & hash verification

The current launch build may show an unknown-publisher or SmartScreen warning while paid public code signing is pending. This does not by itself mean the file is wrong — it means Windows does not yet trust the publisher certificate chain for this build. Verify the download instead:

1. Download only from `https://simulatedsignals.com`, using the ZIP from the official download page.
2. Open PowerShell in the extracted simCAN folder.
3. Run: `Get-FileHash .\simCAN.exe -Algorithm SHA256`
4. Compare the returned hash to the value on the official download page or in `SHA256SUMS.txt`.
5. If the hash matches and the file came from the official site, proceed. **If it does not match, do not run the file.**

05 First launch & license activation

simCAN runs once it has a valid, machine-bound license — either a **free 7-day trial** you start on the spot, or a **purchased license** you activate with a key. Both unlock the full workbench; the trial is simply time-limited.

License splash — shown when no license is installed. Note the Machine ID and the six actions.

When no license is installed, simCAN opens the license splash. It shows the simCAN wordmark, a status line (such as *No simCAN license file found*), this computer's **Machine ID**, and six buttons:

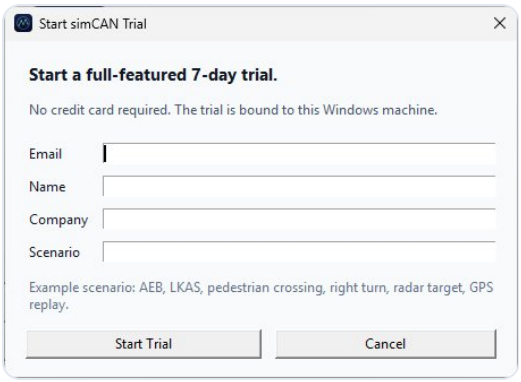
- **Start 7-Day Trial** — begin a free, full-featured trial bound to this machine.
- **Buy License** — open the purchase page to get an activation key.
- **Activate Online** — redeem a key and install the signed license.
- **Load License** — install a license file you were sent.
- **Copy Machine ID** — copy this machine's ID for support or manual licensing.
- **Exit** — close simCAN.

TRIAL, THEN BUY, THEN ACTIVATE

A typical first run is: **Start 7-Day Trial** to evaluate simCAN immediately, then **Buy License** from the same splash when you're ready, and finally **Activate Online** with the key from your purchase email. Each step is below.

Start a free 7-day trial

1. On the splash, click **Start 7-Day Trial**.
2. In the **Start simCAN Trial** dialog, enter your **Email**, **Name**, and **Company**, plus an optional **Scenario** of interest (for example AEB, LKAS, pedestrian crossing, right turn, radar target, or GPS replay).
3. Click **Start Trial**. No credit card is required, and the trial is bound to this Windows machine.
4. The main workbench opens for seven days with every feature enabled.



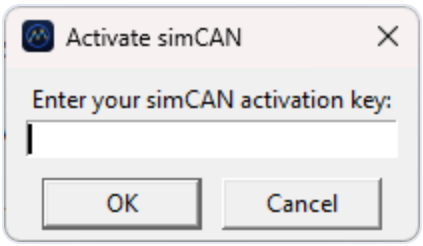
Trial dialog — a full-featured 7-day trial, no credit card, bound to this machine.

Buy a license

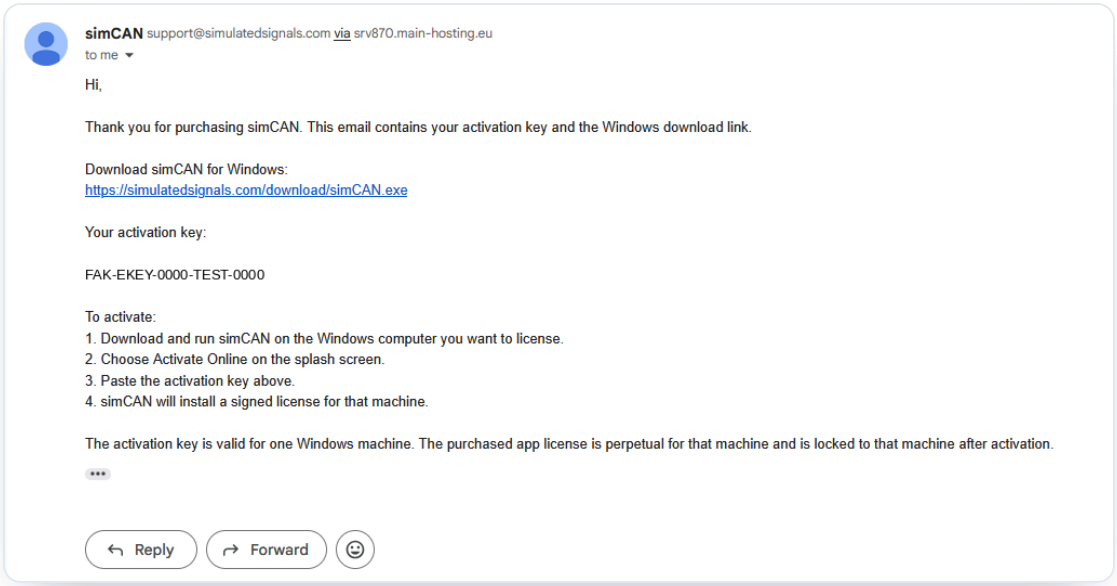
When the trial ends — or whenever you're ready to buy — click **Buy License** on the splash. It opens the simCAN purchase page at simulatedsignals.com. Complete checkout and you'll receive a purchase email with your **activation key** and the Windows download link. Then continue with online activation below; there is no reinstall, and an active trial keeps running until you activate the purchased key.

Online activation

1. Make sure the computer is connected to the internet.
2. On the splash, click **Activate Online**.
3. Copy the activation key from your purchase email.
4. Paste it into the **Activate simCAN** dialog and click **OK**.
5. simCAN contacts the activation server, which binds the key to this Machine ID, then installs the signed license for the current Windows user.
6. The main workbench opens — now licensed permanently for this machine.



Activation dialog



The purchase email contains the Windows download link and your activation key. Save it — you may need the key and order details for support.

Manual license fallback

If online activation is not available, click **Copy Machine ID**, send the ID to the license issuer, receive a machine-specific license file, then click **Load License** and select it. If the file is valid for the machine, simCAN opens.

Where licenses live

The installed license is stored at `%LOCALAPPDATA%\simCAN\simcan.license`. simCAN searches for a license next to `simCAN.exe`, then in `%PROGRAMDATA%\simCAN\`, then `%LOCALAPPDATA%\simCAN\`, and finally a path set in the `SIMCAN_LICENSE_FILE` environment variable.

ACTIVATION KEY ≠ LICENSE

The activation key is used to *fetch* the signed license; the installed license is what unlocks the app and is bound to one machine. The trial installs its own time-limited license, so when it expires simCAN returns to the splash — activate a purchased key to continue. Keep your purchase email and key for support.

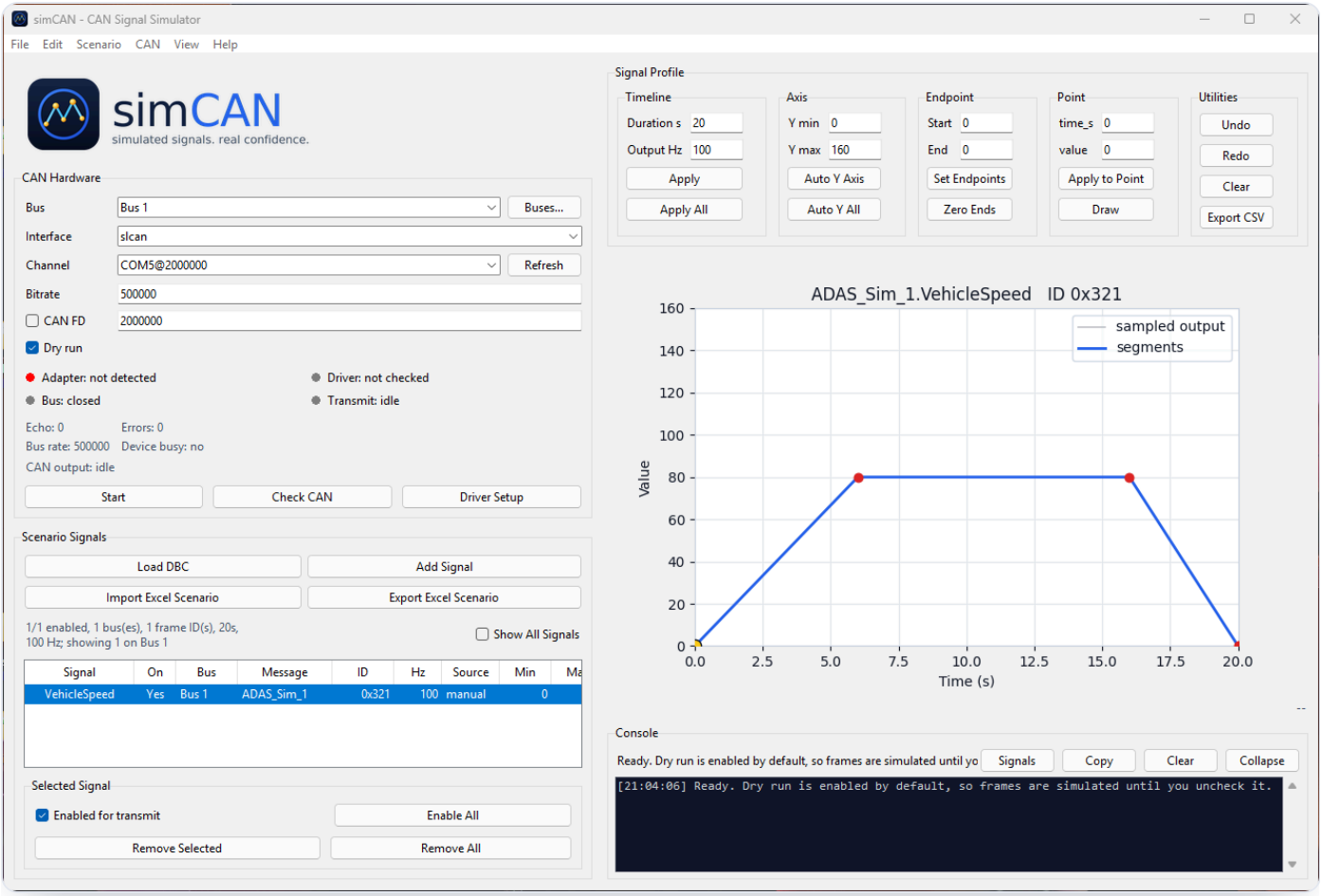
PART TWO

The workbench

A panel-by-panel and menu-by-menu reference for everything in the simCAN main window.

06 Main window tour

After activation, simCAN opens the main workbench. The left column configures hardware and signals; the right column shapes the selected signal’s profile, plots it, and reports status.



The simCAN workbench. Left: CAN Hardware, Scenario Signals, Selected Signal. Right: Signal Profile, the plot, and the Console.

Area	What it does
Menu bar	Top of the window: `File`, `Edit`, `Scenario`, `CAN`, `View`, `Help`.
CAN Hardware	Left. Select bus, interface, channel, bitrate, CAN FD, dry run, and the live CAN actions.
Scenario Signals	Below hardware. Load DBC files, add signals, import/export Excel scenarios, filter the list, and select a signal.
Selected Signal	Enable/disable the selected signal, enable the visible list, or remove signals.
Signal Profile	Top-right. Edit duration, frequency, axis, endpoints, individual points, draw mode, undo/redo, clear, and CSV export.
Plot	Middle-right. The selected signal's profile, sampled output, segment line, points, and the live record trace.
Cursor readout	Below the plot at right: the current cursor time/value as you move over the plot.
Console	Bottom-right. Status, warnings, errors, timing metrics, and quick buttons.
Signal Viewer	A sortable table of live/idle values, opened from the `Signals` button or the `View` menu.

> **TIP**
> The window is resizable, and a draggable divider separates the left and right panes — drag it for more plot width or more signal-list width. Collapse the console when you need more vertical plot space.

07 Menu reference

The menu bar holds less-frequently used commands. Many also have a button on a panel; those are noted below.

File

Command	Action
Load DBC...	Pick a `.dbc` file; it loads onto the currently selected bus. Each DBC signal becomes a simCAN signal with source `dbc`; packing is preserved while the DBC path stays valid.
Record Mode	Toggles between scenario transmit and capture. The main button becomes `Start Record` and the title changes to *simCAN — CAN Recorder*; some editing/transmit actions are disabled. (§22)
Import Excel Scenario...	Imports Signals, Points, AppSettings, and Buses into the bus selected in CAN Hardware. Replaces signals only on that destination bus; other buses and local adapter settings are preserved.
Export Excel Scenario...	Saves the current scenario as `.xlsx` — signal definitions, bus assignment, enabled flags, points, app settings, bus settings, and notes. Do this before editing in Excel or with an LLM.
Export Selected CSV...	Exports sampled `time_s,value` points for the selected signal.
Copy LLM Excel Prompt	Copies the built-in LLM prompt to the clipboard for use with an exported workbook. (§18)
Exit	Closes simCAN.

Edit

Command	Action
Undo Plot Edit `Ctrl`+`Z`	Undoes the last profile edit for the selected signal (draw, drag, endpoints, clear, direct point changes).
Redo Plot Edit `Ctrl`+`Y`	Reapplies the last undone edit.
Clear Selected Profile	Resets the selected signal to a flat zero profile. Use Undo if accidental.

Scenario

Command	Action
Add Manual Signal...	Opens the Add Signal dialog for signals with no DBC entry. (§11)

Command	Action
Enable All Signals	Enables all *visible* signals. With `Show All Signals` off, only the selected bus; with it on, every visible bus.
Remove Selected Signal	Removes only the selected signal.
Remove All Signals	Removes every *visible* signal — selected bus only, or across buses if Show All Signals is on.
Auto Y Axis for Selected / All	Fits the Y-axis around current points for the selected signal, or for every signal. Useful after import, record conversion, or LLM-generated profiles.

CAN

Command	Action
Manage CAN Buses…	Add, rename, remove, or select isolated bus configurations. (§9)
Validate Scenario	Runs preflight checks and writes errors/warnings to the console. Start Transmit also validates first. (§19)
Check CAN	Hardware connection check on the selected bus. Unless dry run is checked, it opens the bus, sends a test frame (ID `0x321`), drains echo/error frames, and updates status. (§20)
Built-in CAN Benchmark…	Requires dry run unchecked. Sends a fixed frame at 50/100/200/500 Hz and reports output health and timing. (§21)
Start / Stop Transmit	Same as the main panel button in scenario mode.
Reset CAN	Stops transmit and logs a reset. If the adapter is still stuck, unplug and replug it.
Playback Log… / Stop Playback	Requires dry run unchecked. Replays a `.jsonl` recording onto the selected bus at original timing. (§23)

View & Help

Command	Action
View › Dark Theme	Toggles light/dark theming across panels, fields, tables, plot, and console. (§26)
View › Signal Viewer	Opens the Signal Viewer (same as the console's `Signals` button).
View › Toggle Console	Collapses or expands the console for more plot space.
Help › simCAN Help…	Opens the built-in help window summarizing scenario mode, timing, benchmark, LLM workflow, multi-bus, record, playback, licensing, the Signal Viewer, plot tips, and hardware.
Help › Driver Setup	Opens the Windows CAN Adapter Setup Helper. (§25)
Help › About simCAN	Shows basic app information.

08 CAN Hardware panel

This panel controls the active physical bus and the main live CAN action. Hardware fields always edit the currently selected bus.

Fields & controls

Control	Notes
Bus	Selects the active isolated bus. Loading a DBC or adding a manual signal assigns it to this bus. Switching the bus also changes which signals are visible when Show All Signals is off.
Buses...	Opens `Manage CAN Buses`.
Interface	The python-can backend: `slcan`, `gs_usb`, `socketcan`, `virtual`, `pcan`, `vector`, `kvaser`. Use slcan for COM-port adapters, gs_usb for candleLight/GS-USB/CANable 2.0 WinUSB devices.
Channel	slcan: a COM string such as `COM5@2000000`. gs_usb: a number such as `0` or `1`. socketcan: a name such as `can0`.
Refresh	Rescans COM ports and GS-USB devices, updates the channel list, and logs detected devices. May suggest gs_usb channels.
Bitrate	Nominal bitrate in bits/s (commonly `500000`, `250000`, `1000000`). Must match the target bus.
CAN FD / Data bitrate	Enable CAN FD only if the adapter and bus support it. The data-bitrate field is the CAN FD data-phase rate and is ignored when CAN FD is off.
Dry run	**Checked by default.** When checked, simCAN does not open the adapter. Keep it checked while building, importing, editing, or testing; uncheck only when bus settings and scenario content are correct.

Status indicators

Indicator	Meaning
Adapter	Detection status. `RED` not detected/failed, orange checking/partial, `GREEN` ready.
Driver	Backend status. For slcan, serial driver is assumed once the COM port exists; for gs_usb the WinUSB/libusb backend must be available.
Bus	`closed`, `opening`, `ready`, `open`, `dry run`, `recording`, `playback`, or `not ready`.
Transmit	`idle`, `starting`, `dry run`, `live`, `benchmark`, or `stopping`.
Echo / Errors	Echo count and detected error-frame count, when available.
Bus rate / Device busy	The selected nominal bitrate, and whether a live CAN operation is active (only one can run at a time).
CAN output	Output health during transmit/benchmark, e.g. `98% 98/100 Hz avg`. *Missed* = skipped stale periods + adapter drops. Below 98% the text turns red.

Action buttons

The main button is a single toggle: `Start` / `Stop` in scenario mode, or `Start Record` / `Stop Record` in Record Mode. `Check CAN` runs a hardware sanity check (recommended before any live operation), and `Driver Setup` opens the Windows adapter helper.

09 Manage CAN Buses

Open from **CAN › Manage CAN Buses** or the `Buses...` button. Each bus carries its own interface, channel, bitrate, CAN FD state, data bitrate, and optional DBC.

Button	Action
Use Selected	Switches the CAN Hardware panel to the highlighted bus.
Add Bus	Creates the next *Bus N*, copying settings from the active bus. If GS-USB hardware is detected, it picks the next unused gs_usb channel.
Rename	Rename a bus (e.g. *ADAS CAN*, *Chassis CAN*, *Body CAN*). Signals assigned to the old name follow the new name.
Remove	Removes a bus. *Bus 1 cannot be removed.* If the bus has signals, simCAN offers to move them to Bus 1.
Close	Closes the window.

KEY IDEA — ONE CHANNEL PER PHYSICAL BUS

Isolation is determined by CAN-H/CAN-L wiring and adapter hardware, not by simCAN. Each isolated bus needs its own CAN channel — separate adapters or a multi-channel adapter. See §24 for the full multi-bus workflow.

10 Scenario Signals & Selected Signal

Scenario Signals is the master list of loaded signals; the Selected Signal panel acts on the current signal or the visible list.

Scenario Signals buttons

`Load DBC` loads all DBC messages/signals onto the active bus (duplicates already on that bus are skipped). `Add Signal` opens the Add Signal dialog. `Import` / `Export Excel Scenario` move scenarios in and out as workbooks. The summary line beneath the buttons reports enabled count, total signals, bus count, frame-ID count, duration, output Hz (or “mixed”), visible count, and the current filter scope.

SHOW ALL SIGNALS

Off (default) shows only signals on the selected bus — best for bus-focused editing. On shows signals from every configured bus. Crucially, `Enable All` and `Remove All` always follow the *currently visible* list, so this checkbox changes their scope.

Signal table columns

Column	Meaning
Signal	Signal name.
On	Enabled for transmit. Double-click a row to toggle.
Bus	Bus assignment.
Message	CAN message name.
ID	Arbitration ID in hex.
Hz	Output frequency.
Source	`dbc` or `manual`.
Min / Max	Current profile Y range.

Click a row to select a signal: the Signal Profile controls update, the plot redraws, and the Selected Signal panel then controls that signal.

Selected Signal panel

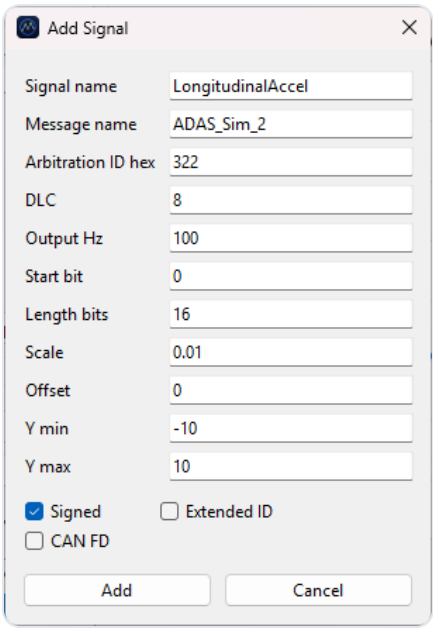
Enabled for transmit includes the selected signal in transmit (unchecked keeps it in the scenario but silent; disabled in Record Mode). Enable All enables every visible signal; Remove Selected removes one; Remove All removes the visible list — all of which respect the Show All Signals scope above.

11 Add Signal dialog

Use manual signals when you don't have a DBC, need a quick test signal, and already know the CAN ID, DLC, bit layout, scaling, and signedness. Open from **Scenario › Add Manual Signal** or the Add Signal button.

Field	Notes
Signal name	e.g. `LongitudinalAccel`.
Message name	The frame group, e.g. `ADAS_Sim_2`.
Arbitration ID hex	Hex digits such as `322` (parsed as hex).
DLC	Data length code; classic CAN uses 8. Above 8 requires CAN FD.
Output Hz	Update rate; default 100.
Start bit	Bit position where the signal begins.
Length bits	1–64 bits.
Scale	Physical scaling; cannot be zero.
Offset	Physical offset.

Field	Notes
Y min / Y max	Initial plot/edit range.
Signed / Extended ID / CAN FD	Flags for signed raw value, extended arbitration ID, and CAN FD.



Add Signal

`Add` validates the fields, adds the signal to the active bus, and selects it. `Cancel` closes without adding.

> **MANUAL PACKING CAUTION**
>
> Manual signals use little-endian packing in this version. You must know the packing, avoid overlapping bit ranges in the same frame, and choose correct scale/offset. For realistic vehicle/ADAS work, prefer DBC-backed signals whenever possible.

12 Signal Profile & the plot

The Signal Profile panel edits the selected signal's time history. The plot is where you shape it by hand. simCAN interpolates linearly between points and repeats the profile once transmit time passes the duration.

The five control groups

Group	Controls
Timeline	`Duration s` and `Output Hz`. `Apply` applies them (and the axis) to the selected signal; `Apply All` applies duration and rate to every signal. Changing duration rescales point times. The profile repeats after its duration during transmit.
Axis	`Y min` / `Y max` (Y max must exceed Y min; points are clamped to range). `Auto Y Axis` fits the selected signal; `Auto Y All` fits every signal — handy after import, draw, or record.
Endpoint	`Start` and `End` set exact values at t=0 and end-of-profile via `Set Endpoints`. `Zero Ends` sets both to zero — useful for signals that should return to neutral.
Point	`time_s` / `value` + `Apply to Point` set the selected point exactly. `Draw` toggles freehand drawing.

Group	Controls
Utilities	`Undo` / `Redo` (`Ctrl`+`Z` / `Ctrl`+`Y`), `Clear` (flat zero), and `Export CSV` (`time_s,value`).

Editing on the plot

- **Add points (freehand):** turn `Draw` on, drag through the plot, release, then turn Draw off.
- **Move a point:** with Draw off, click near a point and drag it.
- **Edit exactly:** select a point, type `time_s` / `value`, click `Apply to Point`.
- **Delete:** with Draw off, right-click a point (endpoints may be protected or recreated by normalization).

> **READING THE PLOT**
>
> Point color reflects run state: **red** = stopped, **green** = transmitting/playback active, **blue** = recording, **yellow** = selected point. The title shows `message.signal` and CAN ID, with a *DISABLED* prefix if the selected signal is off. The cursor readout under the plot shows time/value as you hover — use it to place points by eye.

13 Console

The console is simCAN's running log: startup notes, detected devices, DBC and Excel results, preflight errors, CAN-check status, transmit/record/playback messages, output health, and benchmark results.

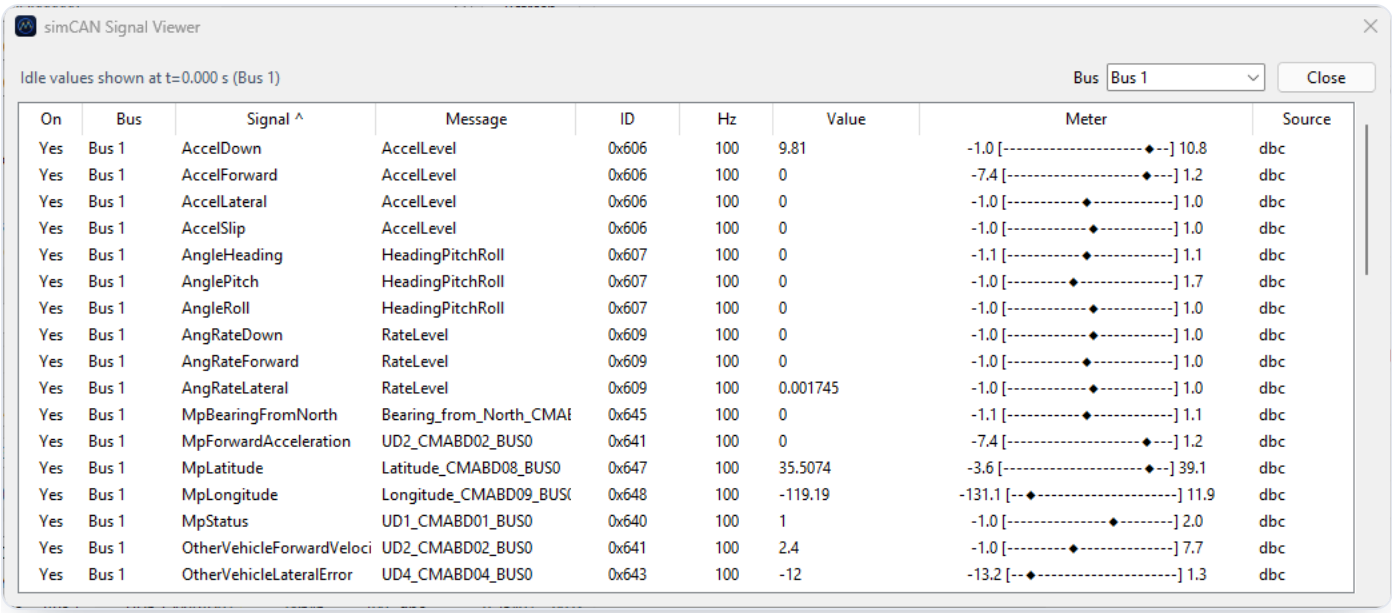
Header buttons: `Signals` opens the Signal Viewer; `Copy` copies all console text to the clipboard (ideal for support emails); `Clear` clears it; `Collapse` / `Expand` frees up plot space.

```
[21:07:40] Excel import note: Signals sheet had no enabled column, so all imported signals were
enabled.
[21:07:40] Imported workbook into selected bus Bus 1.
[21:07:40] Imported Excel scenario with 56 signal(s) into Bus 1 from
simcan_scenario_right_turn.
[21:07:46] Auto-scaled Y axis for all 56 signals.
```

> **TIP**
>
> When something looks wrong, click `Copy` and paste the console text into your notes or a support request — it usually contains the exact warning that explains the behavior.

14 Signal Viewer

A sortable table of decoded/current values across signals. Open from **View › Signal Viewer** or the console's `Signals` button.



On	Bus	Signal ^	Message	ID	Hz	Value	Meter	Source
Yes	Bus 1	AccelDown	AccelLevel	0x606	100	9.81	-1.0 [----------◆--] 10.8	dbc
Yes	Bus 1	AccelForward	AccelLevel	0x606	100	0	-7.4 [----------◆---] 1.2	dbc
Yes	Bus 1	AccelLateral	AccelLevel	0x606	100	0	-1.0 [----------◆----------] 1.0	dbc
Yes	Bus 1	AccelSlip	AccelLevel	0x606	100	0	-1.0 [----------◆----------] 1.0	dbc
Yes	Bus 1	AngleHeading	HeadingPitchRoll	0x607	100	0	-1.1 [----------◆----------] 1.1	dbc
Yes	Bus 1	AnglePitch	HeadingPitchRoll	0x607	100	0	-1.0 [---------◆----------] 1.7	dbc
Yes	Bus 1	AngleRoll	HeadingPitchRoll	0x607	100	0	-1.0 [----------◆----------] 1.0	dbc
Yes	Bus 1	AngRateDown	RateLevel	0x609	100	0	-1.0 [----------◆----------] 1.0	dbc
Yes	Bus 1	AngRateForward	RateLevel	0x609	100	0	-1.0 [----------◆----------] 1.0	dbc
Yes	Bus 1	AngRateLateral	RateLevel	0x609	100	0.001745	-1.0 [----------◆----------] 1.0	dbc
Yes	Bus 1	MpBearingFromNorth	Bearing_from_North_CMAI	0x645	100	0	-1.1 [----------◆----------] 1.1	dbc
Yes	Bus 1	MpForwardAcceleration	UD2_CMABD02_BUS0	0x641	100	0	-7.4 [----------◆---] 1.2	dbc
Yes	Bus 1	MpLatitude	Latitude_CMABD08_BUS0	0x647	100	35.5074	-3.6 [----------◆--] 39.1	dbc
Yes	Bus 1	MpLongitude	Longitude_CMABD09_BUS(	0x648	100	-119.19	-131.1 [--◆----------] 11.9	dbc
Yes	Bus 1	MpStatus	UD1_CMABD01_BUS0	0x640	100	1	-1.0 [----------◆--------] 2.0	dbc
Yes	Bus 1	OtherVehicleForwardVeloci	UD2_CMABD02_BUS0	0x641	100	2.4	-1.0 [---------◆----------] 7.7	dbc
Yes	Bus 1	OtherVehicleLateralError	UD4_CMABD04_BUS0	0x643	100	-12	-13.2 [--◆----------] 1.3	dbc

Signal Viewer. Each row shows the live value plus a text *meter* placing it between Y min and Y max. The diamond marks the current value.

Column	Meaning
On / Bus	Enabled state and bus assignment.
Signal / Message / ID	Names and arbitration ID.
Hz	Output/update frequency.
Value	The current computed, idle, transmit, or decoded value.
Meter	A text bar showing the value's approximate position in range; a diamond marks the current value.
Source	`dbc` or `manual`.

The `Bus` selector chooses one bus or *All* — independent of Show All Signals in Scenario Signals. Click a header to sort; click again to reverse. When idle, values are shown at t=0; while transmitting they update live; while recording with a DBC loaded, decoded incoming values update as matching frames arrive.

PART THREE

Building & running scenarios

From a DBC to live frames — authoring profiles, the Excel and LLM workflows, validation, transmitting, and proving your hardware can keep up.

15 Workflow with a DBC

The recommended path. A DBC gives simCAN correct IDs, scaling, packing, and limits, so the scenario you build is the scenario that goes on the wire.

1. Extract and activate simCAN. Connect the USB-to-CAN adapter and wire it to a properly terminated bench bus.
2. Open simCAN and **keep Dry run checked** while building.
3. In CAN Hardware, select the target `Bus`, `Interface`, `Channel`, and `Bitrate`; leave CAN FD off unless intended. Click `Refresh`.
4. Click `Load DBC`, choose the file, and wait for signals to populate.
5. Select a signal and edit its profile in Signal Profile. Repeat for each signal that matters; disable signals that should not transmit.
6. Use `Auto Y Axis` or `Auto Y All` after editing.
7. Optionally `Export Excel Scenario` to edit in Excel or with an LLM, then import it back.
8. Run **CAN › Validate Scenario** and fix errors; review warnings.
9. When ready for hardware, uncheck `Dry run` and click `Check CAN`; confirm Adapter, Driver, and Bus look ready.
10. Click `Start`. Watch the plot, console, Signal Viewer, and the CAN output metric. If output falls below 98%, reduce load or run the benchmark. Click `Stop` when done.

> **TIP**
> For a new adapter or bus, start with a small scenario or a lower rate, confirm output health, then scale up.

16 Workflow without a DBC

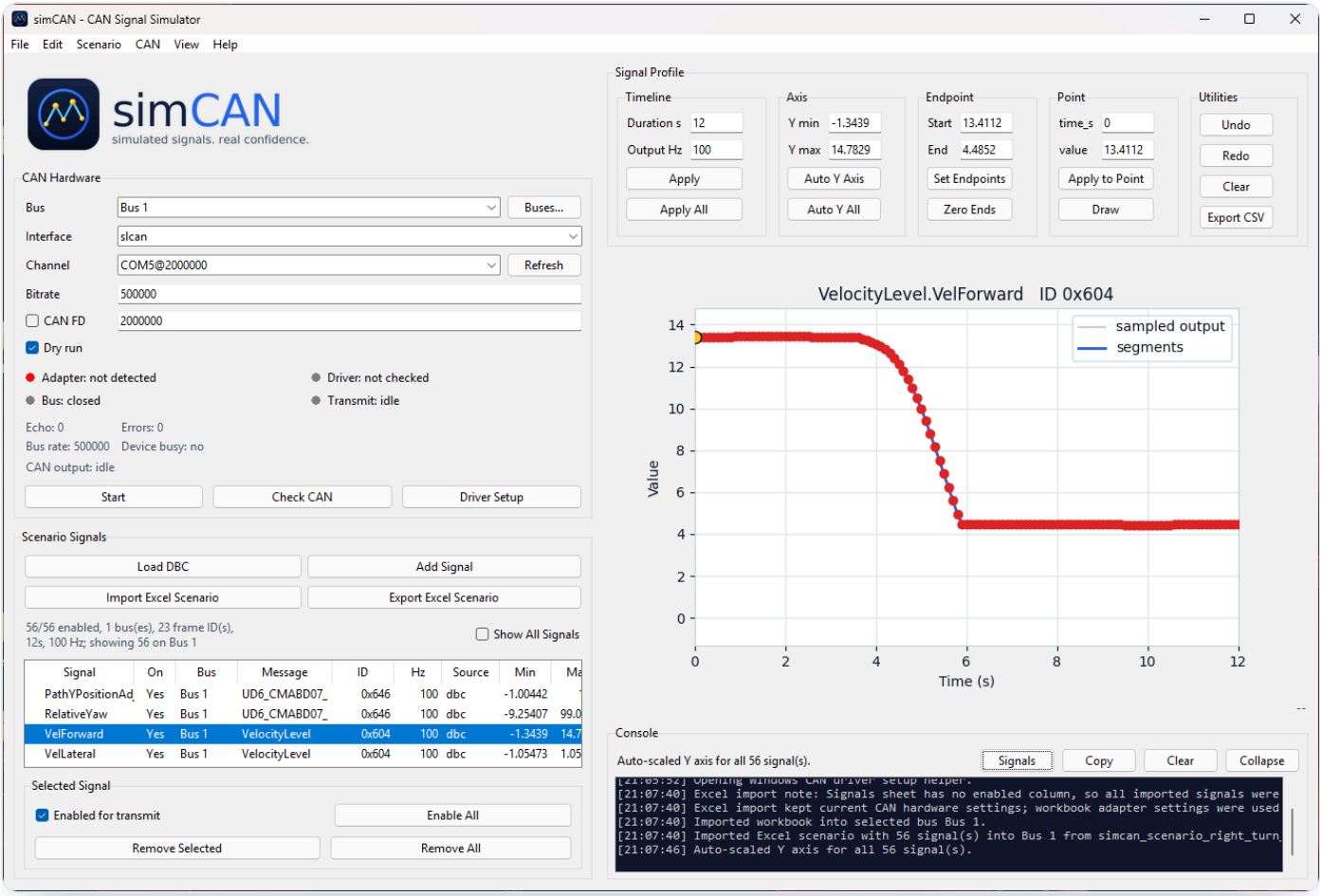
When no DBC is available, define signals by hand. This is quick for a few test channels, but you own the packing.

1. Open simCAN, select the target **Bus**, and configure Interface, Channel, Bitrate, and CAN FD. Keep Dry run checked.
2. Click **Add Signal** and fill in name, message, arbitration ID (hex), DLC, output Hz, start bit, length bits, scale, offset, and Y min/max. Set the Signed, Extended ID, and CAN FD flags as needed; click **Add**.
3. Select the new signal and shape its profile with points, endpoints, or Draw. Add more manual signals as needed.
4. Run **Validate Scenario** and fix any bit-range overlap errors.
5. Use **Check CAN** before going live, and uncheck Dry run only when ready to send real frames.

> **MANUAL-SIGNAL CAUTIONS**
>
> You must know the packing, avoid overlapping bit ranges in the same frame, and choose correct scaling and offset. For production-like traffic, DBC-backed signals are safer.

17 Excel scenario workflow

Export a scenario to an `.xlsx` workbook to edit many signals quickly, keep a repeatable artifact, or hand it to an LLM — then import it back. Export keeps the DBC/reference data so you only change time histories.



A scenario imported from Excel. 56 signals across 23 frame IDs on Bus 1; the console reports the import and the auto-scaled axes.

Workbook sheets

Sheet	Contents
Signals	One row per signal: definitions, bus assignment, enabled state, source, DBC path, packing fields, output Hz, and axis settings.
Points	One row per time/value breakpoint. `Points.signal_key` must match `Signals.signal_key`; multiple rows define the piecewise-linear profile.
AppSettings	Schema, active bus, reference interface/channel/bitrate, CAN FD, data bitrate, and dry-run setting.
Buses	One row per isolated bus (name, interface, channel, bitrate, CAN FD, data bitrate, DBC path). Local hardware settings are preserved on import.
Notes	Human-readable workbook notes.

Columns that matter

Column	Notes
signal_key	Unique key used by simCAN. **Do not change it** — Points rows depend on this exact value.
bus	Bus assignment. Older workbooks without this column import as Bus 1.
enabled	`TRUE` transmits after import; `FALSE` keeps the signal but silent. If an older workbook lacks it, add an `enabled` header at the *end* rather than moving columns.
source / dbc_path	`dbc` or `manual`, and the DBC path for DBC-backed rows (if unavailable on import, simCAN may warn and load as manual).
message_name / signal_name / arbitration_id / dlc / output_hz	Identity and rate. Arbitration ID may be decimal or hex (e.g. `0x321`).
start_bit / length_bits / signed / scale / offset / byte_order / extended_id / can_fd / y_min / y_max	Packing, scaling, flags, and edit range.

Points rows require `signal_key`, `time_s`, and `value`. Give each signal at least two points, include t=0 and the final scenario time, and add intermediate rows for ramps, holds, steps, or events. simCAN sorts points by time on import.

Import behavior

Select the destination bus in CAN Hardware first, then **File › Import Excel Scenario**. simCAN maps the workbook's active/source bus into the selected bus and **replaces only that destination bus**, preserving other buses and their local adapter settings. Run `Validate Scenario` before any live transmit.

INDEPENDENT BUSES

Because import replaces only the destination bus, you can load a Bus 1 scenario and a Bus 2 scenario independently without wiping the other.

18 LLM Excel authoring workflow

When a DBC has many signals, let an LLM build a realistic scenario from the exported workbook. simCAN ships a prompt that tells the model to ask engineering questions first and to preserve the workbook's structure.

1. Open simCAN, select the target bus, and **Load DBC** so the export carries correct IDs, scaling, and packing.
2. **Export Excel Scenario**, then choose **File › Copy LLM Excel Prompt.**
3. Paste the prompt into your LLM, attach the exported workbook, and ask it to follow the prompt and ask questions first.
4. Choose a scenario type when asked — e.g. AEB lead-vehicle braking, AEB crossing pedestrian, LKAS lane keeping/departure, ACC stop-and-go, cut-in/cut-out, FCW-only, blind-spot/lane-change warning, sensor dropout/fault injection, or custom bench replay.
5. Answer the concise engineering questions (ego/target speed, initial distance, time-to-collision, trigger time, accel/decel, ramp rates, lane offset, curvature, lateral velocity, heading, pedestrian speed, crossing direction, hold/recovery, thresholds, which signals to enable, and so on).
6. Tell the LLM to preserve all sheet names and headers, `signal_key` values, DBC/reference columns, the `bus` column, and the Buses and AppSettings sheets — and to build profiles in the Points sheet.
7. Download the edited workbook, select the destination bus, and **Import Excel Scenario**.
8. Review key signals' plots, use Auto Y as needed, run **Validate Scenario**, and go live only after plots, enabled signals, bus, bitrate, and hardware are correct.

MAKE ENABLED NUMERIC CHANNELS LOOK LIVE

Steady enabled numeric signals should carry tiny realistic noise, drift, or variation within tolerance — but discrete/enum/status/counter/validity signals should change only when a logical event occurs. Keep values physically reasonable and inside DBC limits unless you are intentionally doing fault or boundary testing.

19 Validation

Run **CAN › Validate Scenario** any time; Start Transmit also validates automatically. Errors must be fixed before transmit; warnings should be reviewed.

What validation checks

Category	Checks
Scenario presence	Whether any signals are loaded and any are enabled.

Category	Checks
Bus & rate	Valid bitrate; current vs. imported-scenario bitrate; CAN FD state; high requested frame rates; estimated classic-CAN bus load.
Profiles	Output frequency > 0; duration > 0; Y min < Y max; at least two points per signal.
Packing	DLC > 8 without CAN FD; values exceeding DBC ranges; manual scale of zero; invalid manual bit start/length; bit range exceeding DLC; overlapping manual bit ranges in the same bus/frame.
Metadata	Missing or unavailable DBC metadata/path.

How to respond

- A **DBC range warning** may mean you are intentionally testing a boundary — or that LLM/Excel values are unrealistic.
- A **high bus-load warning** means timing may not be reliable.
- A **bitrate-mismatch warning** means your live hardware setting differs from the reference bitrate in an imported workbook.

20 Transmitting a scenario

Always rehearse in dry run, then go live deliberately. simCAN schedules frames against wall-clock time so scenario values stay aligned with real bench time.

Dry-run transmit

1. Keep Dry run checked.
2. Load or import signals and enable the ones you want.
3. Click `Start` — frames are calculated without opening hardware.
4. Watch the plot, Signal Viewer, and console; click `Stop`.

Live transmit

1. Confirm wiring, termination, bus, interface, channel, bitrate, and CAN FD.
2. Enable only the signals you intend to send; run `Validate Scenario`.
3. Click `Check CAN`; confirm adapter/driver/bus are ready.
4. Uncheck `Dry run`, click `Start`, and watch CAN output and warnings. `Stop` when done.

How timing works

Frame IDs at the same requested rate are phased across the period rather than sent as one burst, and multiple active buses share a common software start clock. If the PC, adapter, or bus cannot keep up, simCAN **skips stale frame periods** instead of stretching the scenario into slow motion — this keeps a 100 Hz scenario aligned with other sensors rather than silently becoming a 50 Hz one. Separate adapters can still show small USB/driver timing differences.

READING THE CAN OUTPUT METRIC

Aim for **98% or higher**; below that the text turns red. *Missed* counts skipped stale periods plus adapter send drops, and per-ID *underrate* warnings name frames that aren't keeping up. If health is poor, reduce enabled frames, lower output Hz, cut bus load, or change adapter/backend. Note that 100% output does *not* prove every receiver accepted every frame — use a separate bus analyzer for critical timing verification.

EDITING IS LOCKED WHILE RUNNING

Profile and rate editing is locked during transmit/playback; stop and restart transmit to apply a new output schedule. Workbooks imported without an `enabled` column import with all signals enabled.

21 Built-in CAN Benchmark

Before a demanding test, qualify the exact laptop / adapter / driver / USB / bus path. Open **CAN › Built-in CAN Benchmark**.

What it sends

One fixed frame, arbitration ID `0x321`, at **50, 100, 200, and 500 Hz** for about two seconds each.

What it reports

Target rate, accepted actual rate, output percentage, accepted/expected sends, skipped stale periods, dropped adapter sends, maximum lateness, and `bus.send()` average / p95 / max timing.

Run it

1. Connect the adapter to a properly terminated bus and select the target bus.
2. Set Interface, Channel, Bitrate, and CAN FD.
3. Uncheck **Dry run** and choose **Built-in CAN Benchmark**.
4. Review the results; if output is below 98%, reduce rate/frame count or change adapter/backend/USB path.

DRY RUN IS NOT ALLOWED HERE

The benchmark exists to measure the real adapter, driver, USB path, and bus — so it always runs live. Confirm critical results with an external analyzer.

PART FOUR

Capture, replay & hardware

Recording live frames into editable profiles, replaying captured logs, running several isolated buses at once, sorting out adapter drivers, and working in the dark.

22 Record Mode

Record Mode flips simCAN from a transmitter into a listener. It captures live frames from a bus and — when a matching DBC is loaded — decodes them into named signals you can keep, edit, and replay.

Capture a session

1. Choose **File › Record Mode** to switch from output to capture.
2. Load the DBC for the bus first if you want frames decoded into named signals rather than raw IDs.
3. Select the `Bus` to capture and uncheck `Dry run`.
4. Click `Start Record`. The selected signal plots live from decoded incoming frames as they arrive.
5. Click `Stop Record` when finished.

What you get back

When recording stops, the captured traces are converted into **editable profiles** and simCAN returns to normal Scenario Mode. From there you can drag points, run Auto Y, and choose `Export Excel Scenario` to save the capture for reuse.

Each session is also written to disk as a timestamped JSONL log (see below) so it can be replayed later with Playback.

The recording file

Captures are written to **logs\simcan_recording_*BUS*_*YYYYMMDD*_*HHMMSS*.jsonl**. The first line is a metadata header (bus, bitrate, start time); every line after it is a single timestamped frame. Because it is line-delimited JSON, the file streams to disk as frames arrive and is safe to inspect in any text editor.

> **CONTROLS LOCK DURING CAPTURE**
>
> Enabled-for-transmit, scenario import/export, and profile editing are locked while a recording is active, because changing them mid-capture could invalidate the result. `Auto Y Axis` and `Auto Y All` stay available so you can keep the live trace in frame.

23 Playback

Playback replays a recorded JSONL capture back onto a live CAN bus, reproducing the original frames with their original timing — useful for re-driving a downstream ECU or test rig with a known-good trace.

1. Select the live destination bus in CAN Hardware and uncheck `Dry run`.
2. Choose **CAN** › **Playback Log** and select a `.jsonl` recording.
3. simCAN replays the frames using the timing stored in the log.

BUS MAPPING IS AUTOMATIC

Logs record their source bus. If a single-bus log is loaded while a different bus is active, simCAN maps the log onto the active bus and notes it in the console — so a capture taken on one bench bus can be replayed onto another without hand-editing the file.

ONE ADAPTER JOB AT A TIME

Transmit, Record, Playback, and Check CAN are **mutually exclusive** — only one may use the adapter handle at a time. Stop the active task before starting another. Error frames found in a log are skipped on replay.

24 Multiple isolated CAN buses

Real vehicles split traffic across several physical networks — ADAS CAN, Chassis CAN, Body CAN. simCAN models each as an isolated bus with its own hardware, bitrate, and DBC, and transmits them in parallel without leaking frames between them.

Add and configure buses

Open **CAN** › **Manage CAN Buses** to add buses. Each one carries its own interface, channel, bitrate, CAN FD setting, and optional DBC. Bus 1 always exists and cannot be removed.

Assigning signals to a bus

A signal is assigned to whichever bus is active in CAN Hardware at the moment you load its DBC or add it. To place signals elsewhere, select that bus first — or set the `bus` column when importing from Excel.

How transmit isolates traffic

Transmit starts **one independent worker per bus** that has enabled signals. Frames on Bus A never appear on Bus B. Leave `Show All Signals` off for focused single-bus work; turn it on to compare across buses.

Physical wiring

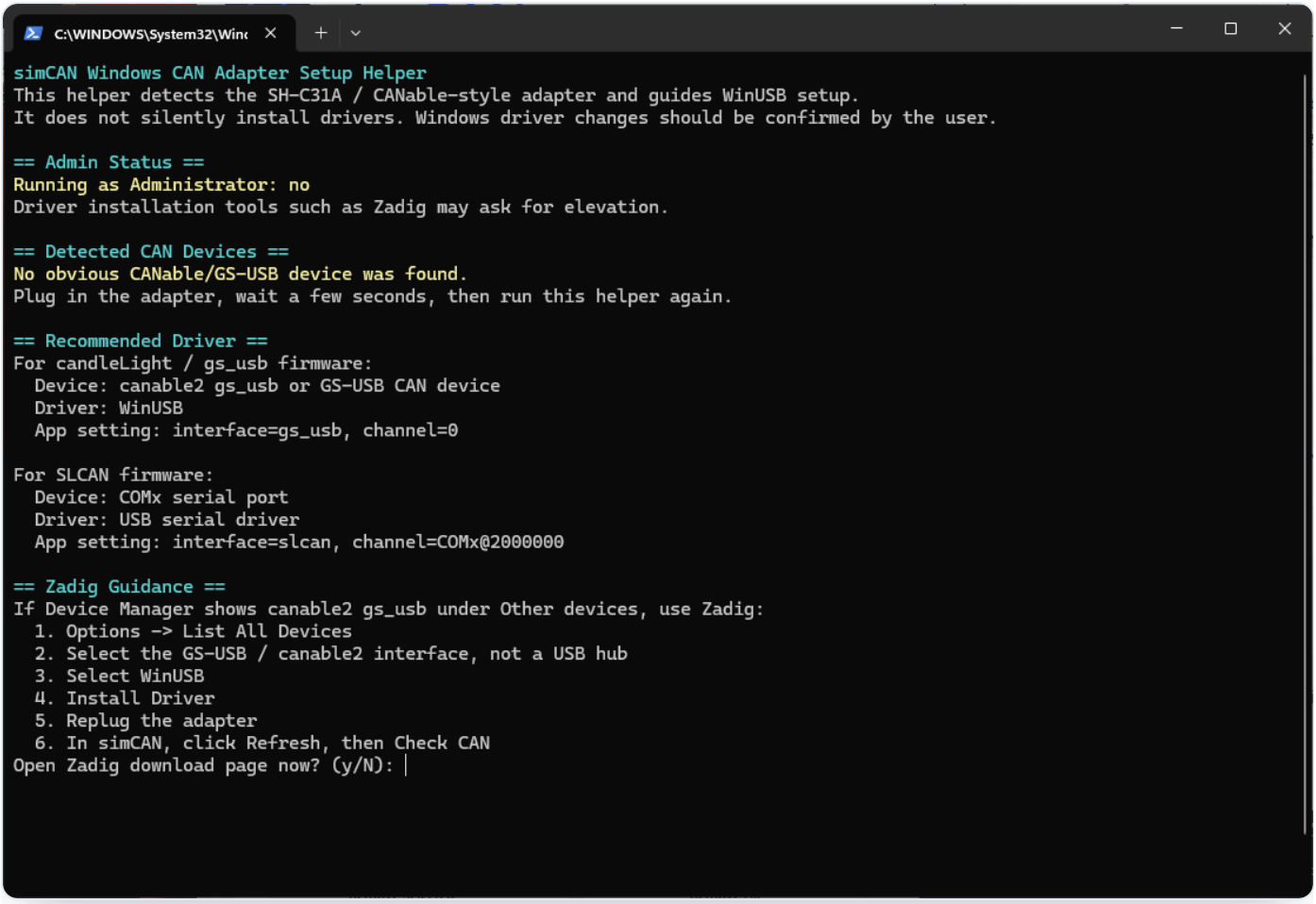
Each bus needs its own channel on the laptop — either separate USB-to-CAN adapters or separate channels on a multi-channel adapter. A powered USB hub is fine if the adapters are stable. When you add a bus, simCAN picks the next unused GS-USB channel if GS-USB hardware is present.

INDEPENDENT PER-BUS IMPORT

Excel import targets only the bus selected at import time and replaces just that bus, so Bus 1 and Bus 2 scenarios load independently without wiping each other. Import also preserves existing local hardware settings, so a stale workbook can't silently switch a detected adapter back to an old COM port. Older one-bus workbooks import as Bus 1.

25 Driver Setup helper

CANable-style and GS-USB adapters need the right Windows driver before simCAN can open them. The built-in helper detects what is plugged in and walks you through WinUSB / Zadig setup — it advises, it does not silently change drivers.



The Driver Setup helper reports admin status, detected devices, and the recommended driver and app settings for each firmware type.

Click **Driver Setup** in CAN Hardware to open it. The helper reports:

Section	What it tells you
Admin Status	Whether simCAN is running as Administrator. Driver tools such as Zadig may need elevation.
Detected CAN Devices	Whether a CANable / GS-USB device was found. If none, plug in the adapter, wait, and re-run the helper.
Recommended Driver	For candleLight / gs_usb firmware: WinUSB driver, app setting `interface=gs_usb, channel=0`. For SLCAN firmware: USB serial driver, app setting `interface=slcan, channel=COMx@2000000`.
Zadig Guidance	Step-by-step: list all devices, select the GS-USB / canable interface (not a USB hub), install WinUSB, replug, then Refresh and Check CAN in simCAN.

DRIVER CHANGES ARE YOUR CALL

The helper does **not** install drivers on its own. Windows driver changes should be confirmed by you, and Zadig may prompt for elevation. Choose the correct interface in Zadig — selecting a USB hub instead of the adapter is the most common mistake.

26 Dark theme

simCAN ships in a light theme by default. A dark theme is available from **View › Dark Theme** for long bench sessions or dim labs.

The theme affects window chrome, panels, and the plot background; signal colors and the red/green/blue point states remain consistent so you can still read transmit, playback, and record status at a glance. The choice persists across launches.

PART FIVE

Reference & support

When something is wrong, when you want a checklist before a live run, and when you need a one-line reminder of how a task is done.

27 Troubleshooting

The most common bench problems, what usually causes them, and what to try first. Work top to bottom within each row — the actions are ordered roughly from quickest to most thorough.

Symptom	Likely cause	What to try
simCAN only opens to the license splash	No license installed, trial expired, license for another machine, or invalid.	Start a trial with `Start 7-Day Trial`, or buy a key via `Buy License` then `Activate Online`; `Copy Machine ID` for support; check **%LOCALAPPDATA%\simCAN\simcan.license**; or `Load License` for a manual file.
Windows shows an "unknown publisher" warning	Launch build ships before paid public code signing is complete.	Confirm the download came from simulatedsignals.com, compare the SHA-256 hash, and continue only if hash and source are trusted.
Adapter is not detected	Unplugged, bad cable, missing driver, wrong backend, or GS-USB needs WinUSB.	Replug; try another port or a powered hub; click `Refresh`; open `Driver Setup`; check Device Manager. Use `slcan` if it appears as COMx, `gs_usb` if it appears as GS-USB/CANable/candleLight.
Second adapter doesn't show as another COM port	Adapter is in gs_usb/candleLight mode, not slcan mode.	Use interface `gs_usb` with channel 0 for the first adapter and 1 for the second. Install WinUSB for each. Don't expect extra COM ports unless flashed to slcan.
Check CAN fails for gs_usb	WinUSB/libusb backend missing, wrong channel, device not detected, or driver on the wrong interface.	Open `Driver Setup`; install/select WinUSB via Zadig for the CAN interface; replug; `Refresh`; use gs_usb channel 0/1; run `Check CAN` again.
Check CAN sends a frame but no echo	Adapter doesn't echo sent frames, or wiring / termination / bitrate issue.	Verify wiring, termination, and bitrate; confirm with an external analyzer; try live transmit to a known receiver only if safe.

Symptom	Likely cause	What to try
CAN output is below 98%	Output Hz too high, too many frames, bus too loaded, adapter/PC/USB latency.	Run the **Built-in CAN Benchmark**; lower output Hz; disable unneeded signals; combine frames where DBC-correct; try gs_usb instead of slcan; use a direct port or powered hub; close heavy apps; confirm with an analyzer.
Scenario imports into the wrong bus	Wrong active bus at import, or workbook bus differs from intended destination.	Select the destination bus in CAN Hardware first, then import again; read the console import message; use **Show All Signals** to confirm assignments.
Importing a Bus 2 scenario wipes Bus 1	Import should replace only the destination bus; other buses are preserved.	Confirm the active bus and the workbook's bus columns, confirm you're on the latest build, and if it persists copy the console output and contact support.
DBC-backed rows import as manual	DBC path in the workbook is invalid here, file moved, or message name not found.	Place the DBC at the expected path or load it first; re-export the workbook on this machine; ask the LLM not to modify DBC path / message / source columns.
Workbook has no enabled column	simCAN enables all imported signals by design.	Export a fresh workbook, or add an `enabled` header at the end of Signals with TRUE/FALSE values — without moving existing columns.
Imported plot range looks wrong	y_min/y_max too narrow, or values outside expected limits.	Use **Auto Y Axis** / **Auto Y All**, inspect Points values, and run **Validate Scenario**.
Record Mode shows no decoded values	No DBC, DBC doesn't match incoming IDs, wrong bus, or wrong bitrate/interface.	Load the correct DBC before recording; confirm bus and bitrate; use Signal Viewer; confirm frames exist with an external analyzer.
Playback won't start	Dry run checked, another CAN operation active, bus mismatch, or adapter can't open.	Uncheck **Dry run**; stop transmit/record/benchmark/check; select the correct bus; run **Check CAN**.
Driver Setup won't run	Helper missing from package, PowerShell policy prompt, or non-Windows system.	Re-extract the full ZIP, run from a normal extracted folder, and on Windows try **Help › Driver Setup** again.

28 Safety & pre-live checklist

simCAN generates real CAN traffic. Treat it like any other bench signal source: contained, validated, and confirmed before it touches anything that matters.

GOLDEN RULE

Use simCAN on a bench or isolated test bus. Do **not** connect generated traffic to a vehicle network unless the IDs, scaling, rates, and bus topology are controlled and approved. A clean dry run does not prove the hardware can keep up, and 100% CAN output does not prove every receiver accepted every frame — confirm with an analyzer or the receiver side when it matters.

Good practice

Start in Dry run. Validate, then Check CAN, before any live transmit. Use the benchmark for high-rate tests and an external analyzer for critical timing. Keep DBC files and scenario workbooks controlled and versioned. Avoid enabling irrelevant signals or feeding unrealistic flat values to live-looking channels, and keep status / enum / counter / validity signals logically valid.

Before every live bench run

1. Confirm adapter wiring and bus termination.
2. Confirm bitrate and CAN FD state.
3. Confirm the selected bus and enabled signals.
4. Confirm scenario duration and output Hz.
5. Confirm DBC path, plot values, and ranges.
6. Confirm there are no validation errors.
7. Confirm output capability via benchmark or prior measurement.
8. Confirm the receiver / logger / analyzer is ready.
9. Start transmit and watch the CAN output metric.
10. Stop if output health, wiring, or receiver behavior is off.

29 Quick reference: common tasks

The short version of every routine workflow — for when you know what to do and just need the order of operations.

Install & activate

Download the ZIP → Extract All → run **simCAN.exe** → Start 7-Day Trial (or Buy License → Activate Online → paste key) → workbench opens.

Load a DBC

Select bus → Load DBC → choose `.dbc` → select a signal.

Create a manual signal

Select bus → Add Signal → fill fields → Add.

Edit a profile

Select signal → set duration / output Hz → set Y min/max → drag points or use Draw → Apply / Apply All → Undo/Redo as needed.

Validate

CAN › Validate Scenario → fix errors.

Check CAN

Configure interface / channel / bitrate → uncheck Dry run → Check CAN.

Transmit

Validate → Check CAN → uncheck Dry run → Start → watch CAN output → Stop.

Record

Select bus → load DBC if decoding → uncheck Dry run → **File › Record Mode** → Start Record → Stop Record.

Export Excel

File › **Export Excel Scenario** → save `.xlsx`.

Use an LLM

Export Excel → **File** › **Copy LLM Excel Prompt** → paste prompt into the LLM → attach workbook → import the returned workbook.

Playback

Select bus → uncheck Dry run → **CAN** › **Playback Log** → select `.jsonl`.

Benchmark

Select bus → uncheck Dry run → **CAN** › **Built-in CAN Benchmark** → review output % and timing.

Open Signal Viewer

Click `Signals` in the console header.

30 Getting support

The more of the picture you send up front, the faster a problem gets solved. Gather the items below before writing in.

What to include

Your purchase/order email, Windows version, and simCAN version/build date. Whether you're using the ZIP/exe from simulatedsignals.com, and your Machine ID if activation is involved.

For hardware issues: adapter model, firmware/backend (slcan or gs_usb), and your Interface, Channel, Bitrate, and CAN FD settings. Note whether Dry run was checked, how many buses are configured, how many signals are enabled, and the output Hz values.

Then the evidence: the CAN output metric shown during transmit, the console text (use the `Copy` button), and any relevant DBC file, Excel workbook, or JSONL log.

CONTACT

Support email

`support@simulatedsignals.com`

Downloads & activation

`https://simulatedsignals.com`

FASTEST PATH

Reproduce the issue with Dry run on, click `Copy` in the console, and paste that text into your email. The console's timestamped lines usually point straight at the cause.