ColdCAN Instructions

We are supplying the ability to examine the software using multiple different ways to make it easier for people unfamiliar with Linux. Directly below you will find the Dev Setup details. If you are not wanting to set up the libraries on your own Linux laptop please skip this step. We have made the software available using a Linux cloud desktop so you do not need to actually set up your own system if you do not want.

Dev Setup

If you would like to do development on the software and test on a linux setup we would recommend starting by setting up a computer using a linux distro. The development and testing of the ColdCAN software was done using Ubuntu 20.04.1 LTS but will work on a majority of linux flavors. This can be completed using a virtual machine if you are not willing or wanting to override an OS or dual boot. Once Linux is set up and the software is placed onto the filesystem go into the "_Setup" folder and run the following command:

```
./DevSetup
```

This will install the required components to do development and testing of the software on your Linux device. Linux is required because it contains SocketCAN which is a package that contains the necessary components to run the CAN protocol.

Software Setup:

Since the J1939 simulator requires a Linux OS with some rather specialized kernel modules, Team ColdCAN has provided you with a few different ways to experience our accomplishments.

1. The first option is to log into a Linux cloud desktop instance that we have built on AWS for you. With this option there are two further options that can be utilized. The software is located on this desktop and can be built and run with some commands provided below. We will also be supplying the executables for the program on this desktop so the software can be run with the created executables
2. The second option is to locally configure a Linux environment to include the packages and kernel modules necessary to build and launch our program. We will provide you the links for the software that is needed to complete this step.

Option 1 – Accessing via Linux cloud desktop:

Logging in:

1. Go to: https://34.217.115.129/
2. Enter Password: i-0c108c1d69cb27bdb

Building Executables:

NOTE: The executables are already on the desktop and this step can be skipped. Completing this step will just show the packaging software that was added to create the executables.

1. Open a terminal
2. Change directory to ColdCAN directory
   cd ColdCAN
3. Execute makeBuilds script
   ./makeBuilds
4. Executables will be located within the newly created builds directory located at ~/ColdCAN/builds/
5. These files can be run with the following commands if you want to run the newly created executables: “~/ColdCAN/builds/sim” and “~/ColdCAN/builds/read”

Execute simulator:

1. To execute the simulator, open a console window and run:
   $ sim

   There is an alias in the ~/.bash_aliases file that will run the sim program from the desktop.
2. To execute the receiver, open another console window and run:
   $ read

   There is an alias in the ~/.bash_aliases file that will run the read program from the desktop.
3. Now the simulator and the receiver are opened and running on the desktop. Each will require you to open a configuration file. Both of these are located on the desktop each labeled for the software that they are running. If you are running sim select the SimConfig.json file, if you are running read select the ReaderConfig.json file.
4. Optional: If you are running the receiver you can select a file that you can save the received content to. To do this just follow the instructions for the file manager and name the file whatever you want and save it in the location you want.
5. Once the configurations are loaded the simulator and receiver will be running for you to use.
6. When finished, please close the sim and read windows to stop the programs from running, and then close the browser tab.

Option 2 – Configure a local environment:

Building Environment:

Like discussed within the Dev Setup area once a Linux OS is set up, download the repository to the computer.

1. Using a terminal navigate to the unzipped or downloaded software
2. cd into the _Setup directory
3. Run the following command: “./DevSetup”
4. The command will now install the required development components onto your desktop to be able to build the executables or run the software

Note:

In a few odd cases your distro might not have the correct modules to support CAN, if you do get a fatal error upon execution that mentions kernel modules, please run the following command.

```
sudo apt-get install linux-modules-extra-$(uname -r)
```

Building Executables:

1. Open a terminal
2. Change directory to ColdCAN directory
   ```
   cd “path”
   ```
3. Execute makeBuilds script
   ```
   ./makeBuilds
   ```
4. Executables will be located within the newly created builds directory

Execute simulator:

This should be completed after the executables are built as it requires the executables to run.

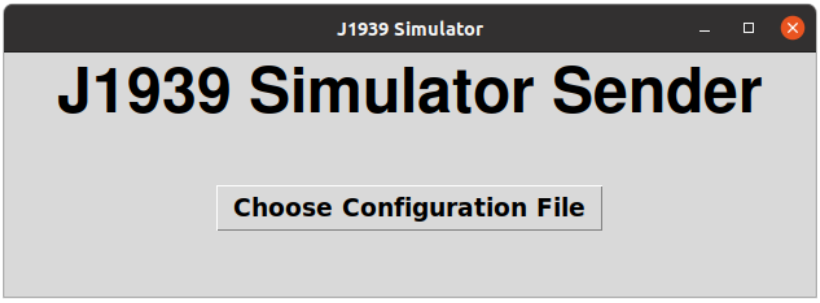
1. Open terminal
2. Change directory to ColdCAN directory
   ```
   cd “path”
   ```
3. Change directory to builds directory
   ```
   cd builds
   ```
4. Execute the simulator
   ```
   ./sim
   ```
5. Open another terminal
6. Change directory to ColdCAN directory
   ```
   cd “path”
   ```
7. Change directory to builds directory
   ```
   cd builds
   ```
8. Execute the simulator
   ```
   ./read
   ```

Running the Software

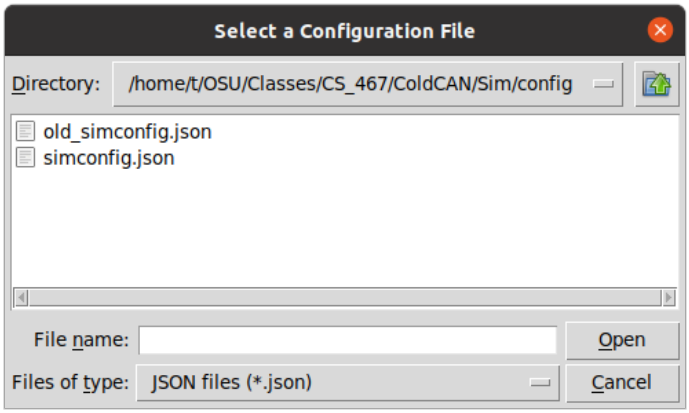
Now that the environment is set up and the software is built (if needed and are running the executables) the user can activate the software. We will run through the two different applications individually with the same config files so you will see the data matching.

Simulator

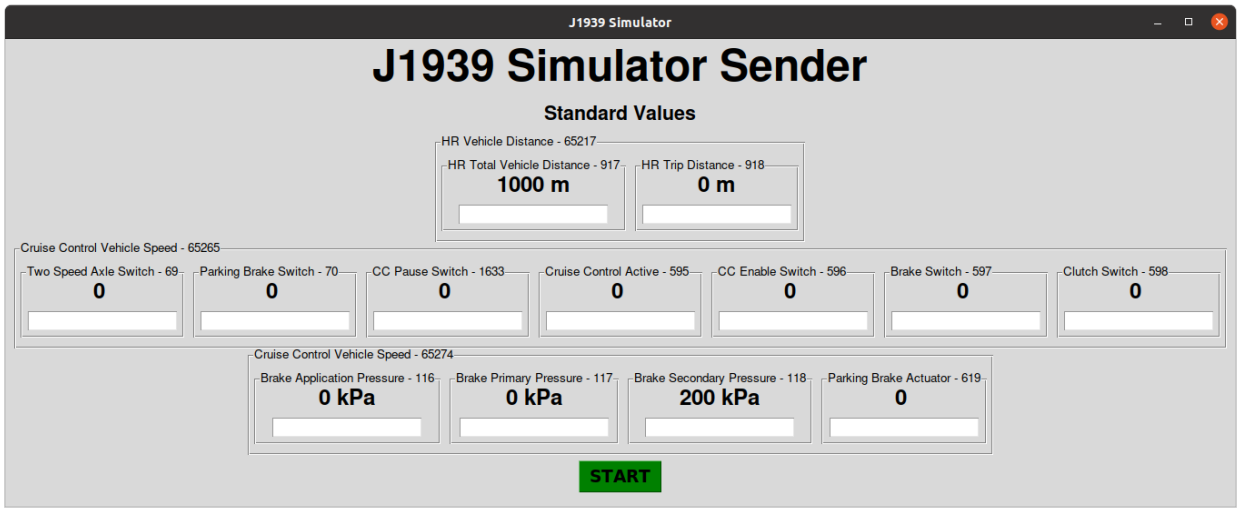
When the simulator software is run the user will likely need to provide their laptop password due to the sudo requirements, then will be brought to a screen to load the configuration file that will be run for the program.



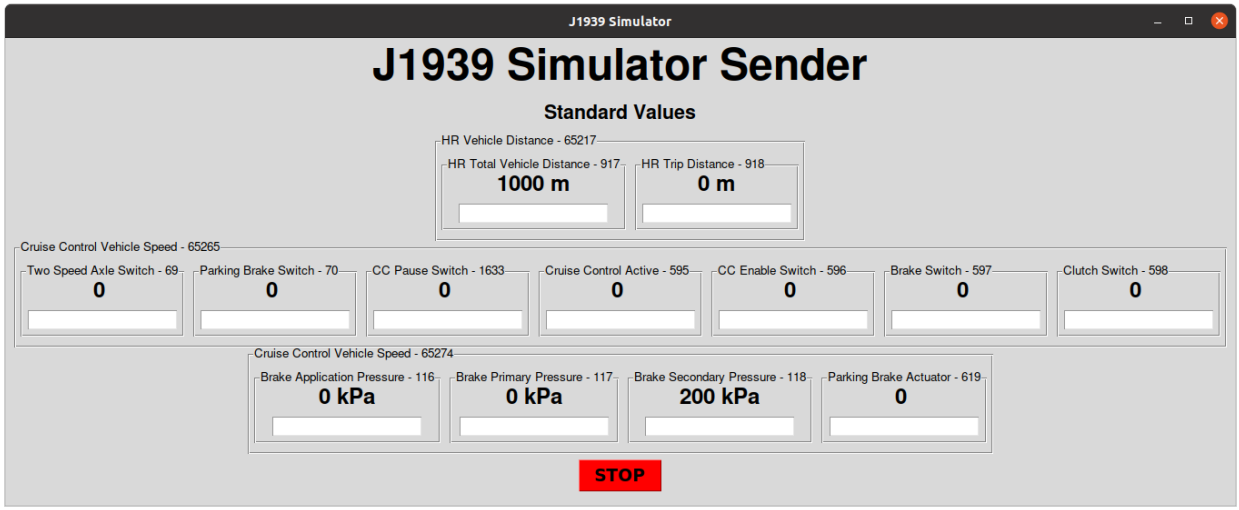
When choosing the configuration file there can be multiple stored in one location, just grab the corresponding simconfig.json file that you are looking for and open it.

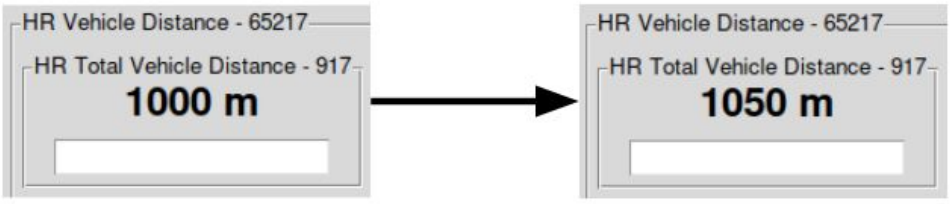


The simulator will now bring the user to the main page that has the details from the configuration file loaded to display the necessary PGN/SPN combinations and the current values for those parameters. The simulator is not doing anything at this point, it just loaded the config, initialized the data in the lower layers, and set up the vcan interface. But it will sit here and stay idle until the START button is pressed.

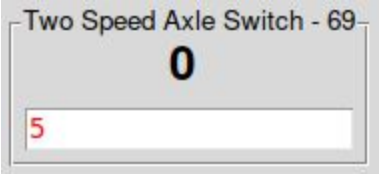


When START is pressed nothing really changes except the button changes from start to stop. At this time the software is waiting for changes to be entered into the input boxes. When values are entered there the values above the boxes will change and the change request will be sent down to the lower layers to process. If the user enters a value that is of a greater precision than accepted by the parameter, the UI will adjust the value to the nearest acceptable value.





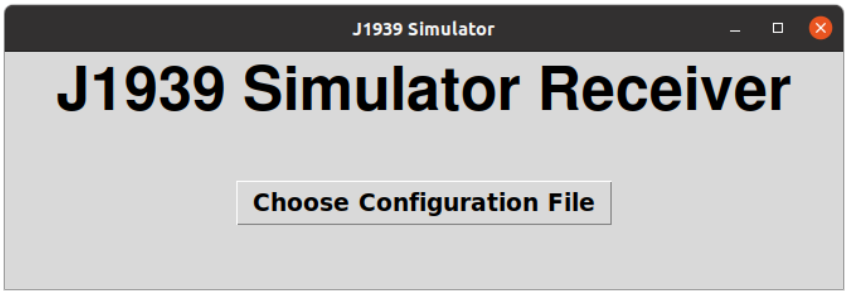
If the user enters in a value that is invalid the software will flag the value with red lettering. Here in this example the value has a low value set to 0 and a high value set to 1. So it is basically a boolean value. Here we are attempting to set the value to 5; since we cannot, we flagged it as red so the user corrects the input.



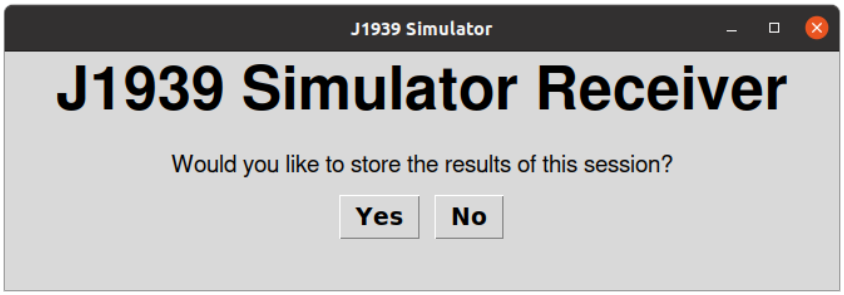
At this time any valid changes to the values will be sent down to the lower levels where they will send it out to the bus. This will happen continually until the software is stopped and closed. The user could at that time load a new configuration file to play new data to simulate some completely separate engine data all while utilizing the same base software.

Receiver/Reader

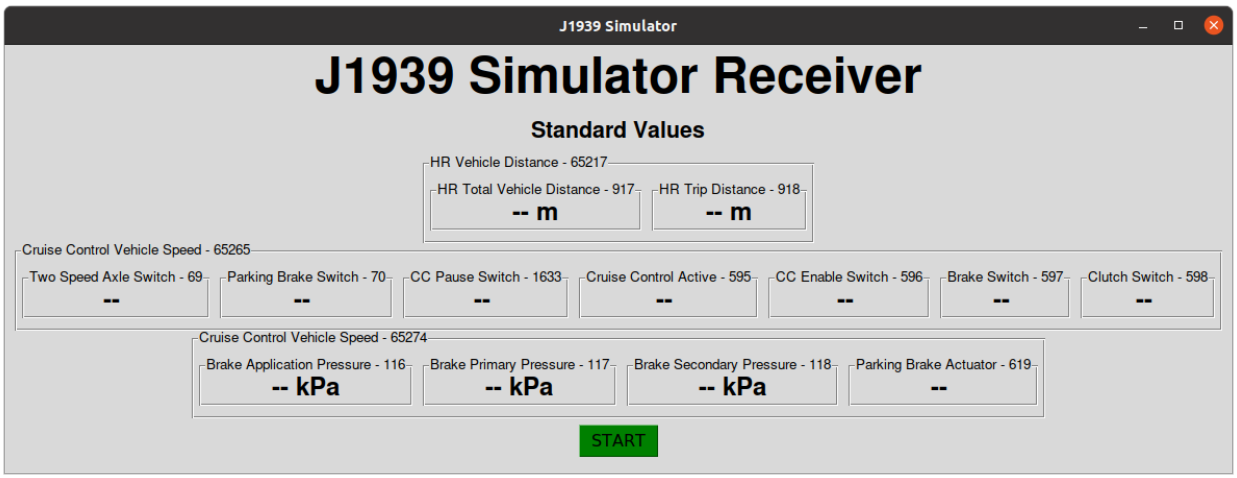
The receiver software is very similar to the simulator in that the user will likely need to provide their laptop password due to the sudo requirements, then will be brought to a screen to load the configuration file that will be run for the program.



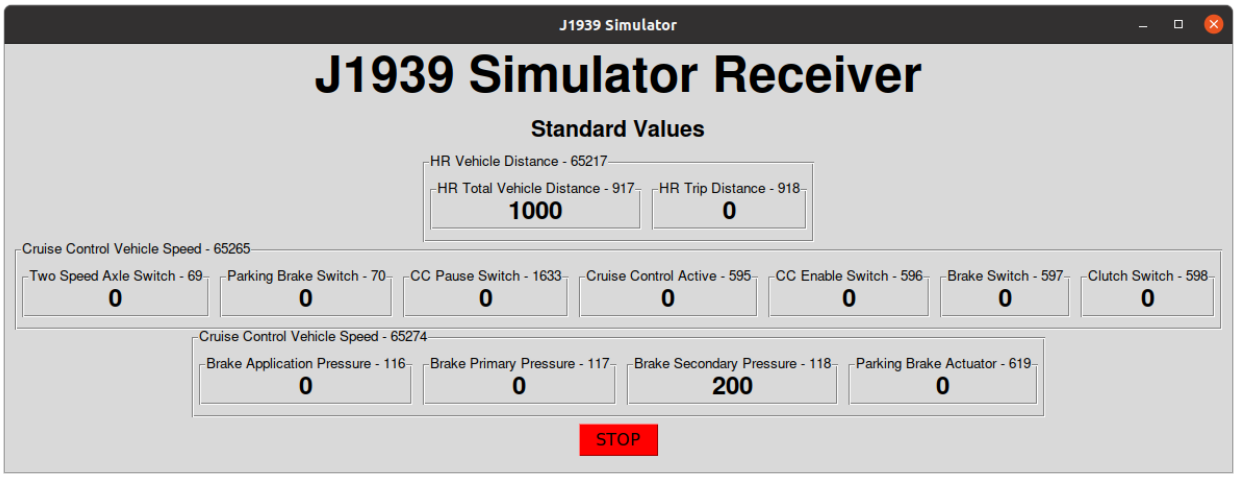
Here, the configuration loading is the same as the simulator, the receiver will have a new selection option after the configuration file. This box will ask if the user will want to save off the received data to a separate file. If the user selects yes they will be brought to another screen to enter in the filename they want to use and where to store it, if they select no it will just continue to the loading screen.

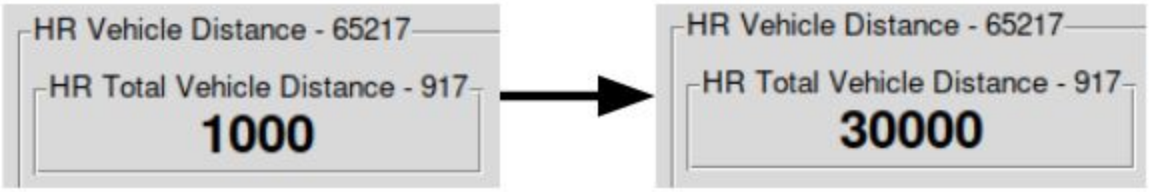


At the loading screen it looks very similar to the simulator. It also has a start stop button at the bottom and will use the configuration file to load the values. As you can see that before the values are received there are no values to start with so the data is all empty.



After starting the receiving software and receiving some content from the simulator the window will now show what the received values are below the SPN details. This looks exactly like the simulator but without an input box. As the values change we will see these values change showing what is currently being received at any given point in time.





This will continue until the simulator is stopped and shut down. At that time the user could load a different configuration file to receive and test different values. A user of this data would potentially connect this to a bus that has hundreds of values processing continually. This software will break out the specific values they care about so they can observe what happens.