# On-Board Diagnostic Module

Girish R,24878,M.Tech Electronic Systems Engineering
Sourish Vijayamadhavan,24140,M.Tech Electronic Systems Engineering, DESE, IISc

## I OBJECTIVE

Develop a compact and programmable system capable of monitoring vehicle parameters, detecting faults through the OBD-II interface, and displaying real-time diagnostic information.

## II SYSTEM ARCHITECTURE

The system consists of a OBD Emulator and OBD Scanner, OBD Emulator is the module that is been built up with Python and TIVA to provide responses and emulate the car behavior, while the OBD scanner consists of TIVA along with Touch LCD to display the parameters and data.The Emulator and scanner communicates CAN with OBD protocol in application layer.

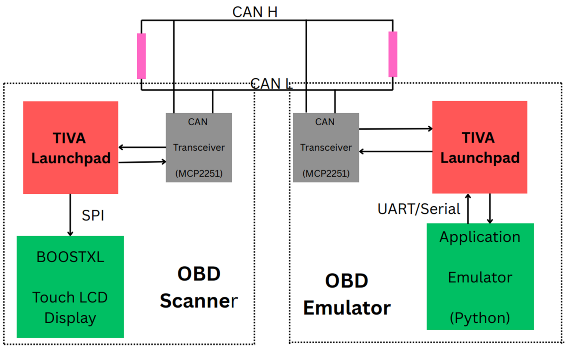

CAN H
CAN L
TIVA Launchpad
CAN Transceiver (MCP2251)
SPI
BOOSTXL Touch LCD Display
OBD Scanner
CAN Transceiver (MCP2251)
TIVA Launchpad
UART/Serial
Application Emulator (Python)
OBD Emulator

Figure 1: Complete system architecture showing (1) TIVA microcontroller with CAN interface, (2) Kentec QVGA touchscreen display, (3) Python emulator communication, and (4) Vehicle ECU connection via CAN Bus

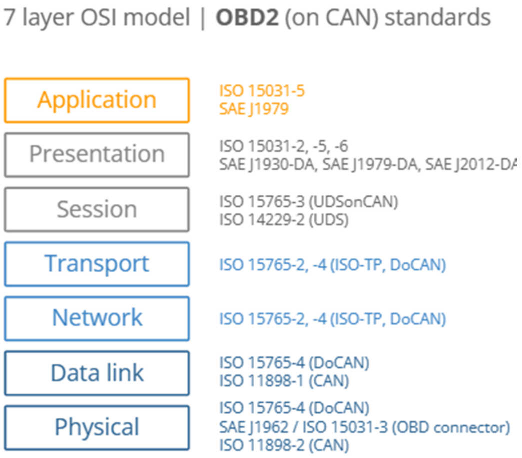
## III DESIGN IMPLEMENTATION

### *3.1 Hardware Components*

- **TIVA TM4C123GH6PM** microcontroller with CAN 2.0B controller
- **MCP2551** CAN transceiver implementing ISO 11898-2
- **Kentec QVGA BoosterPack** with SSD2119 controller (320x240 resistive touch)

### *3.2 Software Components*

- Embedded firmware with OBD-II protocol stack (ISO 15765-4)
- Python emulator using CustomTkinter and pyserial
- GRAM display driver with grlib graphics library



| 7 layer OSI model | **OBD2** (on CAN) standards |
|---|---|
| Application | ISO 15031-5<br>SAE J1979 |
| Presentation | ISO 15031-2, -5, -6<br>SAE J1930-DA, SAE J1979-DA, SAE J2012-DA |
| Session | ISO 15765-3 (UDSonCAN)<br>ISO 14229-2 (UDS) |
| Transport | ISO 15765-2, -4 (ISO-TP, DoCAN) |
| Network | ISO 15765-2, -4 (ISO-TP, DoCAN) |
| Data link | ISO 15765-4 (DoCAN)<br>ISO 11898-1 (CAN) |
| Physical | ISO 15765-4 (DoCAN)<br>SAE J1962 / ISO 15031-3 (OBD connector)<br>ISO 11898-2 (CAN) |

Figure 2: OBD-II protocol stack implementation showing physical layer (CAN), transport layer (ISO-TP), and application layer (OBD services)

## IV TECHNICAL SPECIFICATIONS

### *4.1 CAN Communication*

Table 1: CAN Message Identifiers

| Hex ID | Purpose |
|---|---|
| 0x7DF | Broadcast requests |
| 0x7E0-0x7E7 | ECU-specific requests |
| 0x7E8-0x7EF | ECU responses |

### *4.2 Diagnostic Services*

Table 2: Implemented OBD-II Services

| Mode | Description |
|---|---|
| 01 | Show current data |
| 03 | Read stored DTCs |
| 04 | Clear DTCs |

### 4.3 Parameter Conversion

Table 3: Parameter Conversion Formulas

| Parameter | Conversion |
|---|---|
| Engine RPM | $((A << 8) \| B)/4$ |
| Vehicle Speed | $A$ (km/h) |
| Fuel Pressure | $3A$ (kPa) |
| MAF | $((A << 8) \| B) * 0.01$ (g/s) |

### 4.4 Packet Format

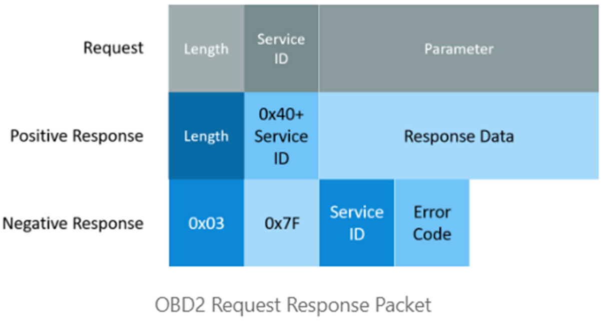


Figure 3: OBD Request and Response packet format over CAN

### 4.5 DTC format

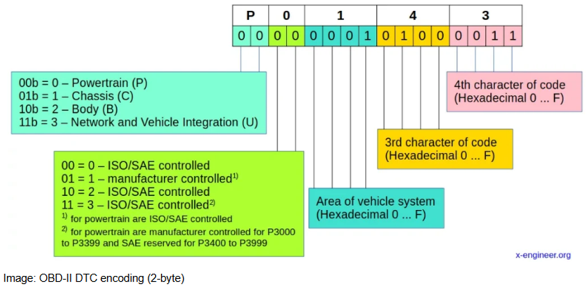

P
0
1
4
3
0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 1
00b = 0 – Powertrain (P)
01b = 1 – Chassis (C)
10b = 2 – Body (B)
11b = 3 – Network and Vehicle Integration (U)
00 = 0 – ISO/SAE controlled
01 = 1 – manufacturer controlled1)
10 = 2 – ISO/SAE controlled
11 = 3 – ISO/SAE controlled2)
1) for powertrain are ISO/SAE controlled
2) for powertrain are manufacturer controlled for P3000 to P3399 and SAE reserved for P3400 to P3999
Area of vehicle system (Hexadecimal 0 ... F)
3rd character of code (Hexadecimal 0 ... F)
4th character of code (Hexadecimal 0 ... F)
x-engineer.org
Image: OBD-II DTC encoding (2-byte)

Figure 4: Diagnostic Trobule Code format encoding in 2 bytes

## V RESULTS

### 5.1 OBD Scanner

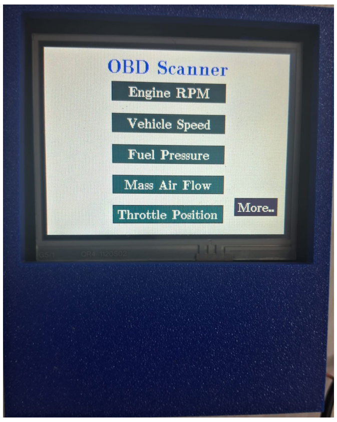


Figure 5: Display interface - showing list of PIDs available in the OBD module

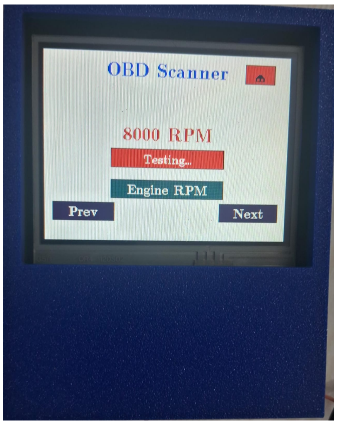


Figure 6: Mode 01 - Showing real-time data for Engine RPM

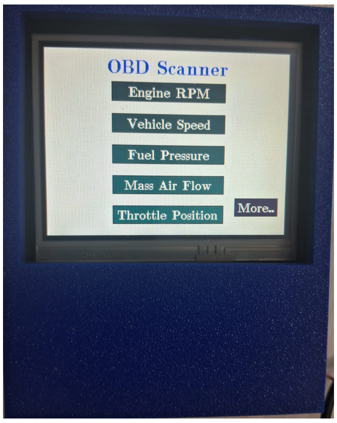


Figure 7: Display interface - showing list of PIDs available in the OBD module

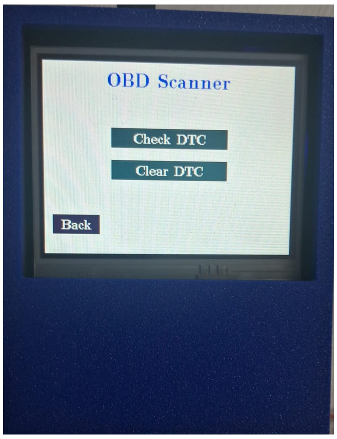
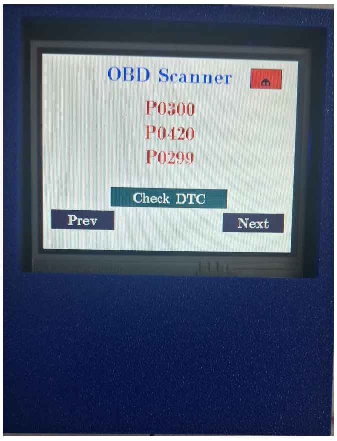


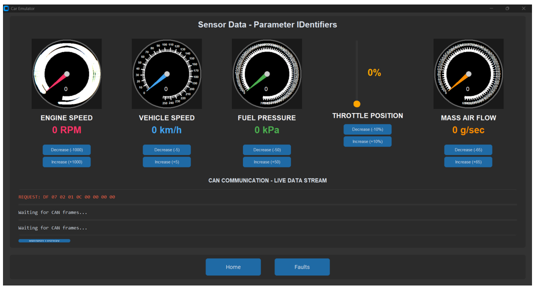
Figure 8: Mode 03 and 04 support buttons in OBD scanner



P0300
P0420
P0299

Figure 9: Display interface - Showing the stored DTCs in the ECU memory

### 5.2 Emulator results and Screens



ENGINE SPEED
0 RPM

VEHICLE SPEED
0 km/h

FUEL PRESSURE
0 kPa

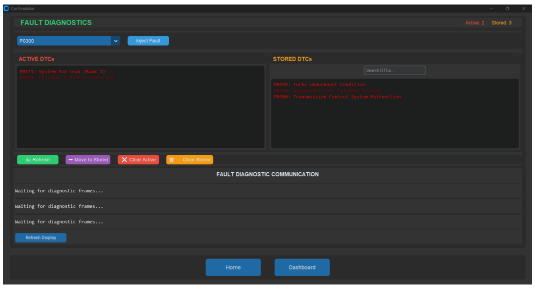
THROTTLE POSITION
0%

MASS AIR FLOW
0 g/sec

CAN COMMUNICATION - LIVE DATA STREAM

```
REQUEST: DF 07 02 01 0C 00 00 00 00
Waiting for CAN frames...
Waiting for CAN frames...
```

Figure 10: Python emulator interface showing Live data simulation along with CAN message monitoring



Active: 2 Stored: 3

ACTIVE DTCs

```
P0171: System Too Lean (Bank 1)
P0301: Cylinder 1 Misfire Detected
```

STORED DTCs

```
P0299: Turbo Underboost Condition
P0300: Random/Multiple Cylinder Misfire
P0700: Transmission Control System Malfunction
```

FAULT DIAGNOSTIC COMMUNICATION

```
Waiting for diagnostic frames...
Waiting for diagnostic frames...
Waiting for diagnostic frames...
```

Figure 11: Python emulator interface showing Fault Management along with CAN message monitoring

### 5.3 Test Data

| Request | Response | Parameter Value |
|---|---|---|
| 7DF [01 0C] | 7E8 [41 0C 1A F8] | 6904 RPM |
| 7DF [01 0D] | 7E8 [41 0D 3C] | 60 km/h |
| 7DF [03 00] | 7E8 [43 01 33 00] | P0133 DTC |

## VI CONCLUSIONS

The developed OBD-II diagnostic system has two components with the features achieved given below

### 6.1 OBD Scanner

- Real-time monitoring of 5+ vehicle parameters
- Complete DTC management (read/clear)
- Interactive touchscreen interface
- Accurate emulation of ECU responses

### 6.2 OBD Emulator

- User interactive car emulation software for 5 PIDs
- Enhanced Fault Management System creation that provides visualization how fault memory is managed in ECU.
- Complete plug and play interface with CAN capability.
- Modifyable and enhancable design software wise.