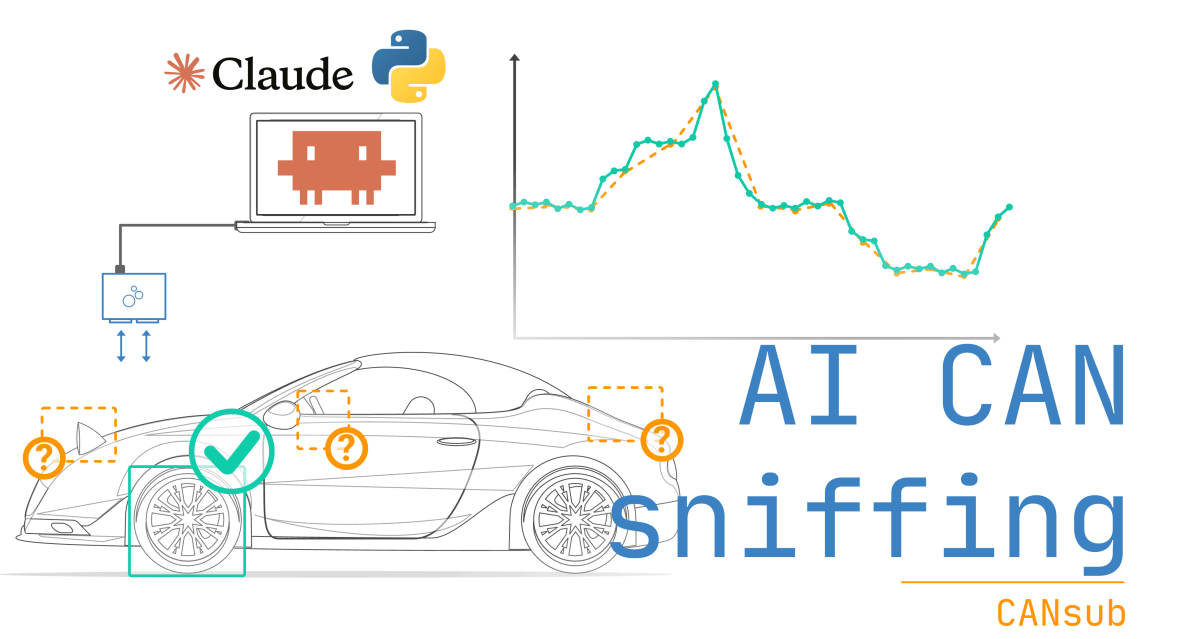




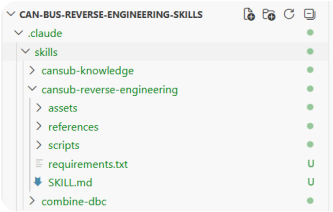
AI CAN bus reverse engineering [Claude Code]

Can an AI agent reverse engineer your proprietary CAN bus signals?

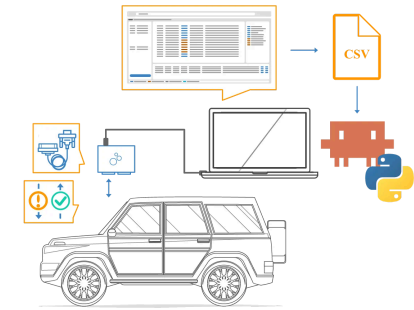
Martin Falch tests this using our CANsub interface, python-can and a custom Claude Code 'skill' to decode speed, RPM and more.

Quick context: Reverse engineering proprietary CAN signals is an extremely common use case (e.g. for mixed fleet telematics - or just for fun). However, doing it by hand is slow and demands deep expertise. This workflow lets you do it with minimal expertise - and in 95%+ less time vs. the 'classic' approach.

The intro covers the below topics:



The skill
We explain how the Claude Code skill works (instructions + context material + Python scripts) and how Claude uses it to dynamically reverse engineer proprietary CAN data.



Reference signals
Learn why 'reference signals' are vital. This can be e.g. a CAN-based signal (like OBD2, GPS-to-CAN, ...), a filmed dashboard value (leveraging vision OCR) - or noisy human input via a built-in browser app.



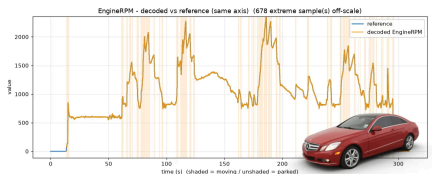
Step-by-step analysis
Get the technical breakdown of what happens under-the-hood incl. how Claude performs bit-activity surveys, correlation ranking, exhaustive bit searching for start bit/length/byte order/sign and finally scale & offset fitting with physical sanity checks - and how the result is formalized in ready-to-use DBC files - all using python-can.



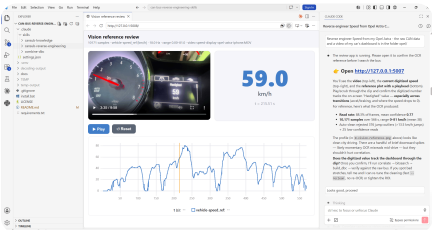
Why use the CANsub?
4 key reasons why we recommend using the CANsub.2 as your CAN sniffer interface.



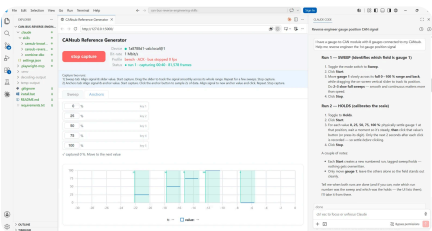
How to get started
We provide step-by-step guidance to enable you to try this yourself within 10 minutes - incl. free sample data and the 100% open source skill.



Ex 1: Mercedes speed & RPM (OBD2)
Using OBD2 PIDs as reference, Claude decodes proprietary speed + RPM signals in 5 min - producing a ready-to-use Mercedes DBC file.



Ex 2: Opel Astra speed (vision OCR)
If no CAN-based reference is available, we can sometimes fall back to vision-based OCR (Optical Character Recognition). Here, the car's dashboard speed is filmed and digitized - letting Claude decode the proprietary speed signal.



Ex 3: Gauge position (human input)
Claude single-shot decodes a gauge position signal from a noisy 'human input' reference signal (via a built-in browser app) - and extrapolates it to 7 other gauges. We then verify the validity live via webCAN.

As the intro shows, **this concept works extremely well**. We believe this will enable a real democratization of CAN sniffing - and a potential 95%+ time reduction per signal.

Tip: The intro is also available as a 30 min YouTube video at the top of the article.

Learn more below - and try it yourself with our free sample data pack!

Read article