PhysiCAN app

Sital Technology

v 1.0

User Guide

INTRODUCTION

Sital's PhysiCAN app is a graphical user interface (GUI) tool for real-time CAN bus physical layer fault monitoring and cybersecurity. The app uses Sital's PhysiCAN (Sital's SnS device) to detect intermittent or continuous open and short circuit faults and indicate its findings as notifications on the monitored bus's topology diagram. Cybersecurity capabilities will be added in future releases, and will enable the detection of "spoofing" (Impersonating) messages by analyzing their fingerprints (or signatures) at the physical layer. The open and short circuit detection features can reduce CAN bus circuit repair time by hours by eliminating the need for time-consuming manual troubleshooting. This is a preliminary user manual for the early-stage PhysiCAN app, describes its features for detecting and locating open and short circuit faults.

MINIMUM SYSTEM REQUIREMENTS

In order to work with the PhysiCAN app, 2 assets are required:

1. Sital's SnS (Safe and Secure) device "PhysiCAN" - a physical CAN bus node that connects to the bus and monitors activity.
2. A Windows PC computer with a USB 2.0 port.
3. Topology file - A dedicated file format that describes key CAN-Bus attributes for a specific bus. Note that topology files are interpreted by the PhysiCAN app, and therefore should be carefully checked for syntax and logic as if it was a piece of code.

*Future releases of the PhysiCAN app will include a dedicated wizard to create topo files.

TOPOLOGY FILES

Topology files are Sital's own "topo" format (.topo extension) that describe CAN-Bus topologies by enumerating some of its high level attributes. These files are interpreted by the app as providing the topology for the CAN-Bus to be monitored. At this early stage, the interpreter includes no error detection mechanism. This will be added in future releases. Therefore, writing topo files should be done carefully and double-checked for syntax and logic errors, however, they are very easily read and written.

Structure:

- Each line describes an attribute.
- No comments or any content other than the required attribute lines are allowed.
- The following block is a topology file template that includes an explanation for each line. The explanations are indented and start with a '$' symbol. Strings inside <> brackets are values

inserted by the author of the file. Any other string outside <> brackets are constant strings that are included in the specific line for every topo file, and should not be altered by the author.

$Constant title for every topology file

Topology File Format Version 1.0, Sital Technology format,

$Name of the bus

Name of this bus: <string: BUS_NAME>

$CAN frequency

CAN Frequency: <int: CAN_FREQ>

$CAN-FD frequency

CAN FD Frequency: <int: FD_FREQ>

$the program's faults isolation algorithm interval. This value affects the sensitivity of detection. The lower the rate, the more sensitive the algorithm.

fault isolation interval: <double: INTERVAL_RATE_IN_SECONDS>

$the SnS device transmits a message to the bus to detect and locate short circuits. These messages must be provided with an available msg ID that is not owned by any other ECU in the described topology.

short frame ID: <int: AVALIBLE_MSG_ID>

$This is the matrix of ECUs in the described topology. They should be sorted by their real position on the bus, relative to one of the terminations.

$Note that the SnS's position is mentioned as well by dedicating a row in the matrix that states "SnS," and nothing else.

$Each column is separated by a comma. First element is the name of the ECU and the rest of the columns are message IDs transmitted by this ECU.

$each line can have a different number of columns indicating that individual ECUs "own" different numbers of messages

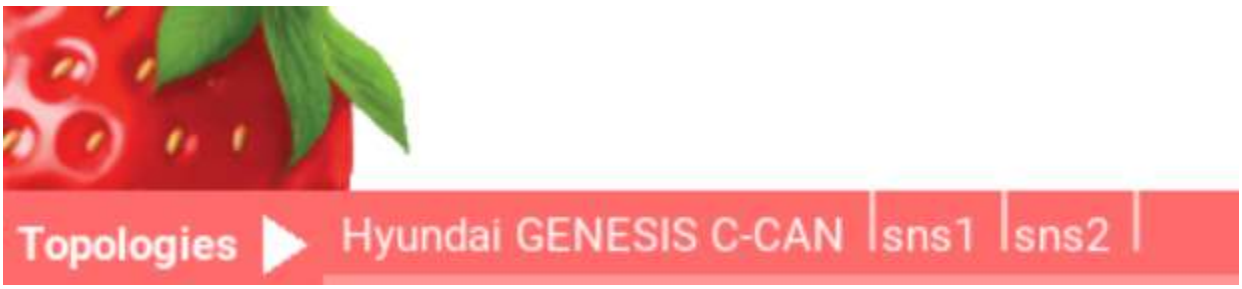
<string: ECU_NAME_1>, <hex, MSG_ID_1>, <hex, MSG_ID_2>, …. <hex, MSG_ID_n>
<string: ECU_NAME_2>, <hex, MSG_ID_1>, <hex, MSG_ID_2>, …. <hex, MSG_ID_n>
. . . . . .
. . . . . .
SnS,
. . . . . .

. . . . . .

<string: ECU_NAME_n>, <hex, MSG_ID_1>, <hex, MSG_ID_2>, ..., <hex, MSG_ID_n>

Notes:

- Topology files are stored in .../PhysiCAN/Topologies. (Topologies' directory inside the application's main directory)
- To create a topology file simply write it with your favorite editor and change its extension to .topo, then move the file to the Topologies directory.
- A topology file template can be found in the Topologies directory named topo_example.txt
- Note that files without .topo extension in Topologies directory are not recognized by the application.
- In the app, available topo files can be found in the lower red bar, next to the "Topologies" label followed by a white triangle.



Available topo files can be found in the lower red bar, next to the "Topologies'' label followed by a white triangle.

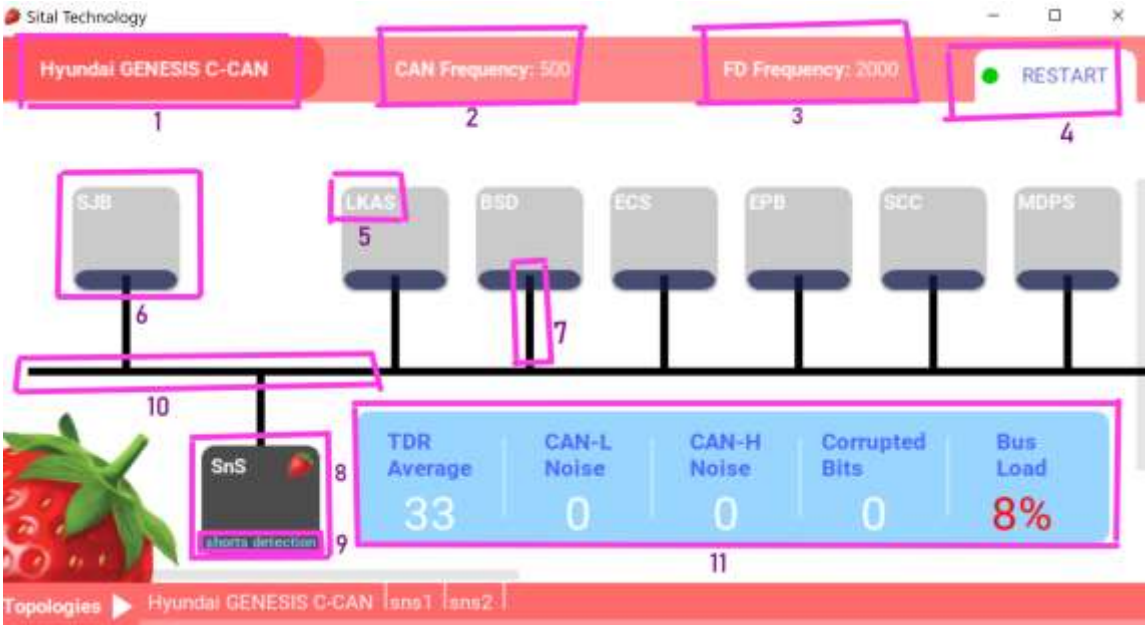
UI Overview

Upper bar [indexes 1-4 in the picture]:

The upper bar shows general information about the topology and the PhysiCAN connection state. The *1*name of the topology is on the left, and its *2*CAN Frequency and *3*CAN FD Frequency are followed to the right. To the right of the frequencies' labels on the upper bar, there is a *4*white tab. The state of connection with a PhysiCAN device is deduced from the white tab. If it displays a red LED and a label 'SnS not found', no device is connected to the app. If the LED appears green and the label turns to a 'RESTART' button, the device is connected successfully. Pressing the RESTART button will restart the connection with the device and clean any previous fault indications from the topology diagram. If no topology is selected, the upper bar won't display anything.

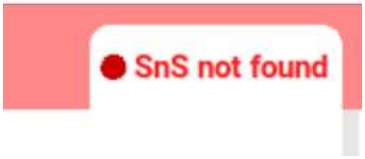
Topology diagram [indexes 5- 9 in the picture]:

The PhysiCAN app draws a topology diagram based on a topo file. This diagram represents the CAN-Bus that is currently being monitored. The diagram consists of: ***10***main bus-line - a horizontal black line that represents the main bus. ***7***Stub - a vertical black line that connects the main bus to a CAN-node (ECU). It represents a physical stub.***6***ECU - a grey square, connected by a stub to the main bus-line, and positioned above the main bus-line. It represents a physical ECU (CAN-node). The ECU is ***5***labeled by its name and shows the message IDs it owns (only when the mouse hovers above the ECU). ***8***SnS - a dark grey square, connected by a stub to the main bus-line. Positioned under the main bus-line. It represents the physical SnS device connected to the bus and to the computer. The SnS has an "SnS" label on it, a little strawberry icon, and a ***9***"shorts detection" label that changes colors from light grey to light blue. The shorts detection label color change indicates the SnS's transmission state (blue- active, grey-passive). ***11*** info panel - see next section for more information.



DEVICE CONNECTION AND THE INFO PANEL

In order to connect to a PhysiCAN, simply connect it to the computer with a USB cable. In the app, a connection LED indicator is located on a white tab on the right corner of the upper bar. If the app doesn't recognize any PhysiCAN device, the LED will turn red, and a flashing red label 'SnS not found' will appear next to it. Following successful connection, the LED indicator will turn green, and a RESTART button will show up next to it. This RESTART button will initialize the connection with the device and clean any previous open and short circuit fault indications from the topology diagram . Note that it takes a second or two for the app to find and connect to PhysiCAN.



Another important element that appears following a successful connection with a PhysiCAN device is the Info Panel. The Info Panel is a large light blue widget located on the lower right corner of the topology diagram. It shows some important metrics, measured by PhysiCAN in real-time. The following is a list of the metrics displayed in Info Panel:

1. **TDR average** -

   TDR is a measure of the deviation of a signal's voltage transition time from the time it should take according to the CAN bus standard. The TDR average is the average TDR value of all messages from all nodes. Significant TDR average means that most messages are transmitted poorly and are often linked with circuit faults.

2. **CAN-L Noise** -

   CAN-L Noise is a count of CAN-low signals that surpassed 2.5v or dropped to less than -2.5v.

3. **CAN-H Noise** -

   CAN-H Noise is a count of CAN-high signals that surpassed 2.5v or dropped to less than -2.5v.

4. **Corrupted Bits** -

   Corrupted bits count is a count of bits with non-standard bit width.

Bus Load -

Percentage of bus time utilization

These 5 metrics are key elements for the diagnosis of a faulted CAN-Bus.

FAULTS DETECTION

Fault detection using the PhysiCAN app is done by looking at the topology diagram and the info panel. When the bus contains faults, the 5 metrics on the info panel are used to classify their type. For instance, unusual TDR average and CAN-low noise may suggest CAN-low wire disconnection on the main bus. The following table enumerates the different types of faults that can be deduced out of the info panel, and characterizes their metrics values.

fault type	TDR average	CAN-H noise	CAN-L noise	visual indicator on diagram
Main bus CAN-dual wire disconnection	~(>40)	0	0	a single or a sequence of ECUs are red
Main bus CAN-high wire disconnection	~(>40)	(>0)	0	a single or a sequence of ECUs are red
Main bus CAN-low wire disconnection	~(>40)	0	(>0)	a single or a sequence of ECUs are red
Stub CAN-dual wire disconnection	normal	0	0	an ECU is red
Stub CAN-high wire disconnection	normal	(>0)	0	an ECU is red
Stub CAN-low wire disconnection	normal	0	(>0)	an ECU is red

Understanding the topology diagram:

When a topology is successfully loaded and the PhysiCAN is successfully connected to the host computer, the CAN-Bus and the various ECUs change their color. ECUs may be green, yellow, or red. Green indicates that the ECU transmits messages on every *fault isolation interval** without exceptions. A red ECU means that it did not transmit any message at the current faults isolation interval. If an ECU is yellow, it means that it transmitted a message on the current fault isolation interval, but did not transmit any message on some previous intervals. Once the metrics on the info panel suggests some kind of fault, these color changes on the topology diagram can help locate them. For example, if the metrics suggest a stub CAN-dual wire disconnection, and there is an ECU that is constantly red, it might mean that this ECU is disconnected from the bus. If the metrics suggest a CAN-dual wire disconnection on the main bus and the diagram shows a sequence of red ECUs, the fault may be between the first red ECU in the sequence and last green [or yellow] ECU relative to the position of the SnS.

**Faults isolation interval - a period in which all messages are stored in the SnS device for a learning algorithm to isolate received messages from the knowingly unreceived messages. The interval rate is configured in the topology file.*

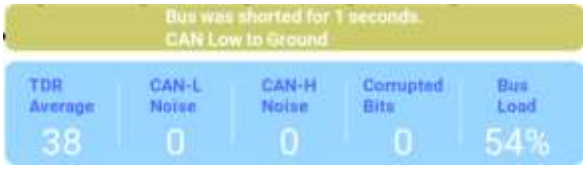
SHORTS DETECTION

Short circuit detection by the app is an internally automated process. When the app determines the occurrence of a short circuit, it notifies the user with a banner next to the Info Panel. The banner will include the short's type and timing. If the short is in real time, i.e., the bus is currently shorted, then the banner will be red. If an intermittent short was detected and passed, the banner will be yellow. The following table shows all the possible types of short circuits and how they appear on-screen:

CAN-H to power short:

A CAN-high to power short is currently undetectable programmatically. This kind of short is quite rare. This short should be suspected when no other short or fault is found, the bus is dead (all ECUs are red or grey), CAN-L Noise is at maximum value (255) and the Bus Load is indicating 99%.

intermittent short appearance:



Shorts detection activity indicator:

On the SnS widget in the topology diagram there is a small 'shorts detection' label. This label is blue when the algorithm suspects a short, that means that the PhysiCAN device transmits messages to the bus to confirm its suspicions. If the algorithm doesn't suspect any shorts, the label would be grey and it means that the device is currently passive (doesn't transmit messages).

<-- passive

active -->

FUTURE FEATURES

KNOWN ISSUE

Record bus activity: Tell the SnS device to record the activity for later inspection. This operation yields a csv file of recorded messages and SnS PARAMS, and a log file of suspected faults, shorts, and cyber-attacks.

Usage of bus recordings: Bus diagnosis in retrospective using recorded bus activity

Topology file wizard: Easily create .topo files

Cybersecurity: Cybersecurity capabilities, such as detecting authentication violations (impersonation or "spoofing") and preventing DoS (Denial-of-Service) attacks by transmitting nodes.

Stub and main bus color faults indication: Color change the representation of the wire themselves

Short detection history database: Detected shorts are saved in an event log

ECU TDR on widget: Each ECU widget will display real time TDR value

ECUs with low transmission rate might alternate between red and yellow: Fault isolation basically checks for each known message and whether it was transmitted within a certain time window. This means that if there's an ECU that transmits at a relatively slow rate and within a fault isolation interval, no instance of the message was transmitted, the ECU would then be categorized as red. Since this is not necessarily a fault, the ECU would soon turn back to yellow.

Topo files error prone:

The topology files interpreter is pretty basic and therefore is prone to errors.

Aggressive use of the restart button kills the program:

The restart button directs the program to do some heavy computation and thus if overused, it will crash it.

CAN-low to ground short distance indication isn't meaningful:

CAN-low-to-Ground shorts indicate incorrect distance value