Flex Board Meets OBD

Alessio Balsini

Università degli Studi di Pisa
Scuola Superiore Sant'Anna
a.balsini@sssup.it

Abstract

The aim of the project is the development of a system which retrieves and presents to the user diagnostic data from a vehicle. This system is made of Flex Board developed by Evidence, Bluetooth-to-UART module and a Bluetooth Elm327 device. Information gathered are presented to the user by an LCD display embedded in the Flex Board or by an interactive graphical interface on PC.

I. Introduction

Nowadays almost all the vehicles adopt, by law, a standard way of communicating diagnostic information.

The OBDII standard is actually the most common protocol. It allows the user not only to retrieve diagnostic data, but also to program the control units and modify vehicle's parameters in real time.

II. Hardware

II.1 Modules Involved

Many interfaces for transfering vehicle diagnostics are not directly available in common PCs or PDAs.

As bridge between OBDII and Bluetooth it is possible to use Elm327 modules.

Figure 1: *Elm 327 Bluetooth/OBDII module*

By default, our Elm327 Bluetooth module is discovered with the Bluetooth name "CHX" and requires the pin code "6789" for pairing. Another common configuration for this kind of devices is a Bluetoooth name similar to "OBDII" and pin code "1234".

The Flex Light is the core of the project.

It comes with a dsPIC developed by Microchip and is programmed with Microchip's MPLAB ICD tools.

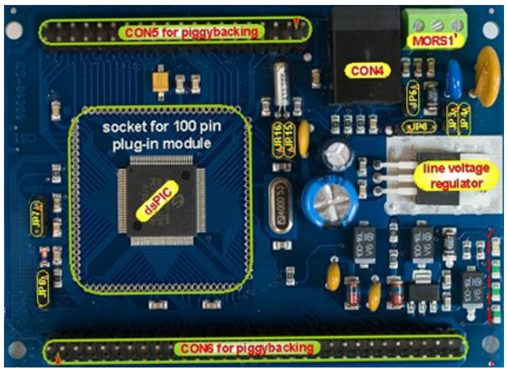


Figure 2: *Flex Light board*

The communication between Light Board and PC is made possible by a USB-UART converter.

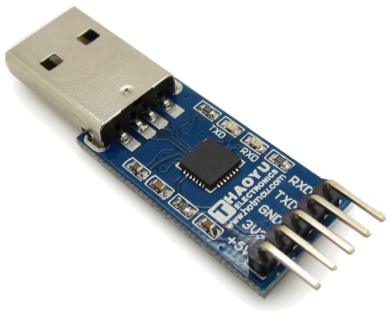


Figure 3: *USB-UART converter*

Flex Light functionalities can be extended with many different daughter boards. One of these is the so called Flex Demo Board.

In particular, for this project, will be used the LCD screen, push buttons and the UART connector provided by the Demo Board.

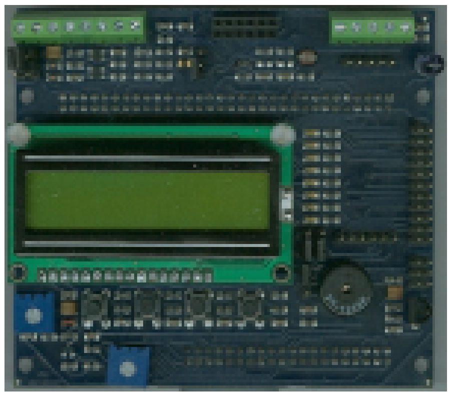


Figure 4: *Flex Demo board*

The Bluetooth communication is allowed by the BlueSmirf, a little board which embeds the Roving Networks RN-42 module.

Figure 5: *BlueSmirf Bluetooth module*

This module acts as a Bluetooth communication manager and as a converter of the data transferred within Bluetooth and UART port.

II.2 Connectors and Wirings

The Flex Light Board pins involved in the project, from CON6 (see the Flex Light Base Board datasheet) are the following:

- pin 37: UART 2 input;
- pin 39: UART 1 output;
- pin 40: UART 2 output;
- pin 42: UART 1 input;
- pin 55: Vout (5 V);
- pin 56: Vout (5 V).
- pin 59: GND.
- pin 60: GND.

The Flex Demo Board pins involved in the project are the following:

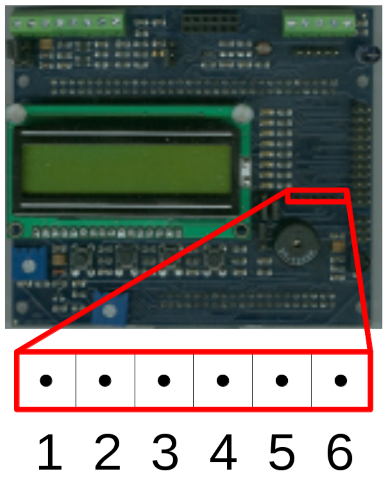


Figure 6: *Flex Demo Board pinout*

- pin 1: 5 V;
- pin 2: TX-O, UART output;
- pin 3: SCK, Serial ClocK;
- pin 4: RX-I, UART input;
- pin 5: i.e. Vout;
- pin 6: i.e. GND.

The Bluetooth device uses has the following pinout:

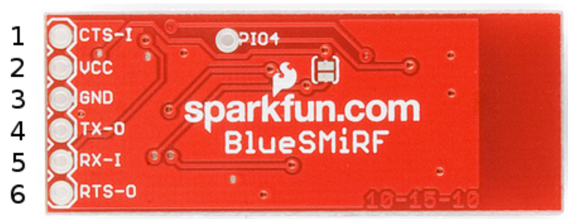


Figure 7: *Bluesmirf (Bluetooth/UART module) pinout*

- pin 1: CTS-I, Clear to Send (handshake pin);
- pin 2: VCC (from 3.3V to 6V);
- pin 3: GND;
- pin 4: TX-O, UART output;
- pin 5: RX-I, UART input;
- pin 6: RTS-O, Request to Send (handshake pin).

The Bluesmirf module has been connected to the Flex Demo Board with the following wiring:

- Flex RX-I <-> Bt TX-O;
- Flex TX-O <-> Bt RX-I;
- Flex 5 V <-> Bt Vcc;
- Flex GND <-> Bt GND.

The Bluesmirf module can be connected with the Flex Light with the same logic, choosing UART 1 or 2 depending on the program's constants definitions.

And the same approach must be followed during the commection between the Flex Light Board and the USB-UART module.

The standard OBDII female connector has the following pinout:

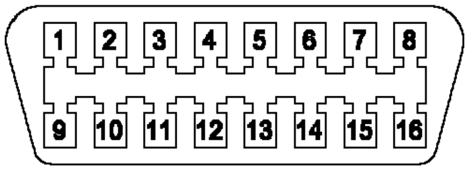


Figure 8: *OBD female connector pinout*

- pin 1: NC;
- pin 2: Bus+ (J1850);
- pin 3: NC;
- pin 4: GND (chassis);
- pin 5: GND (signal);
- pin 6: CAN High (J-2284);
- pin 7: K-Line (ISO9141-2);
- pin 8: NC;
- pin 9: NC;
- pin 10: Bus (J1850);
- pin 11: NC;
- pin 12: NC;
- pin 13: NC;
- pin 14: CAN Low (J-2284);
- pin 15: L-Line (ISO9141-2);
- pin 16: Board Power (12 V).

III. Background

III.1 Elm327 Communication Basics

Elm327 accepts basically two types of input:

- commands: allow various kind of operations on device;
- data requests: allow to retrieve data from the vehicle.

Elm327 is case-insensitive and does not care about spacing symbols (tabs or white spaces). Nevertheless, in this document will be used uppercases and a correct amount of spacing, with the aim of improving readability.

III.1.1 Basic Commands

All commands are prefixed with "AT". It is possible to distinguish between four classes of commands:

- general commands;
- OBD commands;
- protocols specific commands;
- miscellaneous commands.

The syntax for commands is, usally

```
AT CMD [param]
```

```
AT Z
```

Resets the module, as if a power reset happened. It brings back all the settings to their default values. At reboot, module prints out the device name and firmware version, like *ELM327 v1.4*.

```
AT SP x
```

Sets the protocol to *x*, in particular, by default:

- 0 automatic;
- 1 SAE J1850 PWM (41.6 Kbaud);
- 2 SAE J1850 VPW (10.4 Kbaud);
- 3 ISO 9141-2 (5 baud init, 10.4 Kbaud);
- 4 ISO 14230-4 KWP (5 baud init, 10.4 Kbaud);
- 5 ISO 14230-4 KWP (fast init, 10.4 Kbaud);
- 6 ISO 15765-4 CAN (11 bit ID, 500 Kbaud);
- 7 ISO 15765-4 CAN (29 bit ID, 500 Kbaud);
- 8 ISO 15765-4 CAN (11 bit ID, 250 Kbaud);
- 9 ISO 15765-4 CAN (29 bit ID, 250 Kbaud);
- A SAE J1939 CAN (29 bit ID, 250 Kbaud);
- B USER1 CAN (11 bit ID, 125 Kbaud);
- C USER2 CAN (11 bit ID, 50 Kbaud).

III.1.2 Basic Data

Each data request has three parameters associated:

- mode: hexadecimal 2 byte identifier;
- pid: hexadecimal 2 byte identifier;
- response length: response bytes.

Each request has the following syntax

```
MODE PID
```

and each response has the syntax

```
MODE PID VALUE
```

where *VALUE* has different fixed sizes for different data. *VALUE*'s length is n-byte, each byte separated by a space symbol.

```
00 PID_REQ
```

where

$$PID\_REQ = 0x10 \cdot k, \quad k = 0,1,2,4 \qquad (1)$$

Requests the list of pids available.

The value returned is a bit map, where each bit is associated to a pid and has value:

- 1: available;
- 0: not available.

```
01 01
```

Requests the list of error codes (*DTCs*, *Diagnostic Trouble Codes*). This returns a 4 byte response, like

```
41 01 AA BB CC DD
```

where:

- *41 01*: is the response to the request;
- *AA*: is the number of trouble codes:
  - most significant bit: Multifunction Indicator Lamp (MIL, aka Check Engine Light) has been turned on;
  - remaining bits: number of stored trouble codes, flagged in the ECU (Engine Control Unit);
- *BB CC DD*: information about the availability and completeness of certain on-board tests, with different interpretations in case of spark or compression ignition engines. More details on SAE J1979 documentation.

So, with the *AA* value set to *81* hex (*10000001* bin) it is possible to recognize that there is one trouble code stored which turned on the Check Engine Light.

BB

Test	Available	Incomplete
Misfire	B0	B4
Fuel System	B1	B5
Components	B2	B6

CC, DD in Spark Ignition

Test	Avail.	Incompl.
Catalyst	C0	D0
Heated Catalyst	C1	D1
Evaporative System	C2	D2
Secondary Air System	C3	D3
A/C Refrigerant	C4	D4
Oxygen Sensor	C5	D5
Oxygen Sensor Heater	C6	D6
EGR System	C7	D7

CC, DD in Compression Ignition

Test	Avail.	Incompl.
NMHC Cat	C0	D0
NOx/SCR Monitor	C1	D1
Boost Pressure	C3	D3
Exhaust Gas Sensor	C5	D5
PM filter monitoring	C6	D6
EGR and/or VVT System	C7	D7

IV. FlexMeetsOBD

IV.1 PC and Flex Light

IV.1.1 Flex Light

There's not so mucht to say about the Flex Light. After the wiring, it just needs to be powered up. The PC interface will do the rest.

IV.1.2 Overview

When the PC software is launched, the user will obtain the welcome window shown in figure.

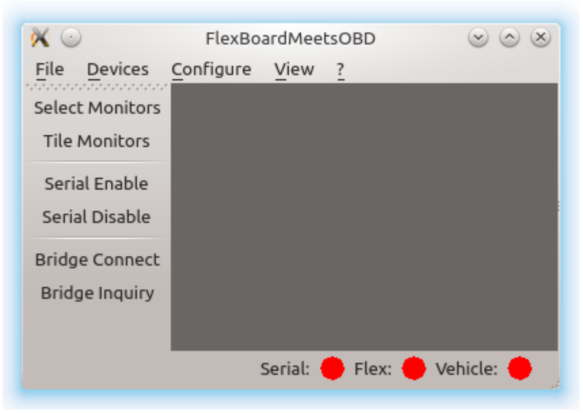


Figure 9: *Welcome window*

Monitors are sub-windows that will be drawn inside the dark grey region of the main window. Each monitor is used to represent the vehicle data associated to it and some additional information.

From each monitor is possible to open a new window containing it's plot diagram, which shows the last received samples.

For monitors management, click on *View* > *Select Monitors* or directly on *Select Monitors* on the right bar menu.

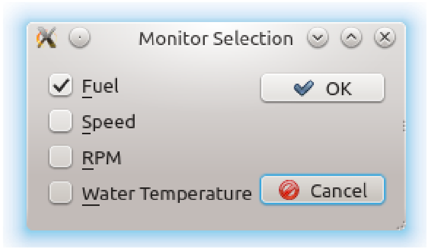


Figure 10: *Monitor selection window*

Once the monitors are chosen, the main window will be populated.

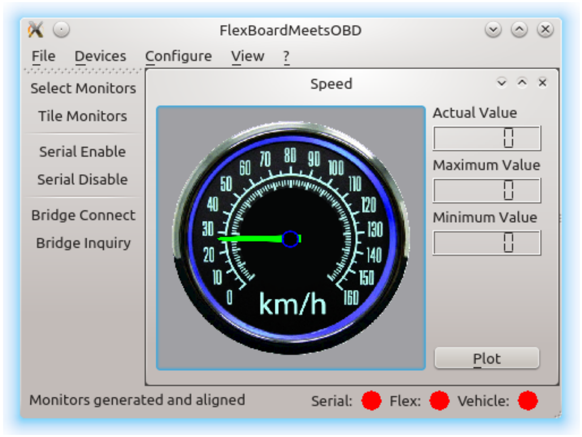


Figure 11: *Mainwindow after monitor creation*

At this point, monitor has inconsistent values.

It is possible to move monitors inside the window. To make an automatic monitors alignment, click on *View* > *Tile Monitors* or on *Tile Monitors* on the left bar.

IV.1.3 Connections

It's time to estabilish the connections.

The first step is to configure the serial port, through *Configure* > *Serial Configuration*.

Remember to set appropriate permissions to the file descriptor associated to the serial port in */dev/* folder.

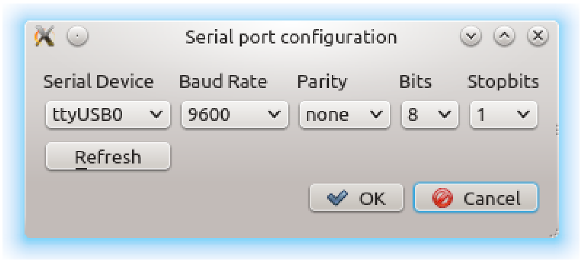


Figure 12: *Serial port configuration window*

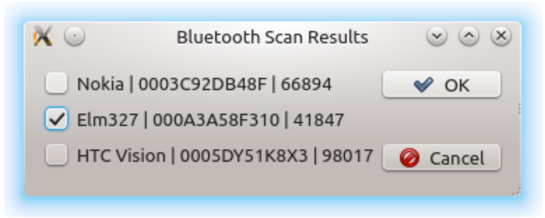
The coloured dots in the status bar, in bottom right of the window, show the status of the Serial, Flex and Vehicle links. Pay attention to them, in order to detect if something is going wrong.

To enable [disable] serial device, click on *Devices > Serial Enable [Disable]* or on *Serial Enable [Disable]* on the left bar. The Serial dot should become green, otherwise, check if the serial device is properly connected and permissions are correct.

To establish the connection with the Flex, click on *Devices > Bridge Connect* or on *Bridge Connect* on the left bar. The Flex dot should become green, otherwise, check if the serial configuration parameters are compatible with Flex's ones, Flex is powered on, Flex is in a consistent state and Flex is correctly connected to PC.

Not it is possible to request the Flex to perform a Bluetooth inquiry, searching for the Elm327 device.

Inquiry results are presented to the user, after clicking on *Devices > Bridge Inquiry* or on *Bridge Inquiry* on the left bar.



Nokia | 0003C92DB48F | 66894
Elm327 | 000A3A58F310 | 41847
HTC Vision | 0005DY51K8X3 | 98017

Figure 13: *Bluetooth scan results*

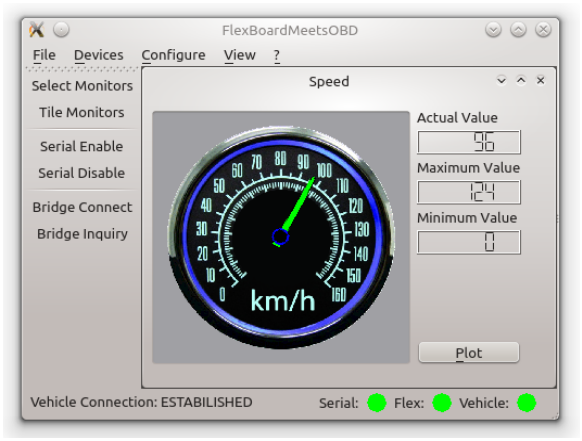
Select the preferred device in list and click *Ok*.

If the device is not in list, click *Cancel*, check if Flex and Elm327 are correctly working and perform another inquiry.

IV.1.4 Samples Presentation

Once the Elm327 device is chosen inside the inquiry list, data Flex will periodically send data to PC.

This data will be shown to the user similarly to how is shown in figures.



Actual Value 96
Maximum Value 124
Minimum Value 0
Vehicle Connection: ESTABILISHED

Figure 14: *Monitor appearance while sampling*

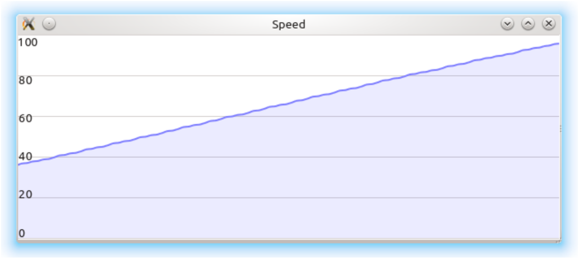


Figure 15: *Plot appearance while sampling*

It is always possible to add new monitors or remove monitors from main window, also while sampling.

IV.2 Flex Demo Board Standalone

After powerup, user receives a welcome message.

```
 Flex2OBD 0.1
   WELCOME!
```

Press any key to continue.

Now Flex is going to:

- initialize the UART port;
- initialize the RN-42 module;
- setup the Bluetooth connection parameters;
- perform an inquiry.

```
BT Init...[Done]
SetMode...[Done]
Inquiry...[Done]
```

If no device is found, push any key to rescan.

Otherwise, inquiry results will be shown to the user with an interactive menu.

```
Devices found: 3
@ Elm327
```

It is possible to navigate the menu using the buttons, with a vi-like meaning:

- button 0: left;
- button 1: down;
- button 2: up;
- button 3: right.

Going up or down shows the devices found.

Pressing left or right navigates from the module's properties: from left to right, name, address and class.

Before the Bluetooth module's name is shown an @ symbol, meaning that pressing left will estabilish a connection with that device.

After the Bluetooth module's class is shown a *!* symbol, meaning that pressing the right key will perform another inquiry.

After connection request, another menu will appear.

```
Estabilish Conn.
Comm|RConn|RScan
```

By pressing one of the two middle buttons, the Flex will attempt to establish another connection with the device.

By pressing the last button will be performed another inquiry scan.

By pressing first button, the Flex will start the communication with the Elm327, setting up the module, choosing the vehicle's protocol and requesting periodically the vehicle data. Before pressing this button, check if the Bluesmirf's led is green, otherwise something went wrong during the connection.

If connection is established, the vehicle's data will appear on the screen, with another dynamic menu.

```
Elm Vers:   ???
RPM:
Speed:
Cool.Temp.:
Fuel:
```

Once data is ready, the fields will be filled, for example:

```
Elm Vers:  1.5
RPM:        800
Speed:        0
Cool.Temp.:  72
Fuel:        48
```

It is possible to scroll the menu by pressing first two buttons (scroll down) of the last two buttons (scroll up).

Many thanks goes to my friend Giovanni C., who bravely allowed me to use his new OBDII compatible car for some days and making it possible to perform all the testing phase of this project.

References

[1] A. S. Huang, L. Rudolph, *Bluetooth Essentials for Programmers*, Cambridge University Press (2007)

[2] *Bluetooth Data Module Command Reference & Advanced Information User's Guide*, Roving Networks (2013)

[3] *ELM327 - OBD to RS232 Interpreter*, Elm Electronics