Honda Civic CAN Bus Logger

Introduction

This wiki page describes how to sniff on a car's CAN bus and hence log the current speed of the car. This wiki describes the procedure to log data specifically from a Honda Civic Hybrid car, model 2008. However, since Honda Civic Hybrid cars manufactured between the years 2006 and 2009 (both inclusive) use almost identical components, therefore, it is safe to assume that this procedure can be applied to those models too.

This procedure can also be easily extended to any vehicle that uses CAN bus for data exchange between different modules of the car. More specifically the wheel speed sensors or the Power-train Control Module (PCM) must transmit the actual speed of the vehicle on the CAN bus. There is one condition though and that is the car's CAN bus data must be available on the Data Link Connector.

Hardware Requirements

Background Information

There is not much required to connect to a car's internal CAN bus. Ever since on-board diagnostics (OBD) has become a complementary feature of modern cars, there have been many different types of devices manufactured to help automotive workshops diagnose any malfunction in the car. ELM Electronics, however manufactured a chip which was called ELM327 that made the life easier for most of the manufactures who wanted to focus more on diagnosing the car and let someone else take care of the hard work required to do the translation between a CAN bus and their own hardware. ELM327 did so by providing a UART interface on the user's end and taking care of the CAN and other OBD protocols by itself.

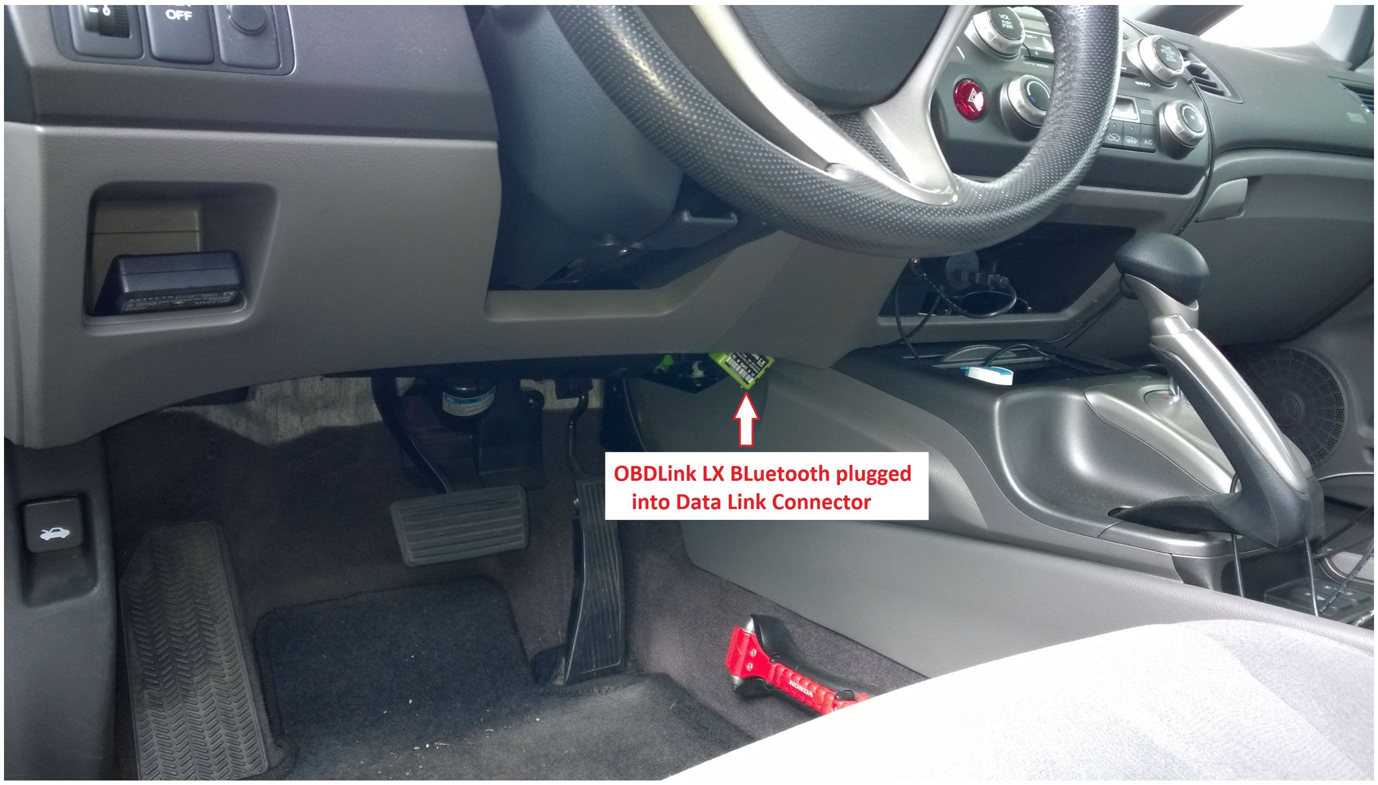
Recently OBD Solutions, another company based in USA, manufactured another chip which is called STN1110. This chip supports all of the ELM327 commands and adds more ST commands of its own. Moreover, it is faster than ELM327 and provides more space for buffering data on-chip. OBD Solutions also manufacture different devices that use the STN1110 chip internally. There are currently four different variants of their devices available, namely OBDLink LX Bluetooth, OBDLink MX Bluetooth, OBDLink MX WiFi, and OBDLink SX USB. All these devices provide identical functionality but expose different interfaces (on the UART side) to external hardware namely USB, Bluetooth, and WiFi. More information is available on manufacture's website http://www.obdlink.com/

The OBDLink LX Bluetooth

The specific device that was used to tap into the CAN bus of Honda Civic was the OBDLink LX Bluetooth.

The Data Link Connector (DLC)

The Data Link Connector (DLC), also known as OBD-II socket is where the OBDLink device needs to be plugged in. In Honda Civic Hybrid models 2006~09, the DLC connector is located under the dashboard, on the right hand side of the steering wheel. Here is a picture showing the location of the connector with an OBDLink LX plugged into it.



Software Requirements

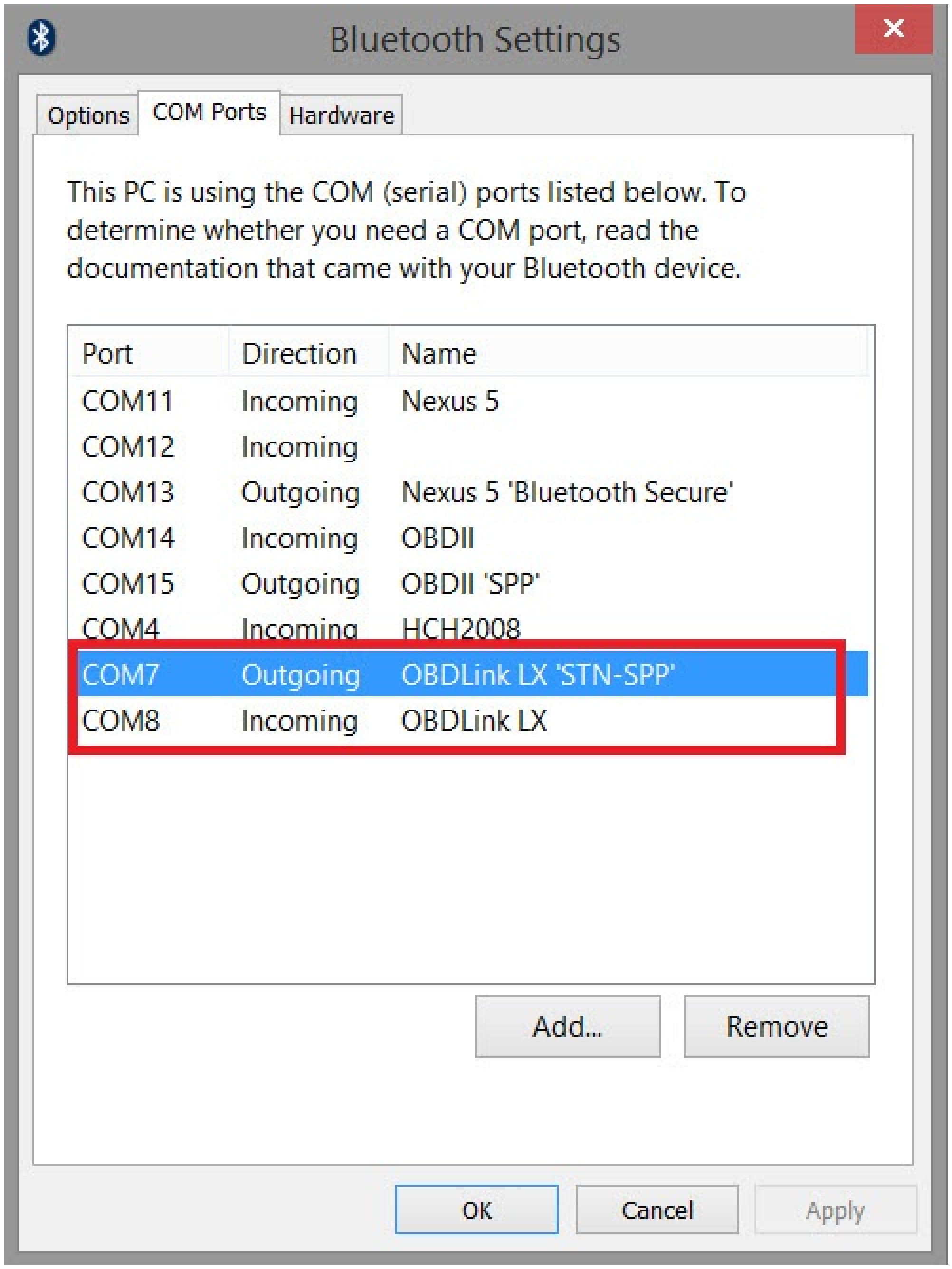
The Basics

The software can, in theory, be built on any platform e.g. Windows, Mac, Linux, Android, etc. as long as the operating system provides a way to communicate with hardware over USB, Bluetooth, or a WiFi link. The following lines may apply only to the OBDLink LX/MX Bluetooth devices.

Note: Apple iPhone devices as of this writing are unable to communicate with Bluetooth versions of OBDLink devices, and hence it is required to use OBDLink MX WiFi if the software needs to be run on an iPhone.

Microsoft Windows 7, and 8.

Before starting to communicate with the OBDLink LX, the device needs to be paired with PC running Windows 7 or 8. After Windows finishes installing the right drivers for this device, two COM ports will be installed on PC, as shown in the screenshot below. The port that has "Outgoing" listed under Direction is the one that Windows will need to make use of in order to communicate with the device. For instance, COM7 is the right port in the screenshot below.


Bluetooth Settings

This PC is using the COM (serial) ports listed below. To determine whether you need a COM port, read the documentation that came with your Bluetooth device.

Port	Direction	Name
COM11	Incoming	Nexus 5
COM12	Incoming	
COM13	Outgoing	Nexus 5 'Bluetooth Secure'
COM14	Incoming	OBDII
COM15	Outgoing	OBDII 'SPP'
COM4	Incoming	HCH2008
COM7	Outgoing	OBDLink LX 'STN-SPP'
COM8	Incoming	OBDLink LX

Windows Phone 8.x and Android

In order for an app to communicate with OBDLink from an Android phone or a Nokia Lumia phone with Windows 8.0 or 8.1, the app needs to make use of the Bluetooth Serial Port Profile (SPP). The screen of the mobile device also needs to stay unlocked in order to keep the Bluetooth link alive.

The Communication Protocol

ELM327 and OBDLink series devices use AT commands that need to be sent to these devices from host computer or mobile phone in order to interact with a vehicle. The ELM327 datasheet lists all the commands that these devices support. However, OBDLink provides a set of additional commands which start with ST, more about these commands can be found in STN1100 datasheet.

These devices work in a command-response mode. After the serial port is opened on PC, or a Bluetooth SPP port is opened on PC or mobile device, the first command can be sent to the device, which is usually ATI to initialize the device. The device responds with an echo of the command, and it's version number, ELM327 v1.xx. In order for this device to work properly, further commands should only be sent after receiving response to the previous command.

Every command needs to be terminated with a CR (carriage return) character. Likewise, every single line of the response ends with CR+NL (new line).

The Honda CAN bus

There are four different variants of the CAN bus protocol in use in vehicles today. They differ only in baud rate and the length of the message identifier. Honda Civic Hybrid 2006~09 operate the CAN bus at 500 Kbaud, and use 11-bit message identifiers.

Sample session with Honda Civic Hybrid 2008

Following is a sample command/response session with the OBDLink device to collect speed data off the CAN bus. Bold face letters are the commands sent to the device, the rest are the responses back from device. After STM command is transmitted to the device, the device will continuously monitor the CAN bus for messages and will keep sending the filtered messages to the host software until being interrupted by the host PC.

```
ATI
ATI
ELM327 v1.3a
>ATH1
ATH1
OK
>ATAL
ATAL
OK
>ATSP6
ATSP6
OK
>ATS0
ATS0
OK
>STFCP
STFCP
OK
>STFAP 158, FFF
STFAP 158, FFF
OK
>STM
STM
15802BF000002CD303D
15802BE000002C63026
15802C1000002CD300D
```

and so on...

How to interpret the CAN bus data

Thanks to OBDLink, the CAN protocol layer has been taken care of, and the only data that the software receives is the payload (including the message ID) of the CAN messages. The specific message that carries the wheel speed sensors data has the message ID 158 (HEX).

Let us assume that we received the following message from the CAN bus.

15806720000068F3734

Here is how to get the speed values out of it.

158 **0672** 0000 **068F** 3734

The first three nibbles, parsing the message from left to right, are the message ID. i.e. 158 (in HEX notation)

The next two bytes are the speed data from one speed sensor, i.e. 0672 => 1650, divide it by 100 and we get 16.50, that is the current speed in km/h recorded by one of the four speed sensors of the vehicle.

Leave the next two bytes alone, they are of no use to us.

Come to the next two bytes which are 068F. Convert it to decimal and then divide it by 100, we will get 16.79 km/h, that is the speed recorded by the wheel speed sensor located on the other side of the car.

The last two bytes can also be discarded since they are of no use to us.

One thing to note here is that there is a slight difference between the speeds recorded by the two sensors attached to different wheels and that happens when the car is making a turn, so one wheel is turning faster than the other.