Open-source browser-based CANopen analyzer

CANviz is an open-source browser-based tool for monitoring and interacting with CANopen and J1939 networks

(Source: Adobe Stock)

The tool installs as a Python package via the pip package manager, starts a local web server, and opens in a browser at localhost:8080. No native application is required. The backend uses the FastAPI framework with python-can for hardware abstraction. The tool runs on any CAN CC (classic CAN) USB adapter.

Zero-configuration CANopen decode

Communication object identifiers (COB-IDs) in CANopen follow fixed object number ranges as defined in the CiA 301 specification. This allows every frame type to be classified from the 11-bit CAN-Identifier (CAN-ID) alone, without an electronic data sheet (EDS) file. After 15 qualifying frames, the decoder activates automatically and labels each frame in the message table by type and node-identifier (node-ID): network management (NMT) control, synchronization (SYNC), emergency (EMCY) objects, transmit and receive process data objects (TPDO/RPDO), service data object (SDO) channels, and heartbeat frames (see Figure 1).

ID	DLC	DATA	COUNT	RATE	LAST SEEN	TYPE	NODE	PROTOCOL INFO	
0x000	2	80 03	1	1	15:41:27.734	NMT	—	Enter Pre-Operational	
0x080	0		12,434	92	15:43:27.295	SYNC	—	SYNC	
0x081	8	00 00 00 00 00 00 00 00	2	1	15:41:26.738	EMCY	0x01	Error Reset / No Error	EMCY
0x181	8	37 00 01 CC 69 01 00 00	12,439	92	15:43:27.289	TPD01	0x01		
0x182	8	0F 00 00 00 00 00 00 00	12,433	92	15:43:27.292	TPD01	0x02		
0x183	8	EB 0A 87 5F 00 00 00 00	1,034	90	15:41:27.750	TPD01	0x03		
0x281	8	AB F8 FF FF E0 02 00 00	6,315	46	15:43:27.283	TPD02	0x01		
0x581	8	4B 17 10 00 FA 00 00 00	21	15	15:43:23.489	SDO-resp	0x01	SDO resp: Upload Response (expedit...	
0x582	8	4B 17 10 00 FA 00 00 00	15	15	15:43:23.551	SDO-resp	0x02	SDO resp: Upload Response (expedit...	
0x583	8	4B 17 10 00 FA 00 00 00	15	15	15:43:23.603	SDO-resp	0x03	SDO resp: Upload Response (expedit...	
0x601	8	40 17 10 00 00 00 00 00	21	15	15:43:23.315	SDO-req	0x01	SDO req: Initiate Upload 0x1017:0	
0x602	8	40 17 10 00 00 00 00 00	15	15	15:43:23.399	SDO-req	0x02	SDO req: Initiate Upload 0x1017:0	
0x603	8	40 17 10 00 00 00 00 00	15	15	15:43:23.458	SDO-req	0x03	SDO req: Initiate Upload 0x1017:0	
0x701	1	05	523	4	15:43:27.025	Heartbeat	0x01	Operational	
0x702	1	05	523	4	15:43:27.026	Heartbeat	0x02	Operational	
0x703	1	7F	523	4	15:43:27.028	Heartbeat	0x03	Pre-Operational	

Figure 1: Message table with passive frame classification (Source: Chanchal Dhiman)

A nodes table tracks every node discovered from heartbeat frames. The NMT state per node (operational, pre-operational, or stopped) updates in real time. Estimated heartbeat interval and total frame count are shown per node without any polling or SDO reads.

SDO transactions arrive as two separate CAN messages: a request from the SDO client in the CAN-ID range from 601_h to $67F_h$ and a response from the node in the CAN-ID range from 581_h to $5FF_h$. The decoder pairs these within a 500-ms window by node-ID, object index, and sub-index, then resolves the object name from a built-in object dictionary of 180 specified objects. This dictionary covers the CiA 301 communication profile objects and the CiA 402 drive profile objects. Object names, hexadecimal values, and decimal interpretations appear in the SDO log automatically from the observed network traffic, without any manual configuration (see Figure 2).

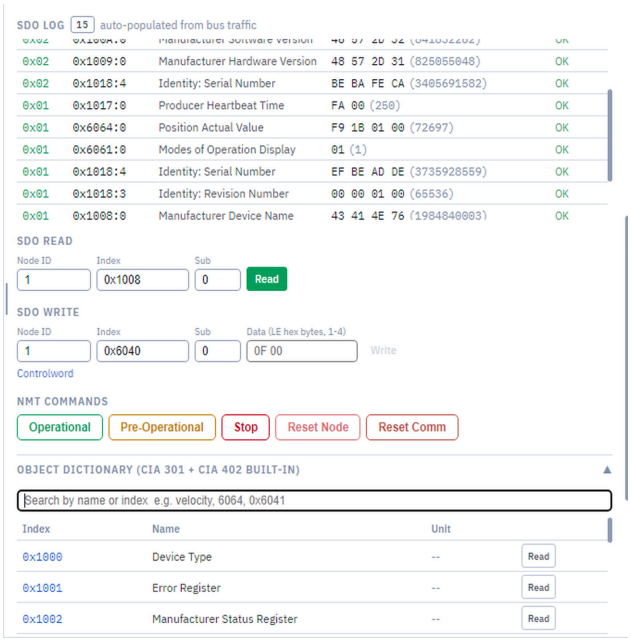

Figure 2: SDO log with object dictionary lookup (Source: Chanchal Dhiman)

EMCY frames are decoded to the 16-bit error code, the CiA 301 error class name, error register flag list, and manufacturer-specific data in hexadecimal values. The EMCY section appears only when fault frames are present and clears when the network is clean.

CiA 402 drive profile

The CiA 402 drive profile is used by servo drives and motor controllers in robotics, industrial automation, and motion control applications. Its default first TPDO mapping provides the status of the drive in the ‘statusword’ placed in byte 0 and byte 1 of the TPDO. The CANviz decoder uses this default mapping to extract the named drive state machine state from live PDO frames without an EDS file. The according state ‘Switch on disabled’, ‘Ready to switch on’, ‘Switched on’, ‘Operation enabled’, ‘Fault’, and ‘Quick stop active’ appears inline under each node in the nodes table (see Figure 3). The annotation ‘(default)’ indicates that the default TPDO mapping was assumed. Loading an EDS or DCF (device configuration file) file removes this assumption, enables full PDO payload decode for any non-default mapping, and adds resulting signals to the signal plot view.

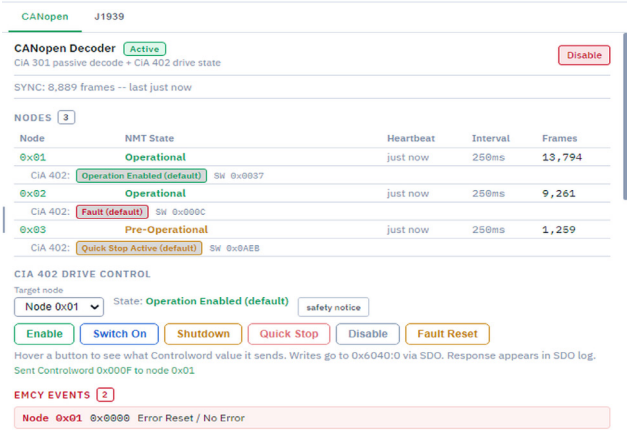

Figure 3: CANopen nodes table and CiA 402 drive control panel (Source: Chanchal Dhiman)

Active network control

NMT commands ('Operational', 'Pre-operational, 'Stop', 'Reset application, and 'Reset communication') can be sent to a specific node or broadcast to all nodes from a panel in the interface. A confirmation step is required before any command is transmitted.

CiA 402 'controlword' commands are available as single-click actions: 'Enable operation', 'Switch on', 'Shutdown', 'Quick stop', 'Disable', and 'Fault reset'. Each button displays the exact 'controlword' value it will send before the user confirms (see Figure 4). This allows drive state transitions to be executed directly from the browser without a separate commissioning tool or terminal.

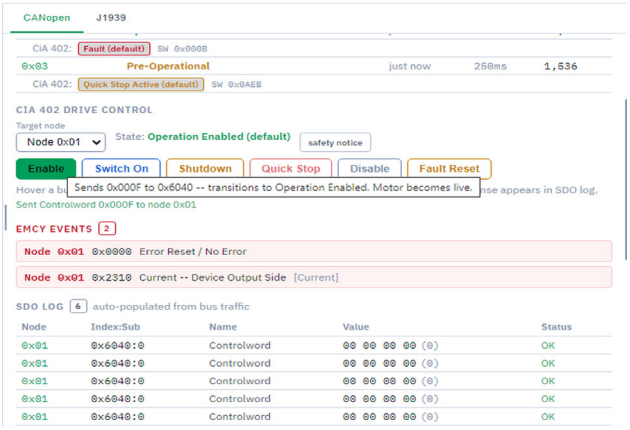

Figure 4: CiA 402 'controlword' button tooltip (Source: Chanchal Dhiman)

SDO reads and writes to any live node are supported from the same panel. The object name resolves automatically as the object index is typed, using the built-in object dictionary. For objects not in the built-in object dictionary, object index and sub-index can be entered manually with a data type selection.

J1939 support

Many embedded networks combine CANopen actuators with a J1939-based vehicle backbone traffic. The tool handles both simultaneously on a single network connection. J1939 uses 29-bit extended CAN-IDs and is auto-detected at 250 kbit/s. Each CAN-ID is split into priority, parameter group number (PGN), source address, and destination address. A database of 54 PGN definitions and 99 source address names built-in in CANviz provides named decode without a DBC (database CAN) file. Diagnostic message 1 (DM1) active fault frames are decoded to suspect parameter number (SPN), failure mode indicator (FMI), occurrence count, and lamp states. Broadcast announce message (BAM) transport protocol reassembly handles multi-packet messages in up to eight concurrent sessions. For physical signal values, a J1939 DBC file can be loaded via the DBC panel.

Signal plotting and logging

Decoded signals from CANopen (via EDS) and J1939 (via DBC) share a common signal plot view with up to eight signals on a shared time axis. The time window is selectable from 10 seconds to 30 minutes. Threshold alerts flag signals that exceed a defined limit (see Figure 5). Session recording to ASC (Vector CANalyzer log format) or CSV (comma-separated values) format runs during the live capture. Replay of saved sessions is supported at any bit rate.

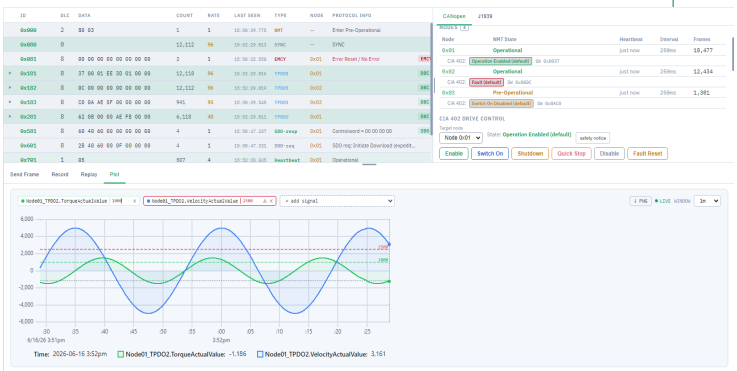

Figure 5: Signal plot threshold breach indicator (Source: Chanchal Dhiman)

Hardware

The tool supports gs_usb devices with Candlelight firmware, Peak PCAN-USB, Kvaser, slcan-compatible adapters over COM port, and SocketCAN on Linux, through the python-can abstraction layer. Low-cost adapters for 10 to 15 US-dollars are sufficient for most use cases. Actual throughput depends on the adapter and driver stack; the tool has been validated at 3500 frames per second with zero frame loss on a reference hardware. Interface setup per adapter type is documented at https://github.com/Chanchaldhiman/CANviz/blob/main/docs/interfaceguide.md.

CANviz is MIT-licensed and is available at https://github.com/Chanchaldhiman/CANviz. It installs as the canviz package via pip.

Author

Chanchal Dhiman
Formerly Michigan Technological University
cdhiman@mtu.edu
www.mtu.edu

Engineering