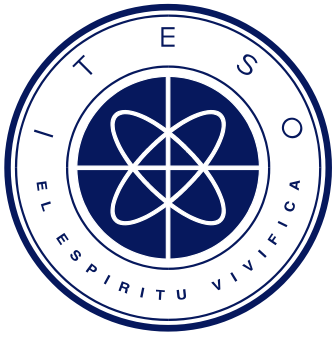
INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE OCCIDENTE

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación el 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática

ESPECIALIDAD EN SISTEMAS EMBEBIDOS



INTERFAZ GRÁFICA EMBEBIDA DE CLUSTER AUTOMOTRIZ

Trabajo recepcional para obtener el diploma de

ESPECIALISTA EN SISTEMAS EMBEBIDOS

Presenta: Raúl Antonio Camacho Mendoza

Tutor: M.C. Héctor Antonio Rivas Silva

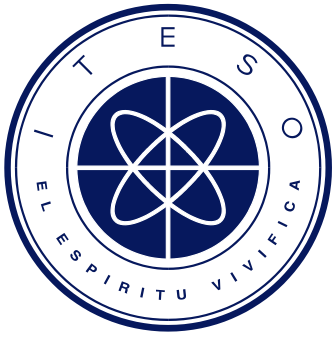
Tlaquepaque, Jalisco. Enero de 2017.

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE OCCIDENTE

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación el 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática

ESPECIALIDAD EN SISTEMAS EMBEBIDOS



INTERFAZ GRÁFICA EMBEBIDA DE CLUSTER AUTOMOTRIZ

Trabajo recepcional para obtener el diploma de

ESPECIALISTA EN SISTEMAS EMBEBIDOS

Presenta: Raúl Antonio Camacho Mendoza
CVU 636170 –No. Beca Conacyt 337394

Tutor: M.C. Héctor Antonio Rivas Silva

Convocatoria de Becas Nacionales 2014 Primer Periodo 290915

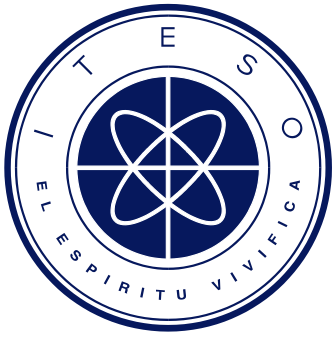
Tlaquepaque, Jalisco. Enero de 2017.

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE OCCIDENTE

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación el 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática

ESPECIALIDAD EN SISTEMAS EMBEBIDOS



EMBEDDED GRAPHIC USER INTERFACE FOR AUTOMOTIVE CLUSTER

Final project reportto obtain the diploma of

EMBEDDED SYSTEMS SPECIALIST

Presents: Raúl Antonio Camacho Mendoza
CVU 636170 – Conacyt Scholarship No. 337394

Advisor: M.Sc. Héctor Antonio Rivas Silva

Tlaquepaque, Jalisco. Enero de 2017.

Table of Contents

Agradecimientos

Antes que nada quiero expresar mi más sincero agradecimiento a mi familia por el apoyo incondicional que recibí en todo momento durante el curso de este programa. Ustedes son la base y soporte que tengo día con día para lograr mis objetivos tanto profesionales como personales.

Agradezco a la institución (ITESO) por crear este programa de especialización que me ha permitido desarrollar mis habilidades hacia un campo que realmente me apasiona y sobre todo lleno de oportunidades. A mis maestros Héctor Rivas, Abraham Tezmol y Francisco Martínez por brindar sus conocimientos y experiencia, logrando que sus clases fueran algo especial.

A mis compañeros y amigos agradezco su paciencia y soporte que tuve durante el desarrollo de este trabajo.

Finalmente, agradezco de manera muy especial al **CONACYT** por creer en este programa y brindar un apoyo invaluable a cada uno de los alumnos que cursan este programa.

Acknowledgements

First all, I would like to express my sincere thanks and gratitude to my family for giving me unconditional support when I was studying this specialization program. You are the base and support that I have day-to-day to achieve my professional and personal goals.

I thank the institution (ITESO) for creating this specialization program that allowed me to develop my skills towards a field that I really enjoy and has a lot of opportunities. To my teachers Héctor Rivas, Abraham Tezmol and Francisco Martínez for provide their knowledge and experience, making their classes something very special.

I would like to extend my thanks to my classmates and friends for their patient and support I had during the development of this work.

Finally, I thank in a very special way the **CONACYT** for believing in this program and give his invaluable support to every student enrolled in this program.

Resumen

Dentro de un automóvil, el dispositivo más importante para la comunicación e interacción entre el vehículo y conductor es el clúster de instrumentos. Este debe ser lo bastante claro y simple para facilitar el entendimiento e interpretación de toda la información relacionada al comportamiento del vehículo.

El presente trabajo explica la implementación de una Interfaz Gráfica Embebida de un Clúster Automotriz mediante la utilización de las librerías gráficas Qt instaladas en un sistema operativo Linux optimizado y personalizado, el cual corre en la microcomputadora de bajo costo Raspberry Pi. Asimismo, se expone la implementación de Memoria Compartida (IPC) utilizada para la comunicación entre la aplicación de la Interfaz Gráfica y el middleware.

Debido a lo anterior, el presente trabajo fue realizado siguiendo el Modelo-V de desarrollo de software, el cual toma en cuenta las etapas de identificación de requerimientos, definición de la arquitectura, desarrollo y pruebas. De esta manera, se facilitó el mantenimiento y actualización del mismo, a tal grado que puede utilizarse como plataforma base para futuros proyectos.

Por último, se muestra la implementación de un “middleware” simple, el cual permite probar y demostrar la funcionalidad de la Interfaz Gráficadel Clúster Automotriz.

Abstract

In a car, the most important device for the communication and interaction between vehicle and driver is the instrument cluster. It must be sufficiently clear and simple to facilitate the understanding and interpretation of all the displayed information related to the vehicle behavior.

This work explains the implementation of an embedded Graphic User Interface (GUI) for an automotive cluster using the Qt graphics libraries installed on an optimized and customized Linux operative system, which runs on a low-cost microcomputer Raspberry Pi. Additionally, the implementation of the Shared Memory for the Inter-Process Communication (IPC) between the Graphic Interface application and the middleware is exposed.

In the same way, this work was realized following the V-Model software development methodology, which takes into account the requirements identification, architecture definition, development and testing stages. Thus, the maintenance and updating of this work was simplified into such degree that it can be usedas a platform for future projects.

Finally, the implementation of a simple middleware that allows testing the embedded Graphic User Interface functionality is shown.

i. List of Figures

ii. List of Tables

iii. Acronyms

- OS - Operative System.
- GUI - Graphic User Interface.
- OEM - Original Equipment Manufacturer.
- ASPICE - Automotive Software Process Improvement Capability Determination
- ECU - Electronic Control Unit.
- HMI - Human Machine Interface.
- LCD - Liquid Crystal Display.
- CAN - Controller Area Network.
- SPI - Serial Peripheral Interface.
- I2C - Inter-Integrated Circuit.
- LIN - Local Interconnection Network.
- AUTOSAR - Automotive Open System Architecture.
- RPi - Raspberry Pi

1 Introduction

This work describes the proposed implementation of an embedded graphic user interface for automotive cluster. It focuses in the stages of implementation from the client's requirements to the test designed for the functional validation. Following one of the most used methodologies on the automotive industry for projects development, all the stages paired with the requirements through documentation and implementation were tracked.

This document is divided into specific chapters designed to detail each stage of the development process and make the reader follow along the same procedures as anengineer would take while creating it.

1.1 Statement of the Problem

Nowadays, Mexico is one of the most important places in the world where the automotive industry has a big impact on the research and design fields. However, only foreign companies develop and implement embedded automotive clusters. None of the Mexican companies or universities that are involved on the automotive industry had been working in the development of embedded automotive clusters.

1.2 Justification

Due to the embedded systems specialization provided the tools on the most common elements used on the automotive industry, such as design and implementation of low level drivers, implementation of different operative systems like Embedded Linux OS and FreeRTOS, implementation of communication protocols like CAN and LIN and plenty more ideas. The next logical step is to make use of most of them and create an embedded automotive cluster that could evolve into a real piece of commercial technology.

Thus, as a result of the design and implementation of this project, it will be the very first step of hopefully more to come, in the pursuit of a fully functional automotive cluster created by ITESO students.

Finally, this is a very rewarding project, because not only creates experience in the automotive industry, also enables creative thinking problem and problem solving. As a consequence, it will allow to the new generations to upgrade all the proposed implementation in hope of a better product and hopefully with the same expectations of future colleagues.

1.3 Project Objectives

1.3.1 General Objective

Propose a new embedded system platform (hardware and software) in order to create an automotive cluster.

1.3.2 Specific Objectives

a) Use an open source embedded OS for robustness and portability.

b) Perform the software application using only free development tools to avoid cost.
c) Develop the project based on an open source hardware platform.
d) Create a functional prototype capable to communicate with an automotive communication protocol.
e) “Frontend” and “Backend” system creation.
f) Apply all the possible knowledge learned on the embedded systems specialization.
g) Create an embedded system platform capable of being improved and updated by future student generations.
h) Being one of the first universities to create an automotive cluster prototype.

1.4 Scope

This project arises due the absence of national automotive technology. Thus, this work will contribute to solve this problematic creating an automotive cluster.

Because of an automotive cluster has a lot of complex software and hardware elements, it would take too much time to create a complete one. For this reason, on this thesis only the graphic user interface will be created. It will be prepared to be used or adapted with an automotive communication protocol (CAN). As a consequence, all the project requirements will be satisfied focusing only on the GUI and how it will interact with the CAN automotive communication protocol.

1.5 Background

In the actual automotive industry, the embedded systems are an essential part of the modules that can be found in a car. One of the most important things is how these modules interact with the user along different interfaces or applications. The communication not only relays on handling the information, but also in how this information is presented and used to take actions. These actions may be related to status or security issues to be addressed.

In the car, the first window to the status of the signals is known as the instrument cluster. In here, most of the information of the system is presented in an understandable and practical way to the driver. Information may include status of the speed, mileage, temperature, battery, fuel, light conditions, among others. The information may vary through different Original Equipment Manufacturers (OEM), but the essence remains the same.

Based on these assessments, this thesis addresses the problematic to create a GUI for an automotive cluster and how the information is presented within it. It follows the baseline established into the embedded systems specialization early project, where the first version of an automotive cluster was created.

1.6 Methodology

The methodology used for the project development was the “V-Model”. This model represents a software development process that enables to assess the code against specific testing sections and thus developing a more complete and well-rounded software project. Also, this

model is well used by the car manufacturers and suppliers to ensure the quality of the software used on cars. For a better understanding of the V-Model (by using ASPICE nomenclature) used for this project, please refer to [1].

Although this project development will be based on this model, not all the processes will befulfilled. Only the *ENG.2, ENG.3, ENG.4, ENG.5, ENG.6* and *ENG.10* processes will be developed on this project. This decision was opted to be in compliance with the delivery time. However, this decision will not affect the functionality on the final result. Note that in the automotive industry, this kind of project takes around two years to be finished.

2 Hypothesis

The *Figure 1* describes a proposal for the implementation of the Cluster GUI development. This implementation describes all the elements need it to be considered to solve the aforementioned problematic.

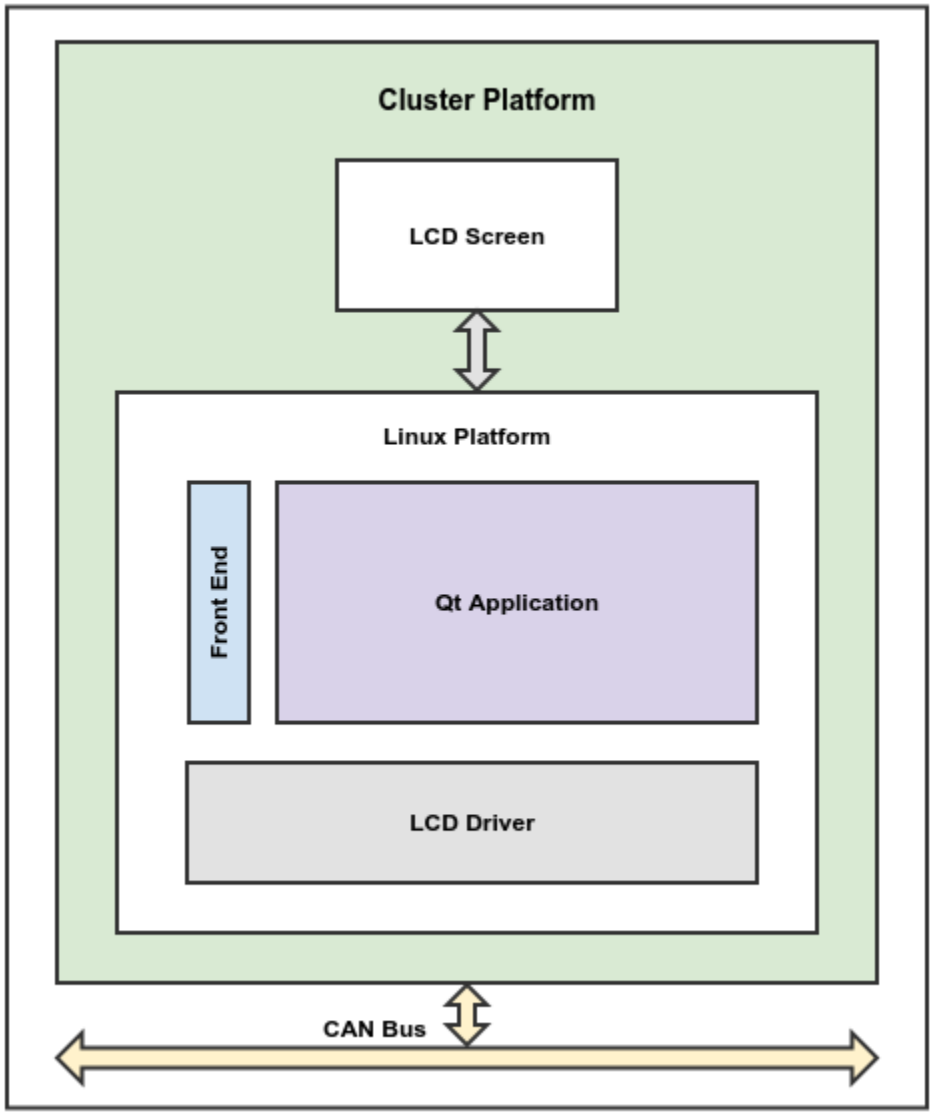


Figure 1: Proposed Cluster Implementation

Although in *Figure 1* shows the CAN bus element, this project only will focus on the GUI development. The intention for showing the CAN bus is to demonstrate this implementation will be able to communicate with automotive protocols (LIN, CAN) and how could be the best way to implement this interface.

The following section will describe the chosen solution for each element of the proposed implementation (cluster platform).

2.1 Hardware

2.1.1 PC Platform

The Raspberry Pi Model 1 B+ was chosen to be the microcomputer where the Linux platform will run. It was chosen because there is a lot of information about it and a big open source community which is continuously working on better features. Due to its hardware characteristics (SPI, I2C, RAM, etc.), flexibility, scalability and low cost is the best option to implement the proposed cluster platform.

2.1.2 Monitor Display

The monitor where the Cluster GUI will be shown is going be a 10”color LCD screen. This LCD screen will allow to display any kind of information and according to the performed research, the best option for an instrument cluster is a LCD screen with a minimum size of 6” that normally does not have touch capability.

2.2 Software

2.2.1 Operative System

An Embedded Linux system will be the operative system to be installed on the RPi. This is the OS in which the GUI (Qt application), LCD driver (driver to communicate with LCD screen) and the Frontend will be running.

The Embedded Linux system was chosen because provides the capabilities and features for sophisticated devices (RPi) than other embedded development platforms. Also, the open source nature can take advantage of the continuous development that it is happening in the open source environment.

Finally, Linux is a well-known OS that can be found on a big variety of devices because it can be modified and adapted to a specific platform characteristics and limited hardware resources. This is the most important feature that allows to use Linux in this project.

2.2.2 GUI Development Tool

The open source Qt Framework is the option to develop the GUI (Qt application) in this Linux platform. This framework takes the advantage of C++ and QML languages to develop impressive GUIs. It is known that most of the graphics interfaces on an Embedded Linux systems use Qt implementation due its graphic libraries portability.

The principal goal for the Linux platform is to have a lightweight windows system, and the Qt Framework meets this approach, because the Qt applications on an Embedded Linux system write directly to the framebuffer, eliminating the need for the X Window System and saving memory resources (principal embedded system characteristic constraint).

2.2.3 LCD Driver

Since the LCD screen is an external device, it is sure that the graphical interface will take care of the information to be displayed on it. For this reason, it is necessary a driver that handles the display and the inter module communication. Fortunately, the Embedded Linux system already contains this driver.

3 System Requirements

For this project it was required to develop a Cluster GUI with the basic premise to display the vital information that the drivers need it at any given time to ensure the safety on the trip the passengers are making.

Based in the actual automotive market, the Cluster GUI **shall** display the follow information:

1. Speedometer
2. Tachometer
3. Fuel Level
4. Battery Voltage
5. Gear Position
6. Odometer
7. Fuel Efficiency
8. Tire Pressure
9. Compass
10. Environment Temperature
11. Clock
12. Key Status
13. Lights
14. Handbrake Status
15. Check Engine Warning
16. ABS Brakes Warning
17. Airbag Status
18. Oil Warning
19. Motor Temperature Warning
20. Seatbelt Status
21. Door Warning
22. Satellite Notification

According to the previous information, the full list of the system requirements will be described in the following sections.

3.1 Speedometer

- (SYSREQ101) The Cluster GUI **shall** always display SPEED value on the screen.
- (SYSREQ102) The SPEED value **shall** be defined in Km/h.
- (SYSREQ103) The SPEED value **shall** be in range of 0 – 190 Km/h.
- (SYSREQ104) The SPEED value resolution **shall** be of decimal value (0.5 Km/h).
- (SYSREQ105) The Cluster GUI **shall** always display SPEED value on the screen.
- (SYSREQ106) The SPEED value **shall** be displayed using a circular gauge.
- (SYSREQ107) The SPEED value **shall** be displayed as an analog measurement.

3.2 Tachometer

- (SYSREQ111) The Cluster GUI **shall** always display a tachometer value on the screen.
- (SYSREQ112) The tachometer value **shall** be defined in RPMs.
- (SYSREQ113) The RPMs value **shall** be in range of 0 - 10000.
- (SYSREQ114) The RPMs value resolution **shall** be of 50 RPMs.
- (SYSREQ115) The RPM unit **shall** always be displayed on the screen (GUI).
- (SYSREQ116) The RPMs value **shall** be displayed using a circular gauge.
- (SYSREQ117) The RPMs value **shall** be displayed as an analog measurement.

3.3 Fuel Level

- (SYSREQ121) The Cluster GUI **shall** display the FUEL LEVEL on the screen.
- (SYSREQ122) FUEL LEVEL **shall** be defined in Liters.
- (SYSREQ123) FUEL value **shall** be in range of 0 to 63.5 Liters.
- (SYSREQ124) FUEL level **shall** be displayed on a circular gauge.
- (SYSREQ125) The FUEL LEVEL value resolution **shall** be of 0.5 Liters.
- (SYSREQ126) FUEL LEVEL measurement **shall** display only the percent of the contained fuel.
- (SYSREQ127) The circular gauge **shall** indicate when the FUEL value is in reserve (1/8).
- (SYSREQ128) When the FUEL value is in reserve, a warning fuel indicator **shall** be displayed.
- (SYSREQ129) The color for the warning fuel indicator **shall** be yellow.
- (SYSREQ130) The FUEL LEVEL **shall** be displayed as an analog measurement.

3.4 Battery Voltage

- (SYSREQ131) The Cluster GUI **shall** display the BATTERY VOLTAGE on the screen.
- (SYSREQ132) BATTERY VOLTAGE **shall** be displayed on a circular gauge.
- (SYSREQ133) When the BATTERY VOLTAGE is low, a warning battery indicator **shall** be displayed.
- (SYSREQ134) The color for the warning battery indicator **shall** be yellow.
- (SYSREQ135) BATTERY VOLTAGE **shall** be in range of 0 to 15.5 Volts.
- (SYSREQ136) The VOLTAGE value resolution **shall** be of 0.5 Volts.
- (SYSREQ137) The BATTERY VOLTAGE **shall** be displayed as an analog measurement.

3.5 Gear Position

- (SYSREQ141) The Cluster GUI **shall** display the GEAR POSITION on the screen.
- (SYSREQ142) TheGEAR POSITIONS to display **shall** be PARKING, NEUTRAL, REVERSE, DRIVE, 1 and 2.

3.6 Odometer

- (SYSREQ151) The Cluster GUI **shall** display the ODOMETER value on the screen.
- (SYSREQ152) Distance ODOMETER value **shall** be defined in Kilometers.

- (SYSREQ153) Distance ODOMETER value **shall** be in range of 0 - 999999 Km.
- (SYSREQ154) Distance ODOMETER resolution **shall** be of 1 Km.

3.7 Fuel Efficiency

- (SYSREQ161) The Cluster GUI **shall** display the FUEL EFFICIENCY value on the screen.
- (SYSREQ162) FUEL EFFICIENCY**shall** be defined in Kilometers per Liter Km/L.
- (SYSREQ163) FUEL EFFICIENCY value **shall** be in range of 0 – 31.75 Km/L.
- (SYSREQ164) FUEL EFFICIENCY resolution **shall** be of 0.25 Km/L.

3.8 Tire Pressure

- (SYSREQ171) The Cluster GUI **shall** display the PRESSURE value for each TIRE.
- (SYSREQ172) TIRE PRESSURE **shall** be defined in PSI.
- (SYSREQ173) PRESSURE value **shall** be in range of 0 – 63 PSI.
- (SYSREQ174) PRESSURE resolution **shall** be of 1 PSI.
- (SYSREQ175) If a tire has low pressure, a PRESSURE WARNING indicator **shall** be displayed.
- (SYSREQ176) The Cluster GUI **shall** have 4 PRESSURE WARNING indicators, one for each tire.
- (SYSREQ177) The color for the PRESSURE WARNING indicator **shall** be yellow.

3.9 Compass

- (SYSREQ181) The Cluster GUI **shall** display a COMPASS direction on the screen.
- (SYSREQ182) The COMPASS **shall** display the following directions: NORTH, EAST, WEST, SOUTH, NORTHEAST, NORTHWEST, SOUTHEAST and SOUTHWEST.

3.10 Environment Temperature

- (SYSREQ191) The Cluster GUI **shall** display ENVIROMENT TEMPERATURE on the screen.
- (SYSREQ192) TEMPERATURE value **shall** be defined in ^{o}C.
- (SYSREQ193)TEMPERATURE value **shall** be in range of -63.5 to 63.5 ^{o}C.
- (SYSREQ194) TEMPERATURE resolution **shall** be of 0.5 ^{o}C.

3.11 Clock

- (SYSREQ201) The Cluster GUI **shall** display CLOCK on the screen.
- (SYSREQ202) The CLOCK **shall** display the time of the current time zone.
- (SYSREQ203) CLOCK resolution **shall** be of 1 Sec.

3.12 Key Status

- (SYSREQ211) The Cluster GUI **shall** display the KEY STATUS indicator (car ignition switch) on the screen.
- (SYSREQ212) The KEY STATUS **shall** have two values, key status ON and key status OFF.

- (SYSREQ213) If KEY STATUS OFF, the Cluster GUI **shall** be capable of turning OFF the display, while still functioning in listen mode.
- (SYSREQ214) If KEY STATUS ON, the Cluster GUI **shall** be capable of turning ON the display.
- (SYSREQ215) On KEY transition from ON to OFF, the GUI **shall** display an animation using all the indicators.
- (SYSREQ216) The KEY STATUS indicator **shall** be yellow.
- (SYSREQ217) The KEY STATUS indicator **shall** be shown only when the Key status is ON.

3.13 Lights

- (SYSREQ221)The Cluster GUI **shall** display TURN LEFT and TURN RIGHT directional indicators on the screen.
- (SYSREQ222) The color for the TURN LEFT and TURN RIGHT directional indicators **shall** be green.
- (SYSREQ223)The Cluster GUI **shall** display HIGH BEAM and LOW BEAM light indicators on the screen.
- (SYSREQ224) The color for the HIGH BEAM light indicator **shall** be blue.
- (SYSREQ225) The color for the LOW BEAM light indicator **shall** be yellow.
- (SYSREQ226) The Cluster GUI **shall** display a HAZARD WARNING light indicator on the screen.
- (SYSREQ227) The color for the HAZARD WARNING light indicator **shall** be red.

3.14 Handbrake Status

- (SYSREQ231) The Cluster GUI **shall** display a HANBRAKE STATUS indicator on the screen.
- (SYSREQ232) The color for the HANDBRAKE STATUS indicator **shall** be red.

3.15 Check Engine Warning

- (SYSREQ241) The Cluster GUI **shall** display a CHECK ENGINE WARNING indicator on the screen.
- (SYSREQ242) The color for the CHECK ENGINE WARNING indicator **shall** be red.

3.16 ABS Brakes Warning

- (SYSREQ251) The Cluster GUI **shall** display an ABS WARNING indicator on the screen.
- (SYSREQ252) The color for the ABS WARNING indicator **shall** be red.

3.17 Airbag Status

- (SYSREQ261) The Cluster GUI **shall** display an AIRBAG STATUS indicator on the screen.
- (SYSREQ262) The color for the AIRBAG STATUS indicator **shall** be red.

3.18 Oil Warning

- (SYSREQ271) The Cluster GUI **shall** display an OIL WARNING indicator on the screen.
- (SYSREQ272) The color for the OIL WARNING indicator **shall** be red.

3.19 Motor Temperature Warning

- (SYSREQ281) The Cluster GUI **shall** display a MOTOR TEMPERATURE WARNING indicator on the screen.
- (SYSREQ282) The color for the MOTOR TEMPERATURE WARNING indicator **shall** be red.

3.20 Seatbelt Status

- (SYSREQ291) The Cluster GUI **shall** display a SEATBELT STATUS indicator on the screen.
- (SYSREQ292) The color for the SEATBELT STATUS indicator **shall** be red.

3.21 Door Warning

- (SYSREQ301) The Cluster GUI **shall** display a DOOR WARNING indicator on the screen.
- (SYSREQ302) The color for the DOOR WARNING indicator **shall** be red.

3.22 Satellite Notification

- (SYSREQ311) The Cluster GUI **shall** display a SATELLITE NOTIFICATION indicator on the screen.
- (SYSREQ312) The color for the SATELLITE NOTIFICATION indicator **shall** be yellow.

3.23 Requirements for Status and Warning Indicators

These are the requirements for all the Status and Warning indicators described on the above sections:

- (SYSREQ321) All the STATUS and WARNING indicators **shall** have two values: ON and OFF.
- (SYSREQ322) If value ON, the (STATUS and WARNING) indicator **shall** be displayed on the screen.
- (SYSREQ332) If value OFF, the (STATUS and WARNING) indicator **shall** not be displayed on the screen.

3.24 General Requirements

- (SYSREQ331) The Cluster GUI **shall** be able to receive signals/information from any automotive protocol driver (CAN, LIN, etc) implemented on this Embedded Linux platform.
- (SYSREQ332) The Cluster GUI information **shall** be updated every 100ms.
- (SYSREQ333) The Cluster GUI design **shall** follow the template shown on *Figure 2.*

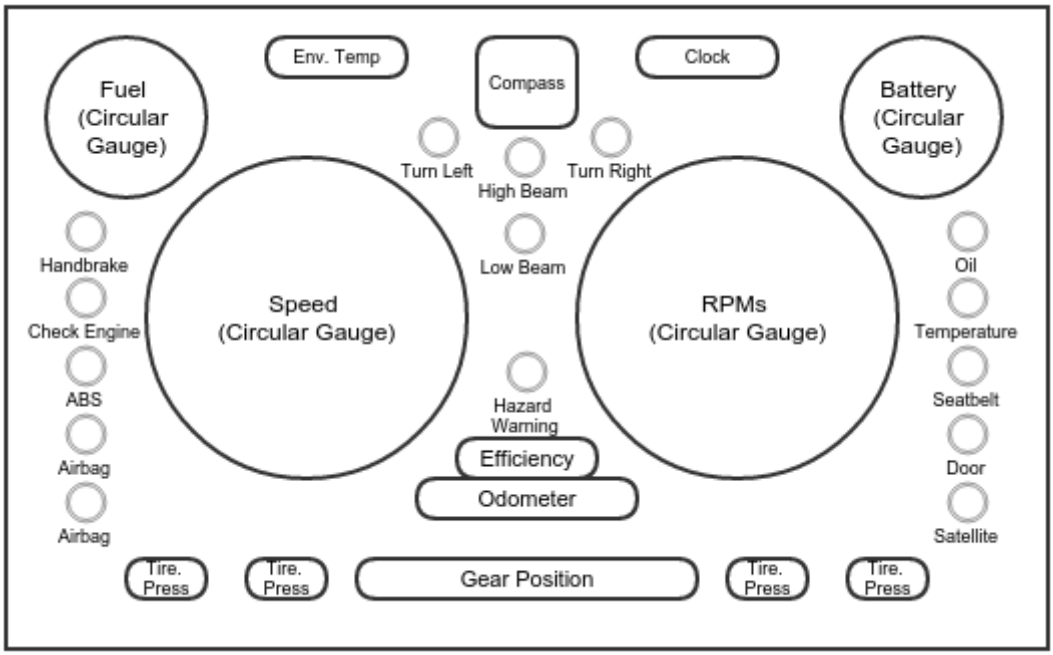


Figure 2: Cluster GUI Template

4 System Architecture

As can be seen in *Figure 3,* the system architecture for the Cluster GUI is very simple. Only the Raspberry Pi will be send the video signal (HDMI) to the LCD screen and those devices will be powered by a DC source. Once the video signal is sent to the LCD screen the Cluster GUI is going to be shown.

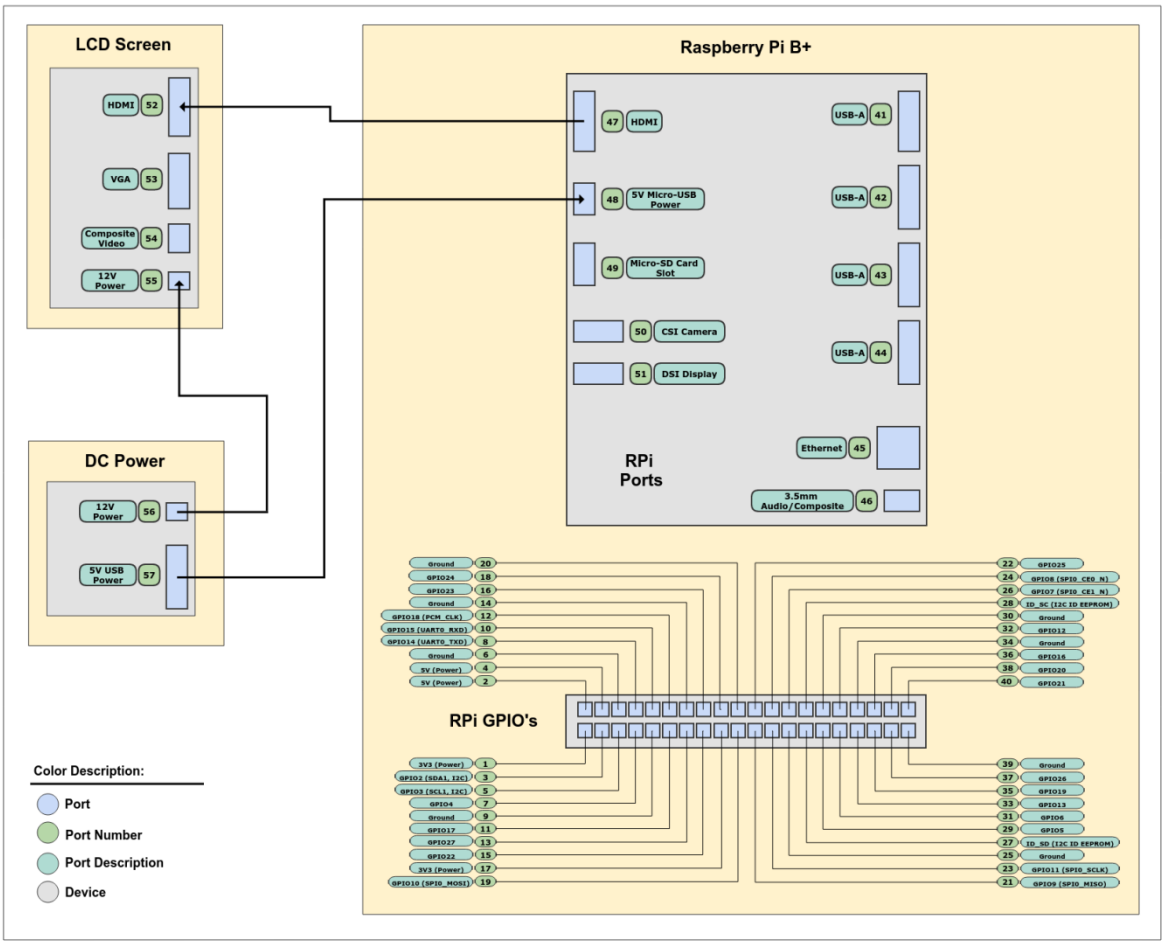


Figure 3: Cluster System Architecture

Note that *Figure 3* does not show the sub-system elements of the Raspberry Pi due to the fact that was designed as a standalone PC platform working with a Linux based OS.

The *Table 1* shows a relation between the hardware ports and software signals to be used on the cluster GUI platform. Only two ports/signals are going to be used, the HDMI video input/output and the power signal (RPi/LCD screen).

Port/Pin Number	Hardware Port/PinName (RPi)	Software Function
1	3V3 (Power)	NOT_USED
2	5V (Power)	NOT_USED
3	GPIO2 (SDA, I2C)	NOT_USED
4	5V (Power)	NOT_USED
5	GPIO3 (SCL1, I2C)	NOT_USED
6	Ground	NOT_USED
7	GPIO4	NOT_USED
8	GPIO14 (UART0_TXD)	NOT_USED
9	Ground	NOT_USED
10	GPIO15 (UART0_RXD)	NOT_USED
11	GPIO17	NOT_USED
12	GPIO18 (PCM_CLK)	NOT_USED
13	GPIO27	NOT_USED
14	Ground	NOT_USED
15	GPIO22	NOT_USED
16	GPIO23	NOT_USED
17	3V3 (Power)	NOT_USED
18	GPIO24	NOT_USED
19	GPIO10 (SPI0_MOSI)	NOT_USED
20	Ground	NOT_USED
21	GPIO9 (SPI0_MISO)	NOT_USED
22	GPIO25	NOT_USED
23	GPIO11 (SPI0_SCLK)	NOT_USED
24	GPIO8 (SPI0_CE0_N)	NOT_USED
25	Ground	NOT_USED
26	GPIO7 (SPI0_CE1_N)	NOT_USED
27	ID_SD (I2C ID EEPROM)	NOT_USED
28	ID_SC (I2C ID EEPROM)	NOT_USED
29	GPIO5	NOT_USED
30	Ground	NOT_USED
31	GPIO6	NOT_USED
32	GPIO12	NOT_USED
33	GPIO13	NOT_USED
34	Ground	NOT_USED
35	GPIO19	NOT_USED
36	GPIO16	NOT_USED
37	GPIO26	NOT_USED
38	GPIO20	NOT_USED
39	Ground	NOT_USED
40	GPIO21	NOT_USED
41	USB-A	NOT_USED
42	USB-A	NOT_USED
43	USB-A	NOT_USED
44	USB-A	NOT_USED
45	Ethernet Port	NOT_USED
46	3.5mm Audio/Composite Jack	NOT_USED
47	HDMI Output	RPi_Video_Output
48	5V Micro-USB Power	RPi_Power
49	Micro-SD Card Slot	OS_Memory_SD_Card
50	CSI Camera Connector	NOT_USED
51	DSI Display Connector	NOT_USED
52	HDMI Input	LCD_Video_Input
53	VGA Input	NOT_USED
54	Composite Video Input	NOT_USED
55	12V/GND Power	LCD_Power
56	12V/GND Power	LCD_Power
57	5V USB Power	RPi_Power

Table 1: Cluster System HW/SW Interface Signals

5 Software Requirements

Based on the systems requirements described on *Chapter 4,* in this section the minimum software requirements for the Cluster GUI implementation will be defined.

5.1 OS

- (SOFREQ101) The Operative System for the Cluster GUI **shall** be an Embedded Linux system.
- (SOFREQ102) The Embedded Linux system **shall** be Raspbian version Nov 2013, Kernel 3.6.11.
- (SOFREQ103) The Boot time **shall** be at least of 10 Seconds.
- (SOFREQ104) The Raspbian image **shall** be optimized by removing all the unnecessary packages for the GUI development.
- (SOFREQ105) Upstart **shall** be used for the system init process.
- (SOFREQ106) Upstart **shall** be the only process started by the Linux kernel.
- (SOFREQ107) Upstart **shall** start all process used on the optimized image.
- (SOFREQ108) The Embedded Linux system **shall** be able to have 100Base-T Ethernet connections.
- (SOFREQ109) The Embedded Linux system **shall** be able to use SSH connections.
- (SOFREQ110) The optimized Raspbian image **shall** be able to use shared memory.
- (SOFREQ111) The optimized Raspbian image **shall** be able to use semaphores.

5.2 GUI

- (SOFREQ121) The GUI **shall** be developed using the Qt graphics libraries – open source Qt Framework.
- (SOFREQ122) The Qt graphic libraries version **shall** be 5.1.1.
- (SOFREQ123) The Qt graphic libraries **shall** be included on the optimized Raspbian image.
- (SOFREQ124) The Qt graphic libraries **shall** be dynamically called.
- (SOFREQ125) Since Qt Framework has C++ and QML modules, the GUI **shall** be implemented using C++ for processing signals and QML for showing the received signals.
- (SOFREQ126) The Qt application **shall** access to shared memory in order to update the indicators and measurement values of the GUI.
- (SOFREQ127) The Qt application **shall** cast the shared memory section into a structure where all the signals are going to be stored.
- (SOFREQ128) The structure **shall** be named `tshared_memory`.
- (SOFREQ129) The `tshared_memory` structure size **shall** be of 44 Bytes.
- (SOFREQ130) The Qt application **shall** be able to use semaphores when access to the shared memory in order to protect the data integrity.

- (SOFREQ131) The Qt application **shall** display the GUI background and circular gauges using PNG images.
- (SOFREQ132) The GUI screen size **shall**be of 1360x768 pixels.
- (SOFREQ133) The GUI background **shall** have the circular gauges for SPEED, RPMs, BATTERY and FUEL (without needles).
- (SOFREQ134) The platform plugin to be used on the GUI implementation **shall** be EGLFS.
- (SOFREQ135) The Cluster GUI **shall** be developed on the user space.

5.2.1 Speedometer

- (SOFREQ141) The SPEED signal value **shall** be stored on a `float` data type.
- (SOFREQ142) The variable name for the SPEED signal value **shall** be `speed`.
- (SOFREQ143) The SPEED value **shall** be displayed on the circular gauge by rotating a needle.

5.2.2 Tachometer

- (SOFREQ161) The RPMs signal value **shall** be stored on an`uint16_t` data type.
- (SOFREQ162) The variable name for the RPMs signal value **shall** be `rpms`.
- (SOFREQ163) The RPMs value **shall** be displayed on the circular gauge by rotating a needle.

5.2.3 Fuel Level

- (SOFREQ181) The FUEL LEVEL signal **shall** be stored on a `float` data type.
- (SOFREQ182) The variable name for the FUEL LEVEL signal value **shall** be `fuel`.
- (SOFREQ183) The FUEL value **shall** be displayed on the circular gauge by rotating a needle.
- (SOFREQ185) The variable name for the LOW FUEL warning indicator signal **shall** be `low_fuel`.

5.2.4 Battery Voltage

- (SOFREQ201) The BATTERYVOLTAGE signal **shall** be stored on a `float` data type.
- (SOFREQ202) The variable name for the BATTERY signal **shall** be `battery`.
- (SOFREQ203) The BATTERY VOLTAGE value **shall** be displayed on the circular gauge by rotating a needle.
- (SOFREQ204) The variable name for the LOW BATTERYWARNING indicator signal **shall** be `low_battery`.

5.2.5 Gear Position

- (SOFREQ221) The GEAR POSITION signal **shall** be stored on an `uint8_t` data type.
- (SOFREQ222) The variable name for the GEAR POSITION signal **shall** be `gear`.
- (SOFREQ223) The valid values for the GEAR POSITION signal **shall** be 0=P, 1=N, 2=R, 3=D, 4=1 y 5=2.

5.2.6 Odometer

- (SOFREQ241) The ODOMETER signal **shall** be stored on an `uint32_t` data type.
- (SOFREQ242) The variable name for the ODOMETER signal **shall** be `odometer`.
- (SOFREQ243) The font size for the ODOMETER value **shall** be 20.
- (SOFREQ244) The "KM" unit **shall** be displayed after the ODOMETER value.

5.2.7 Fuel Efficiency

- (SOFREQ261) The FUEL EFFICINECY signal **shall** be stored on a `float` data type.
- (SOFREQ262) The variable name for the FUEL EFFICIENCY signal **shall** be `efficiency`.
- (SOFREQ263) The font size for the FUEL EFFICIENCY value **shall** be 20.
- (SOFREQ264) The "KM/L" unit **shall** be displayed after the Fuel Efficiency value.

5.2.8 Tire Pressure

- (SOFREQ281) The PRESSURE signal of each tire **shall** be stored on an `uint8_t` data type.
- (SOFREQ282) The variable name for each TIRE PRESSURE signal **shall** be `tire_pressure_X`. Where X is the tire number (1-4).
- (SOFREQ283) The font size for the TIRE PRESSURE value **shall** be 18.
- (SOFREQ284) The "PSI" unit **shall** be displayed after the TIRE PRESSURE value.
- (SOFREQ285) The variable name for the LOW TIRE PRESSURE warning indicator signal **shall** be `low_pressure_X_warning`. Where X is the tire number (1-4).

5.2.9 Compass

- (SOFREQ301) The COMPASS signal **shall** be stored on an `uint8_t` data type.
- (SOFREQ302) The variable name for the COMPASS signal **shall** be `compass`.
- (SOFREQ303) The valid values for the COMPASS signal **shall** be 0=N, 1=NE, 2=E, 3=SE, 4=S, 5=SW, 6=W and 7=NW.
- (SOFREQ304) The font size for the COMPASS direction value **shall** be 68.

5.2.10 Environment Temperature

- (SOFREQ321) The ENVIROMENT TEMPERATURE signal **shall** be stored on a `float` data type.
- (SOFREQ322) The variable name for the ENVIROMENT TEMPERATURE signal **shall** be `env_temp`.
- (SOFREQ323) The font size for the TEMPERATURE value **shall** be 20.
- (SOFREQ324) The "^{o}C" unit **shall** be displayed after the TEMPERATURE value.

5.2.11 Clock

- (SOFREQ341) The Cluster GUI **shall** use anddisplay the system clock.
- (SOFREQ342) The font size for the CLOCK **shall** be 20.
- (SOFREQ343) The "PM" or "AM" **shall** be displayed after the CLOCK.

- (SOFREQ344) The time format **shall** be HH:MM:SS.

5.2.12 Key Status

- (SOFREQ361) The variable name for the KEY STATUS indicator signal **shall** be `key_status`.
- (SOFREQ362) If KEY STATUS0 = OFF, the entire Cluster GUI screen **shall** be black.
- (SOFREQ363) If KEY STATUS 0 = OFF, the `shutdown` variable value **shall** be 1.
- (SOFREQ364) If KEY STATUS 1 = ON, the `shutdown` variable value **shall** be 0.
- (SOFREQ365) The `shutdown` variable **shall** control when the LCD screen should show a black screen (emulate an off screen).
- (SOFREQ366) The `shutdown` value **shall** be stored on an `uint8_t` data type.
- (SOFREQ367) The `shutdown` variable **shall** not be part of the `tshared_memory` structure.

5.2.13 Lights

- (SOFREQ381) The variable name for the TURN LEFT indicator signal **shall** be `turn_left`.
- (SOFREQ382) The variable name for the TURN RIGHT indicator signal **shall** be `turn_right`.
- (SOFREQ383) The variable name for the HIGH BEAM indicator signal **shall** be `high_beam`.
- (SOFREQ384) The variable name for the LOW BEAM indicator signal **shall** be `low_beam`.
- (SOFREQ385) The variable name for the HAZARD WARNING indicator signal **shall** be `hazard_warning_light`.

5.2.14 Handbrake Status

- (SOFREQ401) The variable name for the HAND BRAKE STATUS indicator signal **shall** be `hand_brake`.

5.2.15 Check Engine Warning

- (SOFREQ421) The variable name for the CHECK ENGINE WARNING indicator signal **shall** be `check_engine`.

5.2.16 ABS Brakes Warning

- (SOFREQ441) The variable name for the ABS BRAKES WARNING indicator signal **shall** be `abs_break`.

5.2.17 Airbag Status

- (SOFREQ461) The variable name for the AIRBAG STATUS indicator signal **shall** be `airbag`.

5.2.18 Oil Warning

- (SOFREQ481) The variable name for the OIL WARNING indicator signal **shall** be `oil`.

5.2.19 Motor Temperature Warning

- (SOFREQ501) The variable name for the MOTOR TEMPERATURE WARNING indicator signal **shall** be `motor_temperature_warning`.

5.2.20 Seatbelt Status

- (SOFREQ521) The variable name for the SEATBELT STATUS indicator signal **shall** be `seat_belt`.

5.2.21 Door Warning

- (SOFREQ541) The variable name for the DOOR WARNING indicator signal **shall** be `door_warning_light`.

5.2.22 Satellite Notification

- (SOFREQ561) The variable name for the SATELLITE NOTIFICATION indicator signal **shall** be `satellital_notification`.

5.2.23 Requirements for Status and Warning Indicators

- (SOFREQ581) All the signals for the STATUS and WARNING indicators **shall** be stored on `uint8_t` data type.
- (SOFRE582) All STATUS and WARNING indicators **shall** be displayed using a PNG image.
- (SOFREQ583) The image size for the STATUS and WARNING indicators **shall** be between 60x50 and 70x70 pixels.
- (SOFREQ584) The indicators size **shall** be adjusted by modifying the SCALE option on the QML module.
- (SOFREQ585) The image background for the STATUS and WARNING indicators **shall** be transparent.
- (SOFREQ586) The values for the STATUS and WARNING indicators **shall** be 1 = ON and 0 = OFF.
- (SOFREQ587) If value 1, the (STATUS and WARNING) indicator **shall** be displayed on the screen.
- (SOFREQ588) If value 0, the (STATUS and WARNING) indicator **shall** not be displayed on the screen.

5.2.24 General Requirements

- (SOFREQ611) In the C++ and QML modules, the variable name for the every signal value **shall** be the same.
- (SOFREQ612) Every signal variable **shall** be declared within the `tshared_memory` structure.
- (SOFREQ613) All the signals values that need to be displayed on the screen **shall** be use a digital font type. This does not apply for STATUS and WARNING indicators.
- (SOFREQ614) The font color for all the signals values that need to be displayed on the screen **shall** be white. This does not apply for STATUS and WARNING indicators.

6 Software Architecture

The *Figure 4* shows the software architecture for the Cluster GUI. In this architecture, only the components and modules that need to be developed are described.

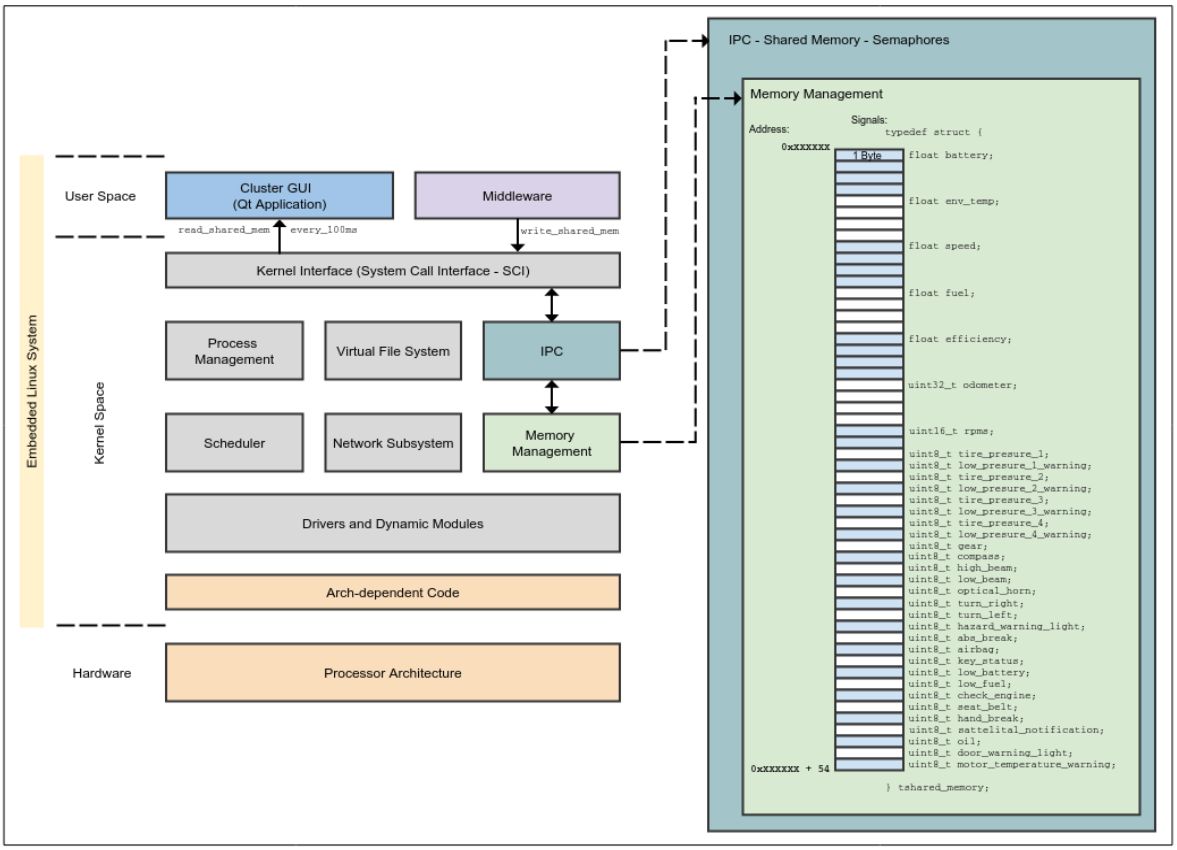


Figure 4: Cluster GUI Software Architecture

According to *Figure 4*, only the communication between the GUI and the middleware through an IPC is described. In this case, some kernel space modules (IPC, Memory Management) are used to establish this communication between these two processes.

The software architecture shown on *Figure 4* can be described as follows:

1. First, the middleware shall create a shared memory section trough the IPC module on the kernel space. This shared memory will be used to communicate the middleware and the Cluster GUI.
2. Then, the Cluster GUI shall be subscribed to the shared memory section trough the same IPC module.
3. Once the shared memory (IPC) was created, the middleware shall cast it to the `tshared_memory` structure.
4. Then the middleware shall write the signal values to the `tshared_memory`.

5. After the signal values were updated, the Cluster GUI shall cast the shared memory section to the `tshared_memory` structure.
6. Finally, the Cluster GUI shall read the signal values every 100ms in order to update their status on the screen.

Note that every time the processes read or write into the shared memory, a semaphore shall be implemented to avoid data corruption. A simpler diagram of this Software Architecture is shown on *Figure 5.*

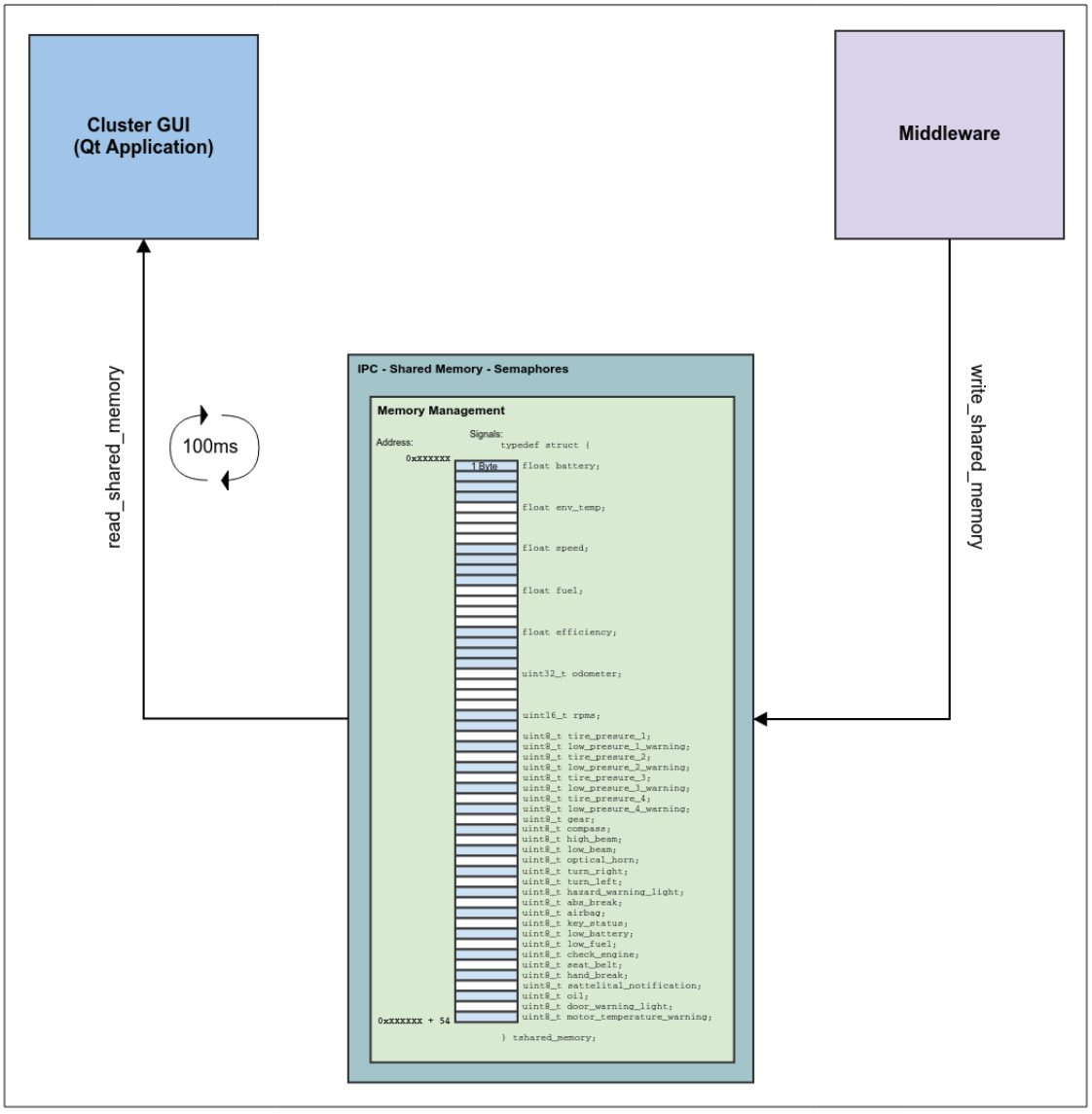


Figure 5: Communication between Cluster GUI and Middleware

7 Design and Implementation

7.1 Linux Image Optimization

In order to optimize the Raspbian Linux distribution, the Buildroot tool was chosen. This tool is widely used for Linux Embedded systems projects and it is well known that is very easy and simple to use.

7.1.1 What is Buildroot?

"Buildroot is a tool that simplifies and automates the process of building a complete Linux system for an embedded system, using cross-compilation". [2] It makes this possible by generating a cross-compilation toolchain, a root file system, a Linux kernel image and a bootloader for the target.

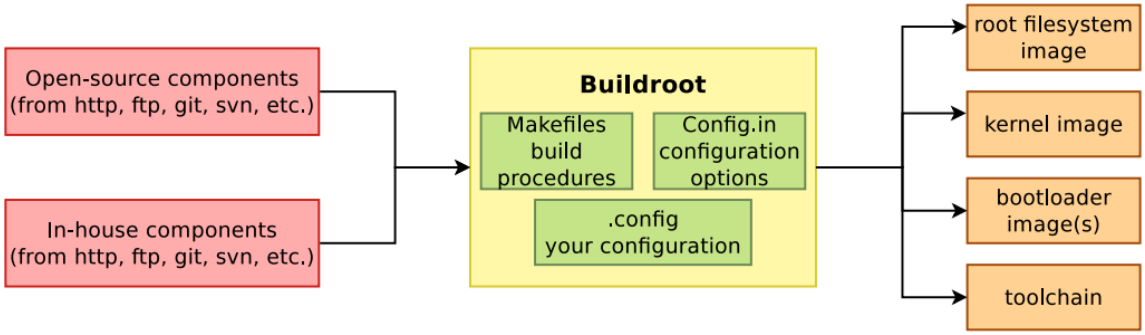


Figure 6: How Buildroot works

On more simple words, as can be seen on *Figure 6,* Buildroot performs three main tasks, in the first one Buildroot downloads all the required components from the internet or from in-house sources, on the second it applies all the set configurations and builds all the packages, and in the last one it generates all the necessary components as the root filesystem, kernel, bootloader and the toolchian for the specific target.

Finally, two important concepts that need to be noted in order to understand the Buildroot functionality are:

- ***Cross-Compilation:*** It refers to compiling code for one computer system (target) on a different system (host).
- ***Toolchain:*** It is the set of tools that allows to compile code for the target system. It consists of a compiler, binary utils like assembler and linker and a C standard library. [2]

Althought Buidlroot was one of the most important tools used on this project; this document will not describe how Buildroot works in deep, this document will limit to show and describe only the final Buildroot configuration files (*recipe, .mk, Config.in, etc.*.) used to generate

the customized and optimized Linux image (Raspbian). Please refer to [2] for a complete Buildroot user manual.

7.1.2 Buildroot on Cluster GUI project

It is noted that in this document will not describe how every package used in the final Linux Image was added or chosen on the Buildroot recipe and what OS configurations were performed.

Due it is impossible to explain and describe every package and configuration that was used on the final Buildroot recipe, in this section only the main packages and the most important configurations that were implemented will be explained. All these tweaks are shown in *Appendix – section 10.1.*

7.1.2.1 Linux Image Tweaks

For Linux image, these are the most important tweaks that were implemented on the final Buildroot recipe:

- Rather than using `sysinitv` or `systemd` for the `system init` process (which is the only process started by the Linux kernel and is the parent of all other processes), it was chosen to use `Upstart`. `Upstart` is event driven, so services are started in terms of what events occur. By converting packages `sysinitv` style service files to `upstart` style, it was possible to reduce boot times to a console to just 7 seconds. It is also much easier to write `upstart` services than it is to write new `sysinitv` ones.
- The stable Raspberry Pi kernel 3.6.11 version was used.
- Busybox was included.
- GDB was installed on the target. Remote debugging enabled.
- QT 5.1.1 graphics libraries were installed on the target.
- GCC 4.6.3 toolchain for building armv6 binaries.
- The Eclipse plugin was enabled. Cross-compiling from Eclipse CDT.
- DLT daemon package is included for logging proposes.
- The Cluster GUI project is copied and configured on the post build script.
- Several unused packages were removed from the original Raspian distribution.
    - X server, audio manager, network manager, office suite, etc. All the removed packages can be identified on the configuration files.

The most important Buildroot configuration files used for the optimized Linux image are shown in *Appendix – section 10.1*.

7.1.2.2 Getting and building the Linux Image

Enter the BuildRoot directory and generate a Makefile:

```
$ cd BuildRoot
$ make raspberrypi_defconfig
```

Start the build (this can take a few hours the first time):

```
$ make
```

7.1.2.3 Using Generated Image on the Raspberry Pi

First, it is necessary to obtain a SD card that has the correct partitions setup. It needs to be setup as follows:

- 75MB fat32 partition
- 500MB or greater ext4 partition (ideally using the remainder of the card)

When this setup is completed, the two partitions need to be mounted (assuming `/media/BOOT` for the `fat32` partition, and `/media/rootfs` for the `ext4`). Then, run the following commands to install the `rootfs`:

```
$ cd output/images
$ tar -zxvf boot.tar.gz -C /media/BOOT
$ sudo tar -zxvf rootfs.tar.gz -C /media/rootfs
```

Root user privileges (sudo) must be used when extracting `rootfs.tar.gz`, otherwise several problems will be present on boot.

Finally, place the SD card in the Raspberry Pi and power on. If everything went as planned, a login prompt should be shown. The default login information is:

- username: `root`
- password: `root`

7.2 Qt Graphics Libraries

7.2.1 How Qt was integrated to the Buildroot Linux Image

The primary package to install the Qt libraries is the `QTBASE` module. "This module provides the essential support for embedded and multi-process Linux environments. Therefore its main focus is to facilitate IPC communication, process and document management". [3] The most common use for this module is the UI development. In this project, this was the only Qt module used to develop the Cluster GUI.

The configuration files (`Config.in and qtbase.mk`) used to integrate the `QTBASE` module in the Buidlroot recipe are shown in *Appendix – section 10.2.1.* The process to integrate this module to Buildroot was performed as described in [2].

7.2.2 Build Qt for Raspberry Pi

Due to the final Linux image generated through Buildroot to be installed on a Raspberry Pi computer; several extra configurations need be performed in order to Qt works on this hardware.

When Qt is configured for a specific hardware, there are arguments that need to be considered, these arguments are known as `mkspecs`.

These `mkspecs` are part of the Qt source code, but also after the Linux image is build they can be found on `output/host/usr/mkspecs`.

Inside the mkspecs/ directory, there are several files, but one of the most important file is `qmake.conf`. This file contains all the necessary variables to know how to cross-compile for the target device.

For the Raspberry Pi device, this configuration file is located on: `output/host/usr/mkspecs/devices/linux-rasp-pi-g++/qmake.conf`. But as it was already mentioned, this file is part of the Qt source code.

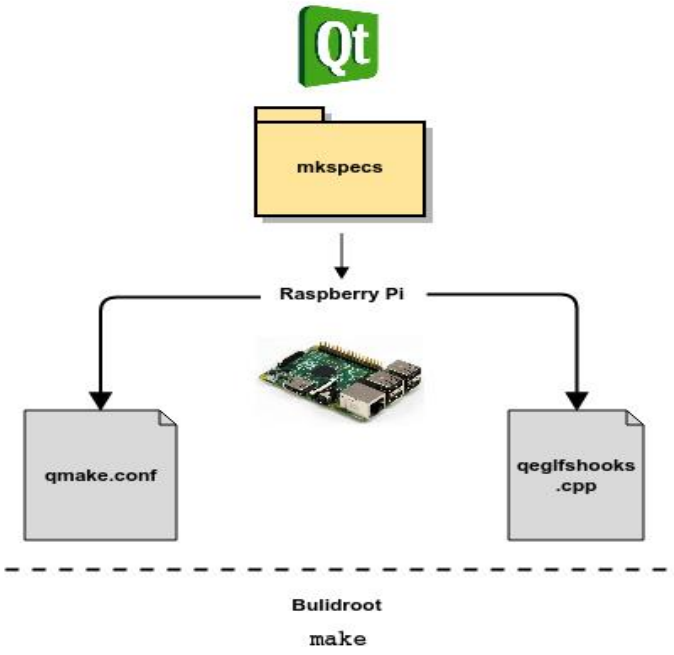


Figure 7: Qt configuration process for RPi

These are the most important configurations that were set on `qmake.conf` to run Qt on a Raspberry Pi:

- `sysroot` location.
- Location of the OPENGL_ES2 and EGL files and how they are linked together at runtime.
- Broadcom specific host library.
- EGLFS hooks: These are additional variables that are needed to initialize EGL and OPENGL libraries. In the case of the Raspberry Pi, this is done by creating an additional class: `qeglfshooks.cpp`.
- `qeglfshooks.cpp`:
  - It is loaded by the EGLFS platform plug-in.
  - Specifies how the hardware is initialized.

- Defines the special hardware capabilities.

An example of the Qt configuration process for the Raspberry Pi is shown on *Figure 7.* Once all the configurations are set, it is the turn for Buildroot to build the Linux image with the Qt libraries included.

The `qmake.conf` and `qeglfshooks.cpp` files are shown in *Appendix – section 10.2.2*.

7.2.3 Qt IDE Configuration for Cross-Compiling

7.2.3.1 Qt Creator

Qt Creator is a cross-platform Integrated Development Environment (IDE) specifically designed for the Qt developers. It contains a visual debugger, GUI layout and forms designer.

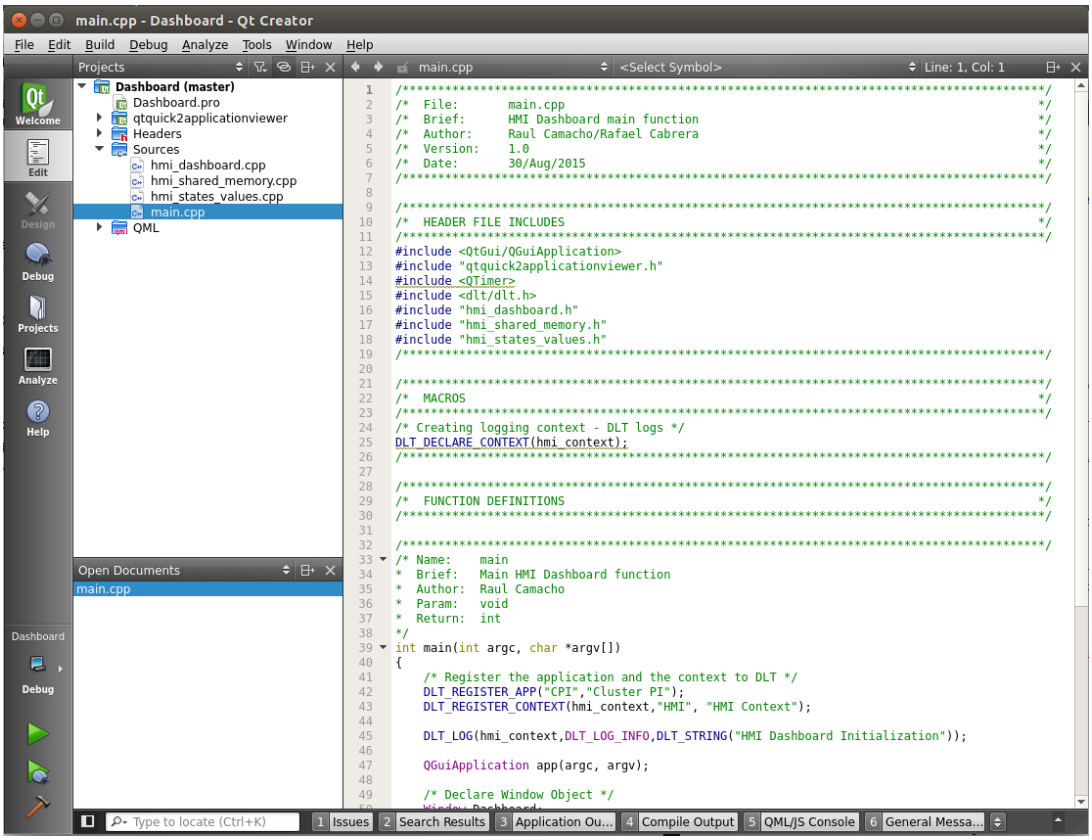


Figure 8: Qt Creator IDE

Qt Creator can be downloaded following the indications exposed in [4].

7.2.3.2 Cross-Compiling Configuration

Once Buildroot finishes building the Linux image, Qt Creator can be configured to perform a cross-compilation using the files on the `output/` directory.

This configuration allows compiling source code from the host machine and copying the generated binaries to the device target (RPi) directly from the Qt Creator. An example of the cross-compiling process used on this project can be seen on *Figure 9.*

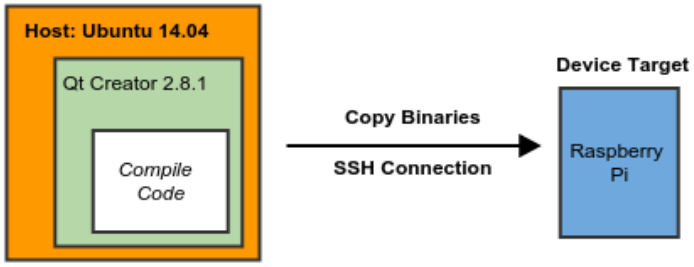


Figure 9: Cross-Compilation process from Qt Creator

The steps to configure Qt Creator for cross-compiling are described in *Appendix – section 10.3*. Finally, it is noted that all the Cluster GUI (HMI) was developed through Qt Creator; no other external tools were used.

7.3 Logging Features

7.3.1 GENIVI Diagnostic Log and Trace - DLT

In the automotive industry field, "DLT is a reusable open source software component for standardized logging and tracing in infotainment ECUs based on the AUTOSAR 4.0 standard". [5] The main purpose of DLT is to unify all the diversity of logging and tracing protocols on one single format.

7.3.2 DLT Elements

DTL is composed of the following elements:

- ***DLT Library:*** Enables DLT logging for DLT user applications and temporary storage of log messages if DLT daemon is not available.
- ***DLT Daemon:*** Transmits the received log messages from DLT user applications to DLT client, also it storages log messages if DLT client is not available and responds to control messages.
- ***DLT Client:*** Receives and storages the log messages from DLT daemon into one single trace file and also is able to sends control messages. The DLT client has a complete Qt visual application known as DLT Viewer which makes easier receiving and analyzing log messages from the DLT daemon. [5] The *Figure 10* shows how the DLT viewer looks.

A complete guide to implement and use these DLT elements for an application can be found in [5].

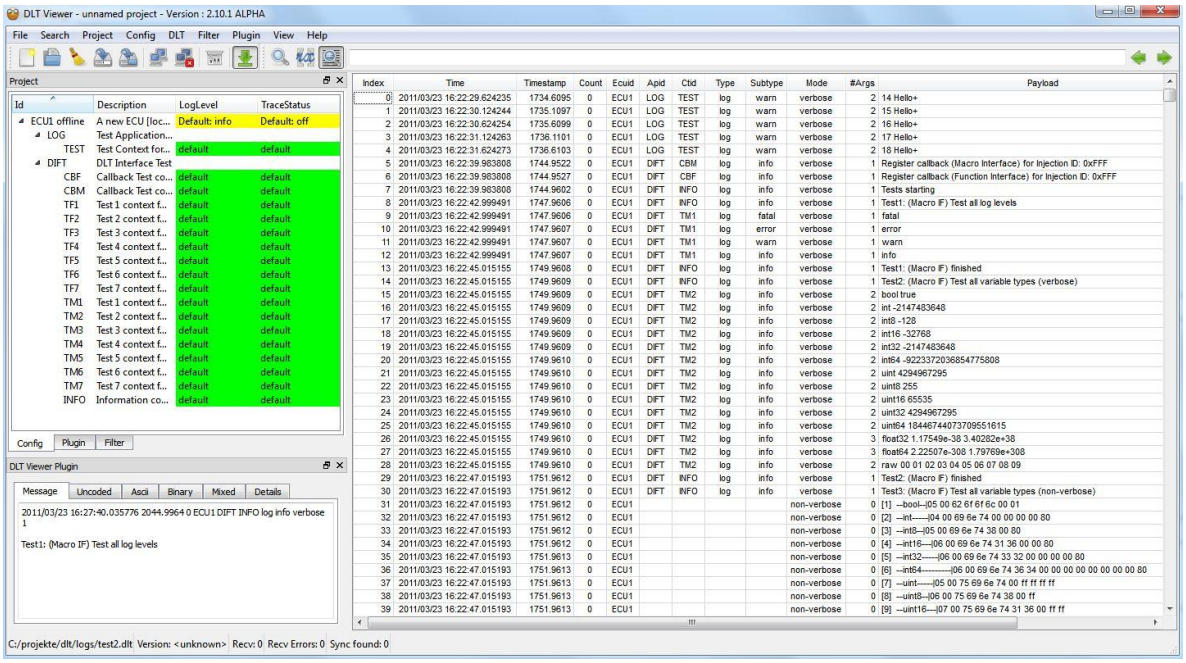


Figure 10: DLT Viewer GUI

7.3.3 Integrating DLT to the Buildroot Linux Image

In order to take the advantage of the logging features described in [5], the DLT package was integrated and used in this project.

The configuration files (`Config.in and dlt-daemon.mk`) that were used to integrate DLT intothe Buidlroot recipe are shown in *Appendix – section 10.4.1.*

7.4 Eclipse IDE

7.4.1 Eclipse IDE Integration on Buildroot

Due to Eclipse is one of the most popular Integrated Development Environment (IDE) used by the software developers; Buildroot integrates with Eclipse in order to ease the development work. This integration simplifies the compilation, remote execution and remote debugging of applications. The way in which Eclipse is integrated with Buildroot is achieved through the *Eclipse Buildroot Toolchain Plugin*. [6]

7.4.2 Activating the Eclipse Buildroot Toolchain Plugin

In summary, the *Eclipse Buildroot Toolchain Plugin* discovers the available Buildroot toolchains by reading a file named `.buildroot-eclipse.toolchains` which is generated in the user `home` directory after Buildroot builds the Linux image. This feature is activated by enabling the `BR2_ECLIPSE_REGISTER` option on the Buildroot recipe. [6]

The way in which this project perfomed the Buildroot and Ecplise integration was following the tutorials described in [6], for this reason this document will not show how the integration between Buildroot and Eclipse is achieved.

Finally, as can be seen in *Appendix – section 10.1.1*, the `BR2_ECLIPSE_REGISTER` option is enabled; thus, the Eclipse IDE was used to develop a test application (middleware) which will be explained later.

7.5 GUI Development

Until now, this Chapter has described how the Buildroot Linux image had been fully customized and optimized. Also, it mentions what important packages (Qt libraries, DLT) were integrated to the Linux image in order to be used on this Cluster GUI project. Now it is the turn for the GUI development.

7.5.1 GUI Development on Qt Framework

As showed on section *5.2 - GUI,* the Cluster GUI shall be developed using the Qt graphic libraries (Qt Framework). Thus, this section explains how the GUI was developed using these graphic libraries.

7.5.1.1 C++ and QML Interaction

It is well known that Qt projects are developed using C++ language, but it is often desirable to mix C++ and QML code. On the Qt framework, QML has the following definition:

> *QML: "*It is a declarative language (JSON-like syntax) that allows user interfaces to be described in terms of their visual components and how they interact and relate with one another". [4]

On the GUI Qt project, the interaction between these two components was implemented on the following way:

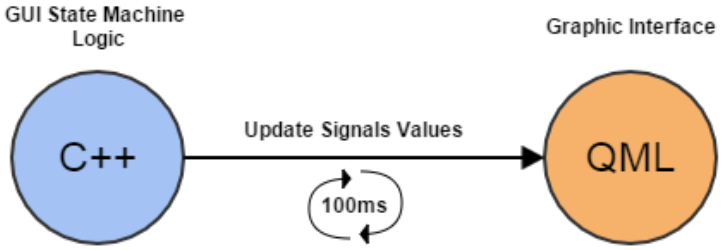


Figure 11: C++ and QML Interaction on GUI Qt Project

As can be seen on *Figure 11*, the user interface code is separated from the application logic code. Thus, the GUI state machine logic was developed using C++ code and every 100ms, this state machine sends the new signals values to QML in order to display them on the graphic interface. Finally, it is noteworthy that this interaction is only on one way: C++ to QML.

7.5.2 GUI State Machine

The state machine that was designed and implemented for the Cluster GUI is described on *Figure 12*:

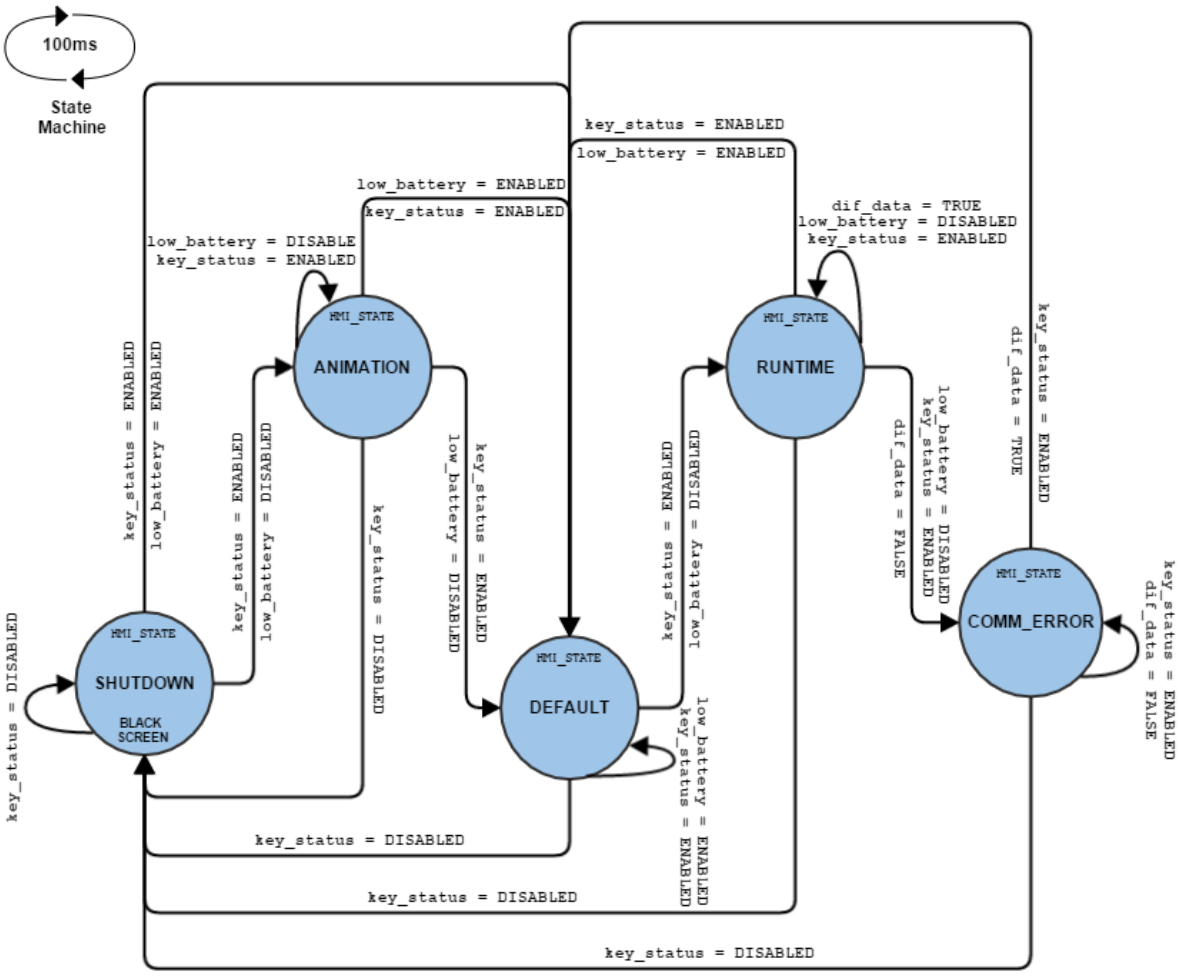


Figure 12: GUI State Machine

As can be seen on the above figure, the state machine has five main states which have the following work flow:

1. *SHUTDOWN*: After the system is started, this is first state for the GUI, in this state a black screen is shown on the cluster display.
    a. If only `key_status`is enabled, the GUI will switch directly to the ANIMATION state.
    b. If both signals `key_status`and`low_battery`are enabled, the GUI will be moved to the DEFAULT state.
    c. The GUI will remain on this state unless `key_status` is enabled.

2. *ANIMATION*: When the GUI enters on this state a short animation will be shown on the screen. This animation tries to imitate a real car dashboard behavior when the ignition key is activated.
    a. The GUI will remain on this state while`key_status`is enabledand`low_battery` is disabled.
    b. If during the animation process `low_battery` is disabled and `key_status` remains enabled, the animation process will be interrupted and the GUI will switch to the DEFAULT state.
    c. If the animation process is finished, the GUI will be automatically moved to the DEAULT state.
    d. If `key_status` is disabled during the animation process, the GUI will be moved to the SHUTDOWN state.
3. *DEFAULT:* Here, almost all indicators and gauges are shown according to default values; only the `key_status` and `low_battery` indicators are shown according to the signal values read from the shared memory. This state is the starting point for others states.
    a. The GUI will remain on this state while `key_status`and `low_battery` are enabled.
    b. If after a 100ms cycle, the `key_status` and `low_battery` remain on the same value (enabled and disabled respectively), the GUI will be moved to the RUNTIME state.
    c. If `key_status` is disabled, the GUI will be moved to the SHUTDOWN state.
4. *RUNTIME:* This is the main state for the GUI; here all the signal values that are read from the shared memory are constantly updated on the screen.
    a. The GUI will remain on this state while `key_status`is enabled, `low_battery` disabled and `dif_data` is true.
    b. If `key_status` and `low_battery`are enabled, the GUI will switch to DEFAULT state.
    c. If `key_status` and `low_battery` remain on the same value (enabled and disabled respectively), but `dif_data` is changed to false, the GUI will be moved to the COMMUNICATION_ERROR state.
    d. If `key_status` is disabled, the GUI will be moved to the SHUTDOWN state.
5. *COMMUNICATION_ERROR:* If the GUI enters in this state means that the signal values read from the shared memory have not changed during a period of time and a possible communication error between the GUI and the CAN driver (middleware) is happening.
    a. The GUI will remain on this state unless `key_status` is disabled or `dif_data` is true.
    b. If `key_status` remains enabled and `dif_data` is changed to true, the GUI will switch to the DEFAULT state.
    c. If `key_status` is disabled, the GUI will be moved to the SHUTDOWN state.

Because the signal values are read from the shared memory every 100ms, the GUI state will be evaluated on this same period of time.

Note that the state machine behavior is defined by the following three parameters:

- `key_status`: This signal value is used to detect when the car ignition switch is on.
- `low_battery`: This signal value helps to detect when the battery charge is low. Thus, it helps to know when the car has electrical problems.
- `dif_data`: It is a flag that helps to know if the values read from the shared memory have not changed during an established period of time. As a consequence, it helps to detect a possible communication error. In the GUI implementation (Qt Creator project), this data validation was performed through a counter named `hmi_validation_ctr`.

7.5.3 GUI Project Tree

The GUI project was developed on Qt Creator and it was named *Dashboard*. The file structure that was implemented for this *Dashboard* project is shown on *Figure 13*.

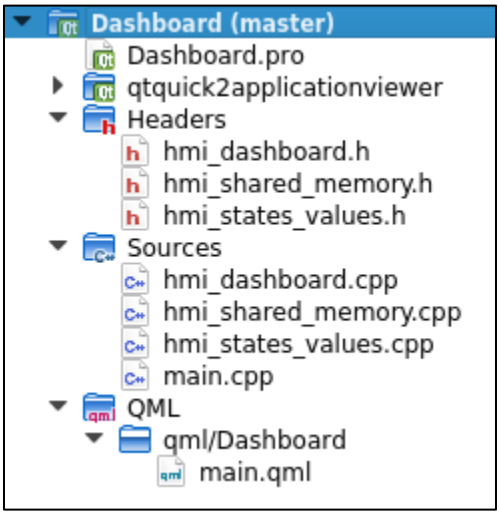


Figure 13: GUI Tree - Dashboard Project on Qt Creator

Project tree description:

- `Dashboard.pro`: This file is automatically created by Qt Creator and it contains all the information required by `qmake` to build the application.
- `qtquick2applicationviewer/`: This directory is also generated by Qt Creator and it contains all the classes needed to use QtQuick Module (QML) on the application.
- `Headers/`: This directory contains all the headers files used on the source files; every source file has its own header file.
    - `hmi_dashboard.h`: It contains the definition of the HMI states used on the state machine and the Window class definition that allows to show the GUI.
    - `hmi_shared_memory.h`: It has the structure definition for the shared memory section and all the function prototypes of hmi_shared_memory.cpp
    - `hmi_states_values.h`: It only contains the function prototypes of hmi_states_values.cpp

- `Sources/`: This directory contains all the source files used on the application.
  - `hmi_dashboard.cpp`: It contains the state machine engine of the application.
  - `hmi_shared_memory.cpp`: It has all the functions that make possible to access to the shared memory section.
  - `hmi_states_values.cpp`: It contains functions that send the signal values to the QtQuick Module (QML) according to a specific HMI state.
  - `main.cpp`: Here the configuration for DLT, Timer and signal to slot connection are set.
- `QML/`: This directory contains all the QML files used to create the graphic interface.
  - `Qml/Dashboard`
    - `main.qml`: This file contains all the graphic elements used on the graphic interface. These elements are shown according the received signal values.

The source code of this Dashboard project (GUI) can be seen in *Appendix – section 10.5.*

7.5.4 GUI – Final Result

The *Figure 14* shows the final result that was obtained for the Cluster GUI (Dashboard project on Qt Creator). As can be seen on this figure, all the cluster indicators and gauges that were included on this GUI are displayed.

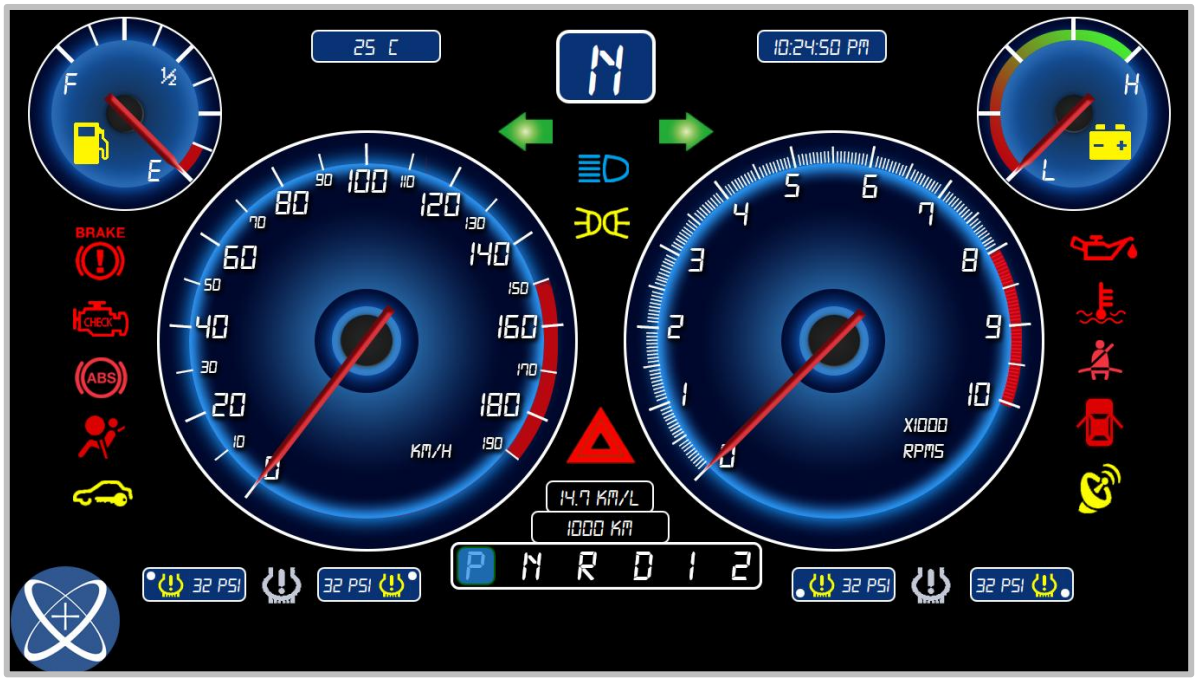


Figure 14: Cluster GUI - Final Result

8 Functional Testing

The final stage for the Cluster GUI project is the functional testing. During this stage, several issues were identified and solved. As a consequence, this stage helped to improve the functional behavior and software quality of this project.

In order to perform the functional testing on this project a simple middleware was implemented. The middleware tries to simulate the CAN driver functionality with out be connected with a real CAN network. The middleware functionality is explained in the following sections.

8.1 Middleware Functionality

First of all, the middleware was designed and developed only to perform the functional testing for the Cluster GUI project. The main goal for it is to evaluate the behavior of indicators and gaugeson runtime.

When the middleware is executed, it has the following functionality*:*

1. It initializes the shared memory section for all the signals used on the Cluster GUI.
2. Through a console menu, it allows to change a signal value of the shared memory section. This console menu can be seen on *Figure 16.*
3. Before the entered value is written on the shared memory, it is evaluated according the signal data type.
4. Finally, it de-initializes the shared memory section when it is closed.

8.2 Middleware Project Tree

The middleware was developed on the Eclipse IDE trough the *Eclipse Buildroot Toolchain Plugin* (cross-compiling). The *Figure 15* shows the file structure implemented for the middleware project on the Eclipse IDE.

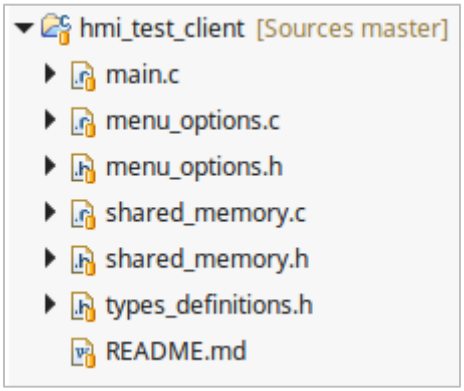


Figure 15: Eclipse Middleware Project Tree

Project tree description:

- `hmi_test_client`: Name of the middleware project on the Eclipse IDE.

- `main.c`: Functions that allow the middleware capabilities are called from here.
- `menu_options.c` and `menu_options.h`: These files allow to show the main menu and get the signal value entered by the user on the linux cosole.
- `shared_memory.c` and `shared_memory.h`: They have functions that make possible to initialize, write and de-initialize the shared memory.
- `types_definitions.h`: Here, the shared memory section is set through a structure definition that contains all the signals used in the GUI.
- `README.md`: This is a help file.

The source code of this middleware project (`hmi_test_client` on Eclipse IDE) can be seen in *Appendix – section 10.6.*

8.3 General Test Case

As a part of the functional test, a general test case was created. Through the middleware, this test checks the behavior of every indicator and gauge shown on the Cluster GUI when its signal value in the shared memory is altered.

Here is an example of this test case using the SPEED signal:

Objective:

- The Cluster GUI shall display the SPEED in the expected values.

Test:

1. Run the middleware application (`hmi_test_client` binary).
2. Select option “x” from the main menu in order to change the SPEED value (*Figure 16*).
3. From the sub-menu enter the desire value, in this case 60 (*Figure 17*).

Expected Result:

- The SPEED value shown on the Cluster GUI shall be 60.

Actual Result:

- The SPEED value shown on the Cluster GUI is 60 (*Figure 18*).

According to the actual result, this test can be treated as PASSED.

Through the main menu, this test case was performed on all indicators and gauges using different signals values.

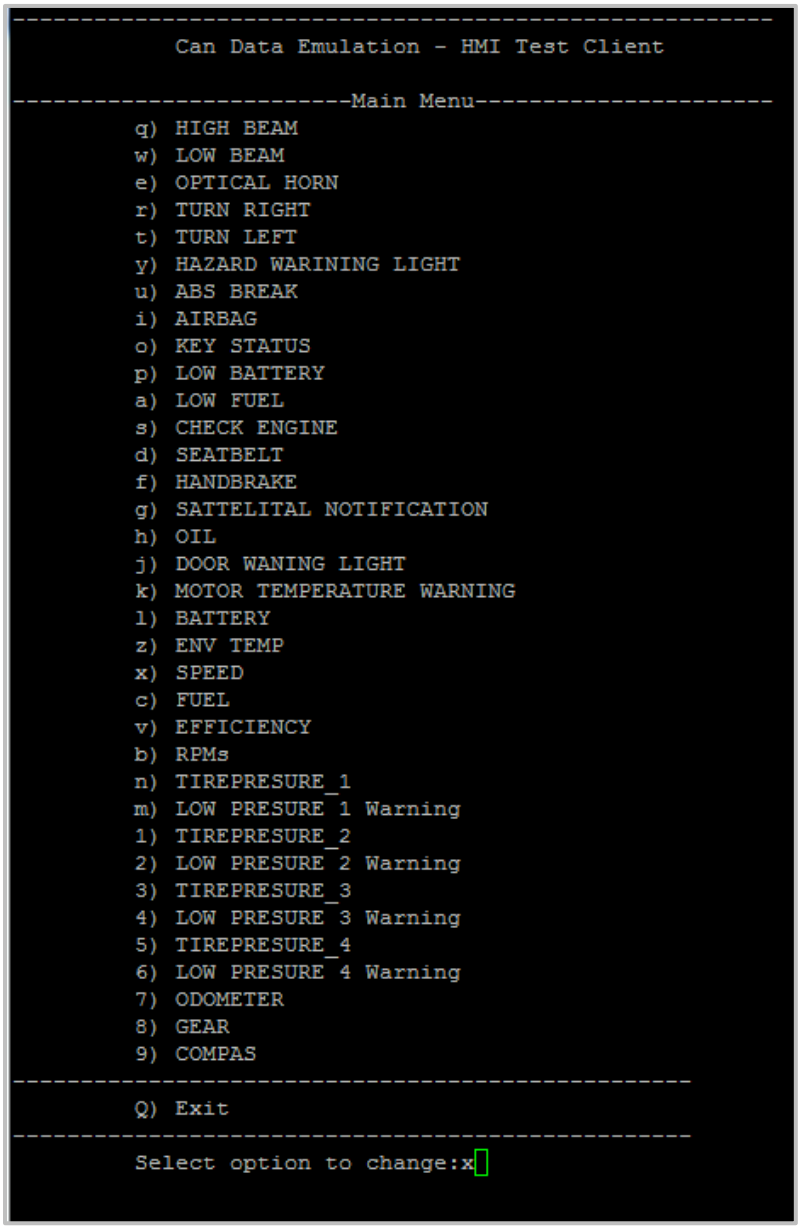


Figure 16: Middleware - Main Menu

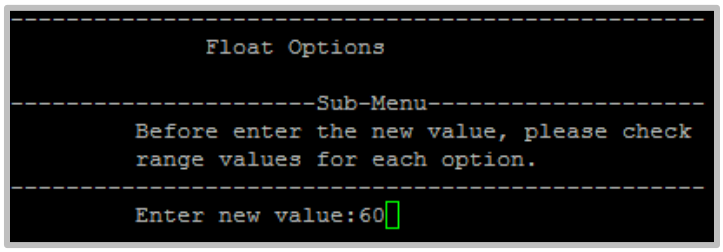


Figure 17: Middleware - Sub-Menu – Changing a Signal Value

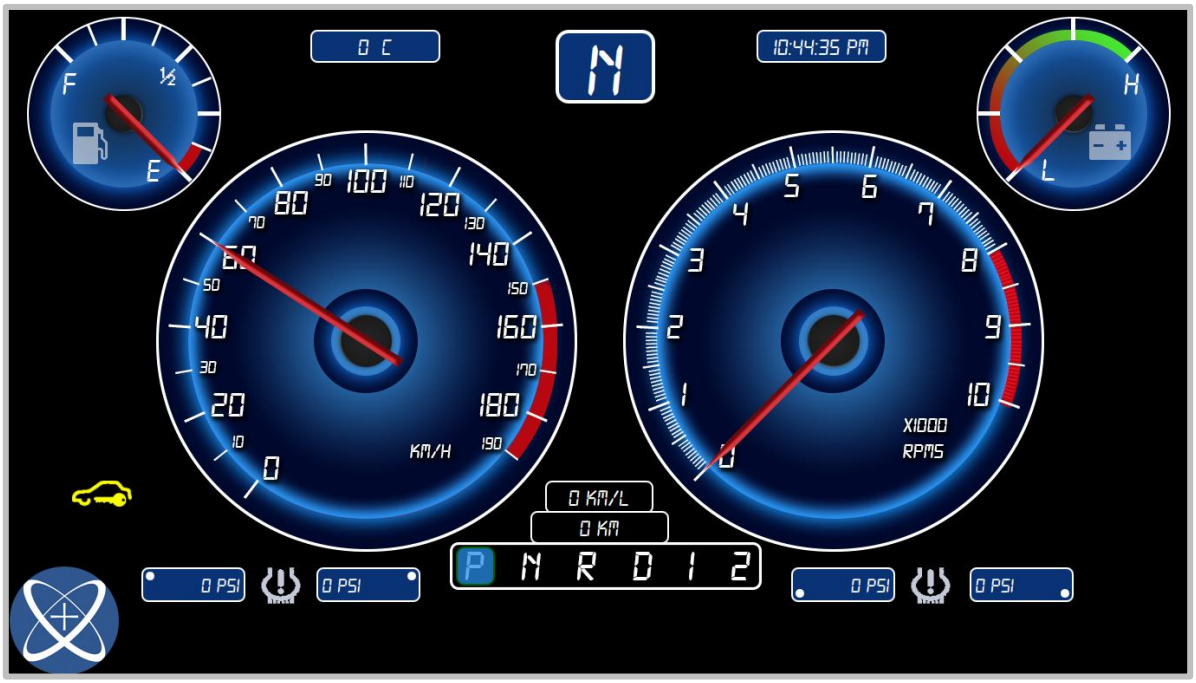


Figure 18: Cluster GUI - New SPEED value is displayed (60 km/h).

9 Conclusions

The design and implementation of this Cluster GUI device was very illustrative in terms of how a development process cycle can be used on a real project. It was a big step to know and understand how automotive projects are handled by the most important companies. Furthermore, the experience gathered between teachers and students was very helpful to create a work environment similar to these companies.

On the other hand, the project also made a great impact on how things are planned against possible improvements and upgrades. It is a common mistake to build something only by fulfilling immediate needs, leaving behind the chance to make any expansions. Here, the knowledge acquired through the work methodology used in this project is invaluable.

One of the most important topics on this project is the requirements. It is very common to start a project without the calculation of effort, the resources, the tools and the knowledge at disposition. For this reason, special care was taken on the definition of the project requirements. Note that the skills needed to identify each requirement were provided by this specialization program.

In terms of the project itself, important achievements were made. An optimized and customized Linux operative system was successfully created, QT graphics libraries were implemented for the GUI and the communication between the middleware and application (GUI) was possible through the Shared Memory IPC. Consequently, this project can be treated as a platform.

Finally, this project was a very rewarding one. It allowed to create experience in the automotive industry, enabled creative thinking, problem solving and helped the engineer to fully understand the things on a real job. Thus, future generations can update this project in hope of a better product.

10 Appendix

10.1 Buildroot Configuration Files (RPi)

10.1.1 raspberrypi_defconfig - recipe

```
################################################################################
BR2_arm=y
BR2_arm1176jzf_s=y
BR2_JLEVEL=0
BR2_CCACHE=y
BR2_ENABLE_DEBUG=y
BR2_DEBUG_2=y
BR2_OPTIMIZE_2=y
BR2_TOOLCHAIN_EXTERNAL=y
BR2_TOOLCHAIN_EXTERNAL_RASPBERRYPI_ARM=y
BR2_TOOLCHAIN_EXTERNAL_DOWNLOAD=y
BR2_TOOLCHAIN_EXTERNAL_PREFIX="arm-raspberrypi-linux-gnueabi"
BR2_TOOLCHAIN_EXTERNAL_GLIBC=y
# BR2_PACKAGE_GDB=y
BR2_TOOLCHAIN_EXTERNAL_GDB_SERVER_COPY=y
BR2_ENABLE_LOCALE_PURGE=y
BR2_ENABLE_LOCALE_WHITELIST="C en_US"
# BR2_SOFT_FLOAT is not set
BR2_TARGET_OPTIMIZATION="-pipe -mfloat-abi=hard -mfpu=vfp -
mtune=arm1176jzf-s -march=armv6zk"
BR2_ECLIPSE_REGISTER=y
BR2_TARGET_GENERIC_HOSTNAME="raspberrypi"
BR2_TARGET_GENERIC_ISSUE="Welcome to Cluster_Pi SDK"
BR2_ROOTFS_DEVICE_CREATION_DYNAMIC_UDEV=y
BR2_ROOTFS_SKELETON_CUSTOM=y
BR2_ROOTFS_SKELETON_CUSTOM_PATH="board/raspberrypi/skeleton"
BR2_TARGET_GENERIC_GETTY_PORT="tty3"
BR2_ROOTFS_POST_BUILD_SCRIPT="board/raspberrypi/post-build.sh"
BR2_PACKAGE_BUSYBOX_SHOW_OTHERS=y
BR2_PACKAGE_GZIP=y
BR2_PACKAGE_DLT_DAEMON=y
BR2_PACKAGE_OPROFILE=y
BR2_PACKAGE_STRACE=y
BR2_PACKAGE_COREUTILS=y
BR2_PACKAGE_FINDUTILS=y
BR2_PACKAGE_GAWK=y
BR2_PACKAGE_GREP=y
BR2_PACKAGE_TAR=y
BR2_PACKAGE_FBSET=y
BR2_PACKAGE_LIBERATION=y
BR2_PACKAGE_DBUS_GLIB=y
BR2_PACKAGE_UDEV_RULES_GEN=y
BR2_PACKAGE_UDEV_ALL_EXTRAS=y
BR2_PACKAGE_USBMOUNT=y
BR2_PACKAGE_USBUTILS=y
BR2_PACKAGE_PYTHON=y
BR2_PACKAGE_PYTHON_BZIP2=y
# BR2_PACKAGE_LIBVORBIS=y
BR2_PACKAGE_BEECRYPT=y
# BR2_PACKAGE_SQLITE=y
```

```
BR2_PACKAGE_FONTCONFIG=y
BR2_PACKAGE_JPEG=y
BR2_PACKAGE_LIBPNG=y
BR2_PACKAGE_NEON=y
BR2_PACKAGE_NEON_ZLIB=y
BR2_PACKAGE_NEON_SSL=y
BR2_PACKAGE_NEON_LIBXML2=y
# BR2_PACKAGE_LIBSOCKETCAN is not set
BR2_PACKAGE_BOOST=y
BR2_PACKAGE_LIBCAP=y
BR2_PACKAGE_PCRE=y
BR2_PACKAGE_LIBXSLT=y
# BR2_PACKAGE_BLUEZ_UTILS=y
# BR2_PACKAGE_BLUEZ_UTILS_COMPAT=y
# BR2_PACKAGE_BLUEZ_UTILS_AUDIO=y
# BR2_PACKAGE_BLUEZ_UTILS_USB=y
# BR2_PACKAGE_BLUEZ_UTILS_WIIMOTE=y
# BR2_USE_MMU=y
# BR2_PACKAGE_CAN_UTILS is not set
BR2_PACKAGE_DHCP=y
BR2_PACKAGE_DHCP_CLIENT=y
# BR2_PACKAGE_IPROUTE2 is not set
BR2_PACKAGE_NTP=y
# BR2_PACKAGE_NTP_NTPD is not set
BR2_PACKAGE_NTP_NTPDATE=y
BR2_PACKAGE_OPENSSH=y
# BR2_PACKAGE_WGET is not set
BR2_PACKAGE_BASH=y
BR2_PACKAGE_FILE=y
BR2_PACKAGE_SCREEN=y
BR2_PACKAGE_SUDO=y
BR2_PACKAGE_HTOP=y
BR2_PACKAGE_KMOD_TOOLS=y
BR2_PACKAGE_PROCPS=y
BR2_PACKAGE_PSMISC=y
BR2_PACKAGE_RSYSLOG=y
BR2_PACKAGE_LESS=y
BR2_PACKAGE_NANO=y
# BR2_PACKAGE_NANO_TINY is not set
# BR2_PACKAGE_VIM is not set
BR2_PACKAGE_VIDEOCORE=y
BR2_PACKAGE_BOOTLOADER=y
BR2_PACKAGE_QTBASE=y
BR2_PACKAGE_QTXMLPATTERNS=y
BR2_PACKAGE_QTJSBACKEND=y
BR2_PACKAGE_QTDECLARATIVE=y
# BR2_PACKAGE_QTMULTIMEDIA=y
# BR2_PACKAGE_WAYLAND is not set
# BR2_PACKAGE_QTWAYLAND is not set
# BR2_PACKAGE_QTGRAPHICALEFFECTS=y
# BR2_PACKAGE_QTQUICKCONTROLS=y
# BR2_PACKAGE_QTGAMEPAD=y
BR2_TARGET_ROOTFS_TAR_GZIP=y
BR2_LINUX_KERNEL=y
BR2_LINUX_KERNEL_CUSTOM_TARBALL=y
BR2_LINUX_KERNEL_CUSTOM_TARBALL_LOCATION="https://github.com/raspberrypi/
linux/tarball/rpi-3.6.y/7849605f5a86c200aef88624c0c2442e35e39954.tar.gz"
```

```
BR2_LINUX_KERNEL_USE_CUSTOM_CONFIG=y
BR2_LINUX_KERNEL_CUSTOM_CONFIG_FILE="board/raspberrypi/linux-3.6-
raspberrypi.config"
BR2_LINUX_KERNEL_UNCOMPRESSED=y
BR2_LINUX_KERNEL_INSTALL_TARGET=y
BR2_PACKAGE_PORTMAP=y
# BR2_PACKAGE_FFMPEG_NONFREE=y
# BR2_PACKAGE_FFMPEG_FFMPEG is not set
# BR2_PACKAGE_FFMPEG_AVFILTER=y
# BR2_PACKAGE_FFMPEG_ENCODERS="ac3,aac"
# BR2_PACKAGE_FFMPEG_MUXERS="spdif,adts"
# BR2_PACKAGE_FFMPEG_PROTOCOLS="http"
# BR2_PACKAGE_FFMPEG_INDEVS is not set
# BR2_PACKAGE_FFMPEG_OUTDEVS is not set
BR2_PACKAGE_BUSYBOX_CONFIG="board/raspberrypi/busybox-1.20.2.config"
# BR2_PACKAGE_OMXPLAYER=y
# BR2_PACKAGE_XWIIMOTE=y
BR2_INIT_UPSTART=y
BR2_ROOTFS_DEVICE_TABLE="board/raspberrypi/device_table.txt"
BR2_PACKAGE_KBD=y
################################################################################
```

10.1.2 post-build.sh

```
################################################################################
TARGETDIR=$1
# Set root password to 'root'. Password generated with
# mkpasswd, from the 'whois' package in Debian/Ubuntu.
sed-i 's%^root::%root:8kfIfYHmcyQEE:%'$TARGETDIR/etc/shadow
# Point /bin/sh to /bin/bash
ln -T -s /bin/bash $TARGETDIR/bin/sh
# Package the /boot partition
tar-czf $TARGETDIR/../images/boot.tar.gz --exclude=Image -C
$TARGETDIR/boot/.
# remove inittab
rm$TARGETDIR/etc/inittab
# remove rc.conf
rm$TARGETDIR/etc/init/rc.conf
# add task to mount everything
cp board/raspberrypi/mount.conf $TARGETDIR/etc/init/
# add task to set hostname
cp board/raspberrypi/hostname.conf $TARGETDIR/etc/init/
# add task to start getty on tty1
cp board/raspberrypi/tty1.conf $TARGETDIR/etc/init/
# add eth0 dhcp entry into /etc/network/interfaces
cp board/raspberrypi/interfaces $TARGETDIR/etc/network/
# make sure that ntpdate is run before sshd is started
cp board/raspberrypi/ntpdate.conf $TARGETDIR/etc/init/
# start bluetooth daemon
# cp board/raspberrypi/bluetooth.conf $TARGETDIR/etc/init/
# start dlt daemon
cp board/raspberrypi/dlt-daemon.conf $TARGETDIR/etc/init/
# copy HMI Dashboard - Qt Project
cp -R board/raspberrypi/Dashboard/$TARGETDIR/opt/
# start HMI Dashboard
cp board/raspberrypi/dashboard.conf $TARGETDIR/etc/init
################################################################################
```

10.1.3 inittab

```
#############################################################################
# /etc/inittab
#
# This inittab is a basic inittab sample for sysvinit, which mimics
# Buildroot's default inittab for Busybox.
id:1:initdefault:
proc::sysinit:/bin/mount -t proc proc /proc
rwmo::sysinit:/bin/mount -o remount,rw /# REMOUNT_ROOTFS_RW
dpts::sysinit:/bin/mkdir -p /dev/pts
moun::sysinit:/bin/mount -a
host::sysinit:/bin/hostname `cat /etc/hostname`
init::sysinit:/etc/init.d/rcS
1:1:respawn:/sbin/getty 115200 tty1
# Logging junk
mess::sysinit:/bin/touch /var/log/messages
sysl:1:respawn:/usr/sbin/syslogd -n -m 0
klog:1:respawn:/usr/sbin/klogd -n
# Stuff to do for the 3-finger salute
rebo::ctrlaltdel:/sbin/reboot
# Stuff to do before rebooting
sklo:6:wait:/usr/bin/killall klogd
ssys:6:wait:/usr/bin/killall syslogd
umou:6:wait:/bin/umount -a -r
swap:6:wait:/sbin/swapoff -a
#############################################################################
```

10.1.4 hostname.conf

```
#############################################################################
# hostname - set system hostname
#
# This task is run on startup to set the system hostname from
/etc/hostname,
# falling back to "localhost" if that file is not readable or is empty
and
# no hostname has yet been set.
description     "set system hostname"
start on startup
task
exec hostname `cat /etc/hostname`
#############################################################################
```

10.1.5 tty.conf

```
#############################################################################
# tty1 - getty
#
# This service maintains a getty on tty1 from the point the system is
# started until it is shut down again.
start on runlevel [2]
stop on runlevel [!2345]
respawn
exec/sbin/getty 38400 tty1
#############################################################################
```

10.1.6 interfaces

```
##############################################################################
# Configure Loopback
auto lo
iface lo inet loopback
# Configure eth0
auto eth0
iface eth0 inet static
address 192.168.1.11
netmask 255.255.255.0
##############################################################################
```

10.1.7 ntpdate.conf

```
##############################################################################
# ntpdate
#
##
description "Sets the date/time from remote server"
start on started network
exec ntpdate -u 0.us.pool.ntp.org
##############################################################################
```

10.1.8 dlt-daemon.conf

```
##############################################################################
# DLT Daemon
description "dlt-daemon"
start on started dbus
stop on stopping dbus
exec dlt-daemon -d
##############################################################################
```

10.1.9 dashboard.conf

```
##############################################################################
# HMI Dashboard - Cluster PI
description     "Start HMI Dashboard Daemon - Cluster_PI"
start on startup
stop on shutdown
script
export QT_QPA_EGLFS_HIDECURSOR=1
export LD_LIBRARY_PATH=\D_LIBRARY_PATH:/opt/vc/lib
exec/opt/Dashboard/bin/Dashboard 1>/opt/Dashboard/dashboard.log 2>&1&
end script
##############################################################################
```

10.2 Qt Configuration Files – Buildroot

10.2.1 qtbase Module

10.2.1.1 Config.in

```
##############################################################################
config BR2_PACKAGE_QTBASE
      bool "qtbase"
      select BR2_PACKAGE_PKGCONF
      select BR2_PACKAGE_UDEV
      select BR2_PACKAGE_LIBGLIB2
```

```
	select BR2_PACKAGE_ZLIB
	select BR2_PACKAGE_JPEG
	select BR2_PACKAGE_LIBPNG
	select BR2_PACKAGE_TIFF
	select BR2_PACKAGE_FREETYPE
	select BR2_PACKAGE_DBUS
	select BR2_PACKAGE_OPENSSL
	select BR2_PACKAGE_SQLITE
	select BR2_PACKAGE_ALSA_LIB
	select BR2_PACKAGE_VIDEOCORE
	help
		Qt 5 qtbase module
################################################################################
```

10.2.1.2 qtbase.mk

```
################################################################################
QTBASE_VERSION= 5.1.1
QTBASE_SITE=http://download.qt-
project.org/official_releases/qt/5.1/$(QTBASE_VERSION)/submodules/
QTBASE_SOURCE= qtbase-opensource-src-$(QTBASE_VERSION).tar.xz
QTBASE_DEPENDENCIES= host-pkgconf udev libglib2 zlib jpeg libpng tiff
freetype dbus VideoCore openssl sqlite alsa-lib
QTBASE_INSTALL_STAGING= YES
define QTBASE_CONFIGURE_CMDS

	-[ -f $(@D)/Makefile ] &&$(MAKE) -C $(@D) confclean
	(cd $(@D)&& MAKEFLAGS="$(MAKEFLAGS) -j$(PARALLEL_JOBS)" ./configure
\
		-prefix /usr \
		-hostprefix $(HOST_DIR)/usr \
		-release \
		-device pi \
		-make libs \
		-make tools \
		-device-option CROSS_COMPILE=$(TARGET_CROSS) \
		-device-option DISTRO=bsquask \
		-sysroot $(STAGING_DIR) \
		-no-neon \
		-opensource \
		-confirm-license \
	)
endef
define QTBASE_BUILD_CMDS
	$(TARGET_MAKE_ENV)$(MAKE) -C $(@D)
endef
define QTBASE_INSTALL_STAGING_CMDS
	$(MAKE) -C $(@D) install
endef
define QTBASE_INSTALL_TARGET_CMDS
	cp -dpf $(STAGING_DIR)/usr/lib/libQt5*.so.* $(TARGET_DIR)/usr/lib
	cp -dpfr $(STAGING_DIR)/usr/plugins $(TARGET_DIR)/usr
endef
define QTBASE_UNINSTALL_TARGET_CMDS
	-rm $(TARGET_DIR)/usr/lib/libQt*.so.*
endef
$(eval $(generic-package))
################################################################################
```

10.2.2 Qt mkspecs for RPi

10.2.2.1 qmake.conf

```
#####################################################################
# qmake configuration for Broadcom's Raspberry PI
# http://wiki.qt-project.org/Devices/RaspberryPi

include(../common/linux_device_pre.conf)

QT_QPA_DEFAULT_PLATFORM= wayland

QMAKE_LFLAGS           += -Wl,-rpath-link,$$[QT_SYSROOT]/opt/vc/lib

QMAKE_LIBDIR_OPENGL_ES2= $$[QT_SYSROOT]/opt/vc/lib
QMAKE_LIBDIR_EGL= $$QMAKE_LIBDIR_OPENGL_ES2

QMAKE_INCDIR_EGL= $$[QT_SYSROOT]/opt/vc/include \

$$[QT_SYSROOT]/opt/vc/include/interface/vcos/pthreads \

$$[QT_SYSROOT]/opt/vc/include/interface/vmcs_host/linux
QMAKE_INCDIR_OPENGL_ES2= $${QMAKE_INCDIR_EGL}

QMAKE_LIBS_EGL= -lEGL -lGLESv2

contains(DISTRO, squeeze) {
    #Debian Squeeze: Legacy everything
QMAKE_LIBS_OPENGL_ES2= -lGLESv2 -lEGL
QT_QPA_DEFAULT_PLATFORM= eglfs
} else:contains(DISTRO, arch) {
    #On principle: no wizardry required
} else {
    #This is not strictly necessary
    DISTRO_OPTS += deb-multi-arch
    DISTRO_OPTS += hard-float
}

QMAKE_CFLAGS           += \
                          -marm \
                          -mfpu=vfp \
                          -mtune=arm1176jzf-s \
                          -march=armv6zk \
                          -mabi=aapcs-linux

QMAKE_CXXFLAGS= $$QMAKE_CFLAGS

EGLFS_PLATFORM_HOOKS_SOURCES= $$PWD/qeglfshooks_pi.cpp
EGLFS_PLATFORM_HOOKS_LIBS= -lbcm_host

include(../common/linux_device_post.conf)
load(qt_config)

#####################################################################
```

10.2.2.2 qeglfshooks.cpp

```
/*############################################################################*/
#include "qeglfshooks.h"
#include "qeglfscursor.h"

#include <QtDebug>

#include <QtPlatformSupport/private/qeglconvenience_p.h>
#include <QtPlatformSupport/private/qeglplatformcontext_p.h>

#include <bcm_host.h>

QT_BEGIN_NAMESPACE

static DISPMANX_DISPLAY_HANDLE_T dispman_display =0;

static EGLNativeWindowType createDispmanxLayer(const QPoint &pos,const
QSize &size,int z, DISPMANX_FLAGS_ALPHA_T flags)
{
    VC_RECT_T dst_rect;
    dst_rect.x = pos.x();
    dst_rect.y = pos.y();
    dst_rect.width = size.width();
    dst_rect.height = size.height();

    VC_RECT_T src_rect;
    src_rect.x =0;
    src_rect.y =0;
    src_rect.width = size.width()<<16;
    src_rect.height = size.height()<<16;

    DISPMANX_UPDATE_HANDLE_T dispman_update =
vc_dispmanx_update_start(0);

    VC_DISPMANX_ALPHA_T alpha;
    alpha.flags = flags;
    alpha.opacity =0xFF;
    alpha.mask =0;

    DISPMANX_ELEMENT_HANDLE_T dispman_element = vc_dispmanx_element_add(
            dispman_update, dispman_display, z,&dst_rect,0,&src_rect,
            DISPMANX_PROTECTION_NONE,&alpha,(DISPMANX_CLAMP_T
*)NULL,(DISPMANX_TRANSFORM_T)0);

    vc_dispmanx_update_submit_sync(dispman_update);

    EGL_DISPMANX_WINDOW_T *eglWindow =new EGL_DISPMANX_WINDOW_T;
    eglWindow->element = dispman_element;
    eglWindow->width = size.width();
    eglWindow->height = size.height();

return eglWindow;
}

// this function is not part of debian squeeze headers
/*
```

```
extern "C" int VCHPOST_
vc_dispmanx_element_change_attributes(DISPMANX_UPDATE_HANDLE_T update,
    DISPMANX_ELEMENT_HANDLE_T element, uint32_t change_flags, int32_t
layer,
    uint8_t opacity, const VC_RECT_T *dest_rect, const VC_RECT_T
*src_rect,
    DISPMANX_RESOURCE_HANDLE_T mask, VC_IMAGE_TRANSFORM_T transform);
*/
// these constants are not in any headers (yet)
#define ELEMENT_CHANGE_LAYER          (1<<0)
#define ELEMENT_CHANGE_OPACITY        (1<<1)
#define ELEMENT_CHANGE_DEST_RECT      (1<<2)
#define ELEMENT_CHANGE_SRC_RECT       (1<<3)
#define ELEMENT CHANGE MASK RESOURCE  (1<<4)
#define ELEMENT_CHANGE_TRANSFORM      (1<<5)

staticvoid moveDispmanxLayer(EGLNativeWindowType window,const QPoint
&pos)
{
    EGL_DISPMANX_WINDOW_T *eglWindow =static_cast<EGL_DISPMANX_WINDOW_T
*>(window);
    QSize size(eglWindow->width, eglWindow->height);

    VC_RECT_T dst_rect;
    dst_rect.x = pos.x();
    dst_rect.y = pos.y();
    dst_rect.width = size.width();
    dst_rect.height = size.height();

    VC_RECT_T src_rect;
    src_rect.x =0;
    src_rect.y =0;
    src_rect.width = size.width()<<16;
    src_rect.height = size.height()<<16;

    DISPMANX_UPDATE_HANDLE_T dispman_update =
vc_dispmanx_update_start(0);
    vc_dispmanx_element_change_attributes(dispman_update,
                                          eglWindow->element,
                                          ELEMENT_CHANGE_DEST_RECT
/*change_flags*/,
0,
0,
&dst_rect,
NULL,
0,
(DISPMANX_TRANSFORM_T)0);

    vc_dispmanx_update_submit_sync(dispman_update);
}

staticvoid destroyDispmanxLayer(EGLNativeWindowType window)
{
    EGL_DISPMANX_WINDOW_T *eglWindow =static_cast<EGL_DISPMANX_WINDOW_T
*>(window);
    DISPMANX_UPDATE_HANDLE_T dispman_update =
vc_dispmanx_update_start(0);
```

```
    vc_dispmanx_element_remove(dispman_update, eglWindow->element);
    vc_dispmanx_update_submit_sync(dispman_update);
delete eglWindow;
}

class QEglFSPiCursor :public QEglFSCursor
{
public:
QEglFSPiCursor(QEglFSScreen *screen): QEglFSCursor(screen){
QSurfaceFormat platformFormat;
        platformFormat.setDepthBufferSize(24);
        platformFormat.setStencilBufferSize(8);
        platformFormat.setRedBufferSize(8);
        platformFormat.setGreenBufferSize(8);
        platformFormat.setBlueBufferSize(8);
        platformFormat.setAlphaBufferSize(8);
        m_config = q_configFromGLFormat(m_screen->display(),
platformFormat);

        createSurface();
        createContext();
        drawInLayer();
}

~QEglFSPiCursor(){
        eglDestroySurface(m_screen->display(), m_surface);
        destroyDispmanxLayer(m_window);
        eglDestroyContext(m_screen->display(), m_context);
}

void createSurface(){
const QRect cr = cursorRect();
        m_window = createDispmanxLayer(cr.topLeft(), cr.size(),50,
DISPMANX_FLAGS_ALPHA_FROM_SOURCE);
        m_surface = eglCreateWindowSurface(m_screen->display(), m_config,
m_window,NULL);
}

void createContext(){
        eglBindAPI(EGL_OPENGL_ES_API);
        QVector<EGLint> attrs;
        attrs.append(EGL_CONTEXT_CLIENT_VERSION);
        attrs.append(2);
        attrs.append(EGL_NONE);
        m_context = eglCreateContext(m_screen->display(), m_config,
EGL_NO_CONTEXT, attrs.constData());
}

void drawInLayer(){
        eglMakeCurrent(m_screen->display(), m_surface, m_surface,
m_context);

        glClearColor(0,0,0,0);
        glClear(GL_COLOR_BUFFER_BIT);
        draw(QRectF(QPointF(-1,1), QPointF(1,-1)));

        eglSwapBuffers(m_screen->display(), m_surface);
```

```
        eglMakeCurrent(m_screen->display(), EGL_NO_SURFACE,
EGL_NO_SURFACE, EGL_NO_CONTEXT);
}

void changeCursor(QCursor *cursor, QWindow *window) Q_DECL_OVERRIDE {
if(!setCurrentCursor(cursor))
return;

        EGL_DISPMANX_WINDOW_T *eglWindow
=static_cast<EGL_DISPMANX_WINDOW_T *>(m_window);
if(QSize(eglWindow->width, eglWindow->height)!= m_cursor.size){
            eglDestroySurface(m_screen->display(), m_surface);
            destroyDispmanxLayer(m_window);
            createSurface();
}
        drawInLayer();
}

void setPos(const QPoint &pos) Q_DECL_OVERRIDE {
        m_cursor.pos = pos;
        moveDispmanxLayer(m_window, cursorRect().topLeft());
}

void pointerEvent(const QMouseEvent &event) Q_DECL_OVERRIDE {
if(event.type()!= QEvent::MouseMove)
return;
m_cursor.pos = event.pos();
moveDispmanxLayer(m_window, cursorRect().topLeft());
}

void paintOnScreen() Q_DECL_OVERRIDE {}
private:
    EGLConfig m_config;
    EGLContext m_context;
    EGLNativeWindowType m_window;
    EGLSurface m_surface;
};

class QEglFSPiHooks :public QEglFSHooks
{
public:
virtualvoid platformInit();
virtualvoid platformDestroy();
virtual EGLNativeDisplayType platformDisplay()const;
virtual QSize screenSize()const;
virtual EGLNativeWindowType createNativeWindow(const QSize &size,const
QSurfaceFormat &format);
virtualvoid destroyNativeWindow(EGLNativeWindowType window);
virtualbool hasCapability(QPlatformIntegration::Capability cap)const;

QEglFSCursor *createCursor(QEglFSScreen *screen)const{
returnnew QEglFSPiCursor(screen);
}
};

void QEglFSPiHooks::platformInit()
{
```

```
    bcm_host_init();
}

EGLNativeDisplayType QEglFSPiHooks::platformDisplay()const
{
    dispman_display = vc_dispmanx_display_open(0/* LCD */);
return EGL_DEFAULT_DISPLAY;
}

void QEglFSPiHooks::platformDestroy()
{
    vc_dispmanx_display_close(dispman_display);
}

QSize QEglFSPiHooks::screenSize()const
{
uint32_t width, height;
    graphics_get_display_size(0/* LCD */,&width,&height);
return QSize(width, height);
}

EGLNativeWindowType QEglFSPiHooks::createNativeWindow(const QSize
&size,const QSurfaceFormat &format)
{
return createDispmanxLayer(QPoint(0,0), size,1, format.hasAlpha()?
DISPMANX_FLAGS_ALPHA_FROM_SOURCE :
DISPMANX_FLAGS_ALPHA_FIXED_ALL_PIXELS);
}

void QEglFSPiHooks::destroyNativeWindow(EGLNativeWindowType window)
{
    destroyDispmanxLayer(window);
}

bool QEglFSPiHooks::hasCapability(QPlatformIntegration::Capability
cap)const
{
switch(cap){
case QPlatformIntegration::ThreadedPixmaps:
case QPlatformIntegration::OpenGL:
case QPlatformIntegration::ThreadedOpenGL:
case QPlatformIntegration::BufferQueueingOpenGL:
returntrue;
default:
returnfalse;
}
}

QEglFSPiHooks eglFSPiHooks;
QEglFSHooks *platformHooks =&eglFSPiHooks;

QT_END_NAMESPACE
/*#########################################################################*/
```

10.3 Guide to configure Qt Creator for Cross-Compiling

This configuration allows creating programs using the Qt libraries (5.1.1) on a remote device (Raspberry PI) with the generated Linux image.

- Requirements:
  - Qt libraries 5.1.1 and Qt Creator must be installed on the host computer.
  - Cluster_PI_Build folder must be available on the host computer.
  - Cluster_PI image must be loaded on the RPI.
- Steps to configure Qt Creator:
  1. Open Qt Creator and go to ***Tools -> Options***
  2. Then ***Build & Run*** menu ->***Compilers*** tab
     - ***Add -> GCC***
       - Name: ***GCC_RaspberryPI***
       - Press ***Browse*** button and select the gcc file from the generated Custer_PI_Build folder, in this case ***Compilerpath***: `/.../output/host/usr/bin/arm-raspberrypi-linux-gnueabi-gcc`
       - Press ***Apply*** button
       - Name: ***GCC_RaspberryPI***
  3. Go to ***Qt Versions*** tab to add the location of qmake
     - Press ***Add*** button
       - Select the path where qmake is located ->`/.../output/host/usr/bin/qmake`
       - Version name: ***Qt_RaspberryPI***
       - Press ***Apply*** button
  4. Go to ***Kits*** tab to add a kit
     - Press ***Add*** button
       - Name: ***RaspberryPI***
       - Device type: ***Generic Linux Device***
       - Press ***Manage*** button
         - Press ***Add*** button
         - Select ***Generic Linux Device***
         - Press ***Start Wizard*** button and set the following values:
           - The name to identify this configuration: ***RaspberryPI***
           - The device's host name or IP address: ***192.168.1.11***
           - The user name to log into the device: ***root***
           - The authentication type : ***Password***
           - The user's password: ***root***
           - Press ***Next*** button
           - Press ***Finish*** button - Verify that the Device was successfully tested.
           - Press ***Close*** button

- Press ***Apply*** button

- Again go to ***Build & Run*** menu ->***Kits*** tab in order to finish the kit configuration
  - Device: RaspberryPI ***(default for Generic Linux)***
  - Sysroot: `/.../output/staging`
  - Compiler: ***GCC_RaspberryPI***
  - Qt version: ***Qt_RaspberryPI***

5. Finally, Qt Creator is configured to program on a remote device using the Qt Libraries.

10.4 DLT

10.4.1 DLT Configuration Files – Buildroot

10.4.1.1 Config.in

```
################################################################################
config BR2_PACKAGE_DLT_DAEMON
	bool "dlt-daemon"
	help
	  DLT_DAEMON receives log messages from DLT user applications and
temporary storage of log messages if DLT Client is not available.
Transmit log messages to DLT Client and response to control messages.

	  DLT is a reusable open source software component for standardized
logging and tracing in infotainment ECUs based on the AUTOSAR 4.0
standard.

	http://projects.genivi.org/diagnostic-log-trace/documentation
################################################################################
```

10.4.1.2 dlt-daemon.mk

```
################################################################################
#
# dlt-daemon
#
################################################################################

DLT_DAEMON_VERSION= v2.12.0
DLT_DAEMON_SITE=http://git.projects.genivi.org/dlt-daemon.git
DLT_DAEMON_SITE_METHOD= git
DLT_DAEMON_INSTALL_STAGING= YES
DLT_DAEMON_INSTALL_TARGET= YES

$(eval $(cmake-package))

################################################################################
```

10.5 Dashboard Project (GUI) - Source Code

10.5.1 hmi_dashboard.h

```
/*****************************************************************/
/*  File:       hmi_dashboard.h                                  */
/*  Brief:      HMI definitions to use on hmi_dashboard.c        */
/*  Author:     Raul Camacho                                     */
/*  Version:    1.0                                              */
/*  Date:       09/Oct/2015                                   */
/*****************************************************************/

#ifndef HMI_DASHBOARD_H    /* Prevent duplicated includes */
#define HMI_DASHBOARD_H

/*****************************************************************/
/*  HEADER FILE INCLUDES                                         */
/*****************************************************************/
#include <QObject>
#include <QQmlContext>
#include <QtQuick/QQuickView>
#include "qtquick2applicationviewer.h"
#include "hmi_shared_memory.h"
/*****************************************************************/

/*****************************************************************/
/*  TYPES                                                        */
/*****************************************************************/
typedefenum{
    HMI_STATE_COMUNNICATION_ERROR,
    HMI_STATE_DEFAULT,
    HMI_STATE_ANIMATION,
    HMI_STATE_RUNTIME,
    HMI_STATE_SHUTDOWN
}HMI_STATES;

class Window : public QQuickView
{
    Q_OBJECT
public:
    explicit Window(QWindow *parent =0);
    QtQuick2ApplicationViewer *Dashboard;
signals:

public slots:
void Update_Can_Data();
};
/*****************************************************************/

#endif  /* HMI_DASHBOARD_H */

/*****************************************************************/
```

10.5.2 hmi_shared_memory.h

```
/*****************************************************************/
/*  File:       hmi_shared_memory.h                              */
/*  Brief:      HMI definitions to use on hmi_shared_memory  */
/*  Author:     Raul Camacho                                  */
/*  Version:    1.0                                              */
/*  Date:       09/Oct/2015                                   */
/*****************************************************************/

#ifndef HMI_SHARED_MEMORY_H     /* Prevent duplicated includes */
#define HMI_SHARED_MEMORY_H

/*****************************************************************/
/*  TYPES                                                        */
*/
/*****************************************************************/
typedefstruct{
float       battery;
float       env_temp;
float       speed;
float       fuel;
float       efficiency;
uint32_t    odometer;
uint16_t    rpms;
uint8_t           tire_presure_1;
uint8_t           low_presure_1_warning;
uint8_t     tire_presure_2;
uint8_t     low_presure_2_warning;
uint8_t           tire_presure_3;
uint8_t           low_presure_3_warning;
uint8_t     tire_presure_4;
uint8_t     low_presure_4_warning;
uint8_t     gear;
uint8_t     compass;
uint8_t     high_beam;
uint8_t     low_beam;
uint8_t     optical_horn;
uint8_t     turn_right;
uint8_t     turn_left;
uint8_t     hazard_warning_light;
uint8_t     abs_break;
uint8_t     airbag;
uint8_t     key_status;
uint8_t     low_battery;
uint8_t     low_fuel;
uint8_t     check_engine;
uint8_t     seat_belt;
uint8_t     hand_brake;
uint8_t     sattelital_notification;
uint8_t     oil;
uint8_t     door_warning_light;
uint8_t     motor_temperature_warning;
} tshared_memory;
/*****************************************************************/
```

```
/*****************************************************************/
/*  FUNCTION PROTOTYPES                                          */
/*****************************************************************/
uint8_t hmi_init_shared_memory(void);
void hmi_deinit_shared_memory(void);
uint8_t hmi_compare_data(tshared_memory* valid_data);
void hmi_copy_data(tshared_memory* valid_data);
/*****************************************************************/

#endif      /* HMI_SHARED_MEMORY_H */

/*****************************************************************/
```

10.5.3 hmi_states_values.h

```
/*****************************************************************/
/*  File:       hmi_states_values.h                              */
/*  Brief:      HMI definitions to use on hmi_states_values.c    */
/*  Author:     Raul Camacho                                     */
/*  Version:    1.0                                              */
/*  Date:       09/Oct/2015                                      */
/*****************************************************************/

#ifndef HMI_STATES_VALUES_H     /* Prevent duplicated includes */
#define HMI_STATES_VALUES_H

/*****************************************************************/
/*  FUNCTION PROTOTYPES                                          */
/*****************************************************************/
void hmi_state_shutdown_event(QtQuick2ApplicationViewer *hmi_dashboard);
void hmi_state_animation_event(QtQuick2ApplicationViewer *hmi_dashboard);
void hmi_state_runtime_event(tshared_memory* data,
QtQuick2ApplicationViewer *hmi_dashboard);
void hmi_state_default_event(QtQuick2ApplicationViewer *hmi_dashboard);
/*****************************************************************/

#endif      /* HMI_STATES_VALUES_H */

/*****************************************************************/
```

10.5.4 hmi_dashboard.cpp

```
/*****************************************************************/
/*  File:       hmi_dashboard.cpp                                */
/*  Brief:      HMI Dashboard functions                          */
/*  Author:     Raul Camacho                                     */
/*  Version:    1.0                                              */
/*  Date:       09/Oct/2015                                      */
/*****************************************************************/

/*****************************************************************/
/*  HEADER FILE INCLUDES                                         */
/*****************************************************************/
#include <dlt/dlt.h>
#include "hmi_dashboard.h"
#include "hmi_shared_memory.h"
#include "hmi_states_values.h"
/*****************************************************************/
```

```
/*****************************************************************************/
/*  LOCAL VARIABLES                                                          */
/*****************************************************************************/
/* HMI states */
HMI_STATES hmi_state;
/* Struct for valid data from Shared Memory - Can Protocol */
tshared_memory hmi_valid_data;
/*****************************************************************************/

/*****************************************************************************/
/*  MACROS                                                                   */
/*****************************************************************************/
/* Import HMI context already registered on DLT */
DLT_IMPORT_CONTEXT(hmi_context)
/*****************************************************************************/

/*****************************************************************************/
/*  LOCAL DEFINES                                                            */
/*****************************************************************************/
#define HMI_VALIDATION_CTR  19
#define ENABLED     1
#define DISABLED    0
#define HMI_ANIMATION_CTR_LIMIT     21
/*****************************************************************************/

/*****************************************************************************/
/*  FUNCTION DEFINITIONS                                                     */
/*****************************************************************************/

/*****************************************************************************/
/* Name:    Window Object
*  Brief:   Declare Dashboard object
*  Author:  Raul Camacho
*/
Window::Window(QWindow *parent):
    QQuickView(parent)
{
    Dashboard = new QtQuick2ApplicationViewer(this);
/* Run shutdown event - Fist event on HMI Dashboard initialization */
    hmi_state_shutdown_event(Dashboard);
/* Initial HMI state*/
    hmi_state = HMI_STATE_SHUTDOWN;
    DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("INITIAL HMI DASHBOARD
STATE: HMI_STATE_SHUTDOWN"));
/* Init all hmi_valid_data struct values to 0  */
    memset(&hmi_valid_data,0,sizeof(hmi_valid_data));
    DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("Init hmi_valid_data
structure values to 0"));
    Dashboard->setMainQmlFile(QStringLiteral("qml/Dashboard/main.qml"));
    Dashboard->showExpanded();
}
```

```
/*****************************************************************/
/* Name:    Update_Can_Data
*  Brief:   Funtion to be called every 100ms to update valid data from
shared memory (CAN).
*           SLOT connected to timer SIGNAL.
*  Author:  Raul Camacho
*  Param:   void
*  Return:  void
*/
void Window::Update_Can_Data()
{
/* Local variables*/
uint8_t hmi_shmem_res;
uint8_t hmi_cmp_data_res;
staticuint8_t hmi_validation_ctr =0;
staticuint8_t hmi_animation_ctr =0;


DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING(__FUNCTION__),DLT_STRING("fun
ction called"));

/* Init shared memory */
    hmi_shmem_res = hmi_init_shared_memory();
if(hmi_shmem_res ==0){
/* Compare valid_data with shared memory */
        hmi_cmp_data_res = hmi_compare_data(&hmi_valid_data);
/* Check if at least one value was changed */
if( hmi_cmp_data_res ==0){
/* No value was changed */
            DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("No value was
changed - Iqual structures (Shared memory and Valid data)"));
hmi_validation_ctr++;
            DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("Increment
HMI_VALIDATION_CTR:"),DLT_INT(hmi_validation_ctr));
/* Check if a communication error occurs */
if(hmi_validation_ctr > HMI_VALIDATION_CTR && hmi_state !=
HMI_STATE_ANIMATION){
                hmi_state = HMI_STATE_COMUNNICATION_ERROR;
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_COMUNNICATION_ERROR"));
hmi_validation_ctr =0;

DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("COMUNNICATION_ERROR occurs,
HMI_VALIDATION_CTR:"),DLT_INT(hmi_validation_ctr));
}elseif(hmi_validation_ctr < HMI_VALIDATION_CTR && hmi_state ==
HMI_STATE_ANIMATION){
/* Skip hmi_validation_ctr from HMI Animation */
                hmi_validation_ctr--;

DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI_STATE_ANIMATION running,
decrement HMI_VALIDATION_CTR:"),DLT_INT(hmi_validation_ctr));
}
}else{
/* At least one shared memory value was changed */
            DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("Diferent
structures (Shared memory and Valid data)"));
/* Copy shared memory values to hmi_valid_data */
```

```
            hmi_copy_data(&hmi_valid_data);
            DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("Copy values from
Shared Memory to Valid data)"));
            hmi_validation_ctr =0;
}
}else{
        DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("Unable to read
SHARED MEMORY - Stay on the same HMI_STATE"));
return;

}
/* Deinit shared memory */
    hmi_deinit_shared_memory();

/* HMI Dashboard states handler */
switch(hmi_state)
{
case HMI_STATE_SHUTDOWN:
{
            DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("ENTER TO HMI
DASHBOARD STATE: HMI_STATE_SHUTDOWN"));

if(hmi_valid_data.key_status == DISABLED){
                hmi_state = HMI_STATE_SHUTDOWN;
                hmi_state_shutdown_event(Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_SHUTDOWN"));
}elseif(hmi_valid_data.low_battery == ENABLED){
                hmi_state = HMI_STATE_DEFAULT;
                hmi_state_default_event(Dashboard);
                Dashboard->rootContext()-
>setContextProperty("low_battery", ENABLED);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_DEFAULT - Low Batery"));
}else{
                hmi_state = HMI_STATE_ANIMATION;
                hmi_state_animation_event(Dashboard);
                hmi_animation_ctr++;
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_ANIMATION"));
}
break;
}
case HMI_STATE_ANIMATION:
{
            DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("ENTER TO HMI
DASHBOARD STATE: HMI_STATE_ANIMATION"));

if(hmi_valid_data.key_status == DISABLED){
                hmi_state = HMI_STATE_SHUTDOWN;
                hmi_state_shutdown_event(Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_SHUTDOWN"));
}elseif(hmi_valid_data.low_battery == ENABLED){
                hmi_state = HMI_STATE_DEFAULT;
                hmi_state_default_event(Dashboard);
```

```
                Dashboard->rootContext()-
>setContextProperty("low_battery", ENABLED);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_DEFAULT - Low Batery"));
}elseif(hmi_animation_ctr <= HMI_ANIMATION_CTR_LIMIT){
                hmi_state = HMI_STATE_ANIMATION;
                hmi_state_animation_event(Dashboard);
                hmi_animation_ctr++;
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_ANIMATION"));
}else{
                hmi_state = HMI_STATE_DEFAULT;
                hmi_animation_ctr =0;
                hmi_state_default_event(Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_DEFAULT"));
}
break;
}
case HMI_STATE_DEFAULT:
{
            DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("ENTER TO HMI
DASHBOARD STATE: HMI_STATE_DEFAULT"));

if(hmi_valid_data.key_status == DISABLED){
                hmi_state = HMI_STATE_SHUTDOWN;
                hmi_state_shutdown_event(Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_SHUTDOWN"));
}elseif(hmi_valid_data.low_battery == ENABLED){
                hmi_state = HMI_STATE_DEFAULT;
                hmi_state_default_event(Dashboard);
                Dashboard->rootContext()-
>setContextProperty("low_battery", ENABLED);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_DEFAULT - Low Batery"));
}else{
                hmi_state = HMI_STATE_RUNTIME;
                hmi_state_runtime_event(&hmi_valid_data, Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_RUNTIME"));
}
break;
}
case HMI_STATE_RUNTIME:
{
            DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("ENTER TO HMI
DASHBOARD STATE: HMI_STATE_RUNTIME"));

if(hmi_valid_data.key_status == DISABLED){
                hmi_state = HMI_STATE_SHUTDOWN;
                hmi_state_shutdown_event(Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_SHUTDOWN"));
}elseif(hmi_valid_data.low_battery == ENABLED){
                hmi_state = HMI_STATE_DEFAULT;
                hmi_state_default_event(Dashboard);
```

```
                Dashboard->rootContext()-
>setContextProperty("low_battery", ENABLED);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_DEFAULT - Low Batery"));
}else{
                hmi_state = HMI_STATE_RUNTIME;
                hmi_state_runtime_event(&hmi_valid_data, Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_RUNTIME"));
}
break;
}
case HMI_STATE_COMUNNICATION_ERROR:
{
            DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("ENTER TO HMI
DASHBOARD STATE: HMI_STATE_COMUNNICATION_ERROR"));

if(hmi_valid_data.key_status == ENABLED && hmi_cmp_data_res ==1){
                hmi_state = HMI_STATE_DEFAULT;
                hmi_state_default_event(Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_DEFAULT"));
}elseif(hmi_valid_data.key_status == ENABLED){
                hmi_state = HMI_STATE_COMUNNICATION_ERROR;
                hmi_state_default_event(Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_COMUNNICATION_ERROR"));
}else{
                hmi_state = HMI_STATE_SHUTDOWN;
                hmi_state_shutdown_event(Dashboard);
                DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI
DASHBOARD CHANGE STATE TO: HMI_STATE_SHUTDOWN"));
}
break;
}
}
}
/*****************************************************************************/
```

10.5.5 hmi_shared_memory.cpp

```
/*****************************************************************************/
/*  File:       hmi_shared_memory.cpp                                        */
/*  Brief:      HMI Shared Memory functions                  */
/*  Author:     Raul Camacho/Rafael Cabrera                                  */
/*  Version:    1.0                                          */
/*  Date:       09/Oct/2015                                  */
/*****************************************************************************/

/*****************************************************************************/
/*  HEADER FILE INCLUDES                                                     */
/*****************************************************************************/
#include <sys/shm.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <dlt/dlt.h>
#include "hmi_shared_memory.h"
```

```
#include "hmi_dashboard.h"
/*****************************************************************************/

/*****************************************************************************/
/*  LOCAL VARIABLES                                                          */
/*****************************************************************************/
tshared_memory     Can_Data;
tshared_memory   *ptr_Can_Data;
key_t Key;
int Id_Shared_Memory;
int Id_Semaphore;
struct sembuf Operation;
/*****************************************************************************/

/*****************************************************************************/
/*  LOCAL DEFINES                                                            */
/*****************************************************************************/
#define FILEKEY         "/bin/ls"
#define KEY             10
#define BYTES           sizeof(Can_Data)
#define SEMAPHORES      1
/*****************************************************************************/

/*****************************************************************************/
/*  MACROS                                                                   */
/*****************************************************************************/
/* Import HMI context already registered on DLT */
DLT_IMPORT_CONTEXT(hmi_context)
/*****************************************************************************/

/*****************************************************************************/
/*  FUNCTION DEFINITIONS                                                     */
/*****************************************************************************/

/*****************************************************************************/
/* Name:    hmi_init_shared_memory
*  Brief:   Function to initialize Shared Memory - IPC
*  Author:  Raul Camacho
*  Param:   void
*  Return:  uint8_t
*/
uint8_t hmi_init_shared_memory(void){


DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING(__FUNCTION__),DLT_STRING("fun
ction called"));
/* Get Key */
    Key = ftok (FILEKEY, KEY);
if(Key ==-1)
{
        DLT_LOG(hmi_context,DLT_LOG_ERROR,DLT_STRING("Error with Key for
Shared Memory and Semaphore"));
return1;
}
/* Get Semaphore Id */
    Id_Semaphore = semget(Key, SEMAPHORES,0777| IPC_CREAT);
if(Id_Semaphore ==-1)
```

```
{
        DLT_LOG(hmi_context,DLT_LOG_ERROR,DLT_STRING("Error with
Semaphore ID, SHARED_MEMORY_ERROR"));
return1;
}
/* Get Shared Memory Id */
    Id_Shared_Memory = shmget (Key, BYTES,0777);
if(Id_Shared_Memory ==-1)
{
        DLT_LOG(hmi_context,DLT_LOG_ERROR,DLT_STRING("Error with Shared
Memory ID, SHARED_MEMORY_ERROR"));
return1;
}
/* Get pointer to Shared Memory*/
    ptr_Can_Data =(tshared_memory *)shmat (Id_Shared_Memory,(char*)0,0);
if(ptr_Can_Data ==NULL)
{
        DLT_LOG(hmi_context,DLT_LOG_ERROR,DLT_STRING("Error getting
shared memory, SHARED_MEMORY_ERROR"));
return1;
}

/* Init semaphore values */
    Operation.sem_num =0;
    Operation.sem_flg =0;
    Operation.sem_op =1;

return0;
}

/*********************************************************************/
/* Name:    hmi_deinit_shared_memory
*  Brief:   Function to deinitialize Shared Memory - IPC
*  Author:  Raul Camacho
*  Param:   void
*  Return:  void
*/
void hmi_deinit_shared_memory(void){


DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING(__FUNCTION__),DLT_STRING("fun
ction called"));
/* Free the shared memory */
    shmdt ((char*)ptr_Can_Data);
return;
}

/*********************************************************************/
/* Name:    hmi_compare_data
*  Brief:   Function to compare data structures (valid data vs shared
memory)
*  Author:  Raul Camacho
*  Param:   tshared_memory*
*  Return:  void
*/
uint8_t hmi_compare_data(tshared_memory* valid_data){
/* Local variables */
```

```
uint8_t cmp_res;

DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING(__FUNCTION__),DLT_STRING("fun
ction called"));
/* Change Semaphore to RED */
    Operation.sem_op =-1;
/* Set the Semaphore to RED */
    semop (Id_Semaphore,&Operation,1);

if(memcmp(ptr_Can_Data, valid_data,sizeof(tshared_memory))==0){
        cmp_res =0;
}else{
        cmp_res =1;
}

/* Change Semaphore to GREEN */
    Operation.sem_op =1;
/* Set the Semaphore to GREEN */
    semop (Id_Semaphore,&Operation,1);

return cmp_res;
}

/*********************************************************************/
/* Name:    hmi_copy_data
*  Brief:   Function to copy data structures (shared memory to valid
data)
*  Author:  Raul Camacho
*  Param:   tshared_memory*
*  Return:  void
*/
void hmi_copy_data(tshared_memory* valid_data){


DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING(__FUNCTION__),DLT_STRING("fun
ction called"));
/* Change Semaphore to RED */
    Operation.sem_op =-1;
/* Set the Semaphore to RED */
    semop (Id_Semaphore,&Operation,1);
/* update Can values - shared memory values to valid data struct */
    memcpy(valid_data, ptr_Can_Data,sizeof(tshared_memory));
/* Change Semaphore to GREEN */
    Operation.sem_op =1;
/* Set the Semaphore to GREEN */
semop (Id_Semaphore,&Operation,1);
}
/*********************************************************************/
```

10.5.6 hmi_states_values.cpp

```
/*********************************************************************/
/*  File:       hmi_states_values.cpp                                 */
/*  Brief:      HMI Dashboard states functions                */
/*  Author:     Raul Camacho                    */
/*  Version:    1.0                                           */
/*  Date:       09/Oct/2015                                   */
/*********************************************************************/
```

```
/*****************************************************************************/
/*  HEADER FILE INCLUDES                                                     */
/*****************************************************************************/
#include <QObject>
#include <QQmlContext>
#include <QtQuick/QQuickView>
#include "qtquick2applicationviewer.h"
#include <time.h>
#include <dlt/dlt.h>
#include "hmi_dashboard.h"
/*****************************************************************************/

/*****************************************************************************/
/*  MACROS                                                                   */
/*****************************************************************************/
/* Import HMI context already registered on DLT */
DLT_IMPORT_CONTEXT(hmi_context)
/*****************************************************************************/

/*****************************************************************************/
/*  LOCAL DEFINES                                                            */
/*****************************************************************************/
#define ENABLED     1
#define DISABLED    0.0
/*****************************************************************************/

/*****************************************************************************/
/*  FUNCTION DEFINITIONS                                                     */
/*****************************************************************************/

/*****************************************************************************/
/* Name:    hmi_state_shutdown_event
*  Brief:   Function to disable (no visible) all values and shows a black
screen when enter
*           to HMI_STATE_SHUTDOWN.
*  Author:  Raul Camacho
*  Param:   QtQuick2ApplicationViewer*
*  Return:  void
*/
void hmi_state_shutdown_event(QtQuick2ApplicationViewer *hmi_dashboard){


DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING(__FUNCTION__),DLT_STRING("fun
ction called"));

/* Set values to Dashboard screen */
    hmi_dashboard->rootContext()->setContextProperty("battery",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("env_temp",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("speed",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("fuel",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("efficiency",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("odometer",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("rpms",DISABLED);
```

```
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_1",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_1_warning",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_2",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_2_warning",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_3",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_3_warning",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_4",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_4_warning",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("gear",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("compass",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("high_beam",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_beam",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("optical_horn",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("turn_right",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("turn_left",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("hazard_warning_light",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("abs_break",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("airbag",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("key_status",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_battery",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_fuel",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("check_engine",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("seat_belt",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("hand_brake",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("sattelital_notification",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("oil",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("door_warning_light",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("motor_temperature_warning",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("shutdown",ENABLED);
}

/*****************************************************************/
/* Name:    hmi_state_animation_event
```

```
*  Brief:   Function to show an animation on Dashboard screen when enter
to
*           HMI_STATE_ANIMATION.
*  Author:  Raul Camacho
*  Param:   QtQuick2ApplicationViewer*
*  Return:  void
*/
void hmi_state_animation_event(QtQuick2ApplicationViewer *hmi_dashboard){
/* Local Counters */
staticuint8_t ctr =0;
staticuint8_t flag =0;


DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING(__FUNCTION__),DLT_STRING("fun
ction called"));

/* Set values to Dashboard screen */
    hmi_dashboard->rootContext()-
>setContextProperty("env_temp",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("efficiency",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("odometer",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_1",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_1_warning",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_2",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_2_warning",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_3",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_3_warning",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_4",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_4_warning",ENABLED);
    hmi_dashboard->rootContext()->setContextProperty("gear",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("compass",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("high_beam",ENABLED);
    hmi_dashboard->rootContext()->setContextProperty("low_beam",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("optical_horn",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("turn_right",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("turn_left",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("hazard_warning_light",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("abs_break",ENABLED);
    hmi_dashboard->rootContext()->setContextProperty("airbag",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("key_status",ENABLED);
```

```
    hmi_dashboard->rootContext()-
>setContextProperty("low_battery",ENABLED);
    hmi_dashboard->rootContext()->setContextProperty("low_fuel",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("check_engine",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("seat_belt",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("hand_brake",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("sattelital_notification",ENABLED);
    hmi_dashboard->rootContext()->setContextProperty("oil",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("door_warning_light",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("motor_temperature_warning",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("shutdown",DISABLED);
/* Set the values for animation */
if(ctr<190&& flag ==0){
        hmi_dashboard->rootContext()->setContextProperty("speed",ctr);
        hmi_dashboard->rootContext()-
>setContextProperty("rpms",ctr*52.63);
        hmi_dashboard->rootContext()-
>setContextProperty("fuel",ctr/2.99);
        hmi_dashboard->rootContext()-
>setContextProperty("battery",ctr/12.25);
        ctr+=19;
}elseif(ctr ==190&& flag ==0){
        flag =1;
        hmi_dashboard->rootContext()->setContextProperty("speed",ctr);
        hmi_dashboard->rootContext()-
>setContextProperty("rpms",ctr*52.63);
        hmi_dashboard->rootContext()-
>setContextProperty("fuel",ctr/2.99);
        hmi_dashboard->rootContext()-
>setContextProperty("battery",ctr/12.25);
return;
}
if(ctr>0&& flag ==1){
        hmi_dashboard->rootContext()->setContextProperty("speed",ctr);
        hmi_dashboard->rootContext()-
>setContextProperty("rpms",ctr*52.63);
        hmi_dashboard->rootContext()-
>setContextProperty("fuel",ctr/2.99);
        hmi_dashboard->rootContext()-
>setContextProperty("battery",ctr/12.25);
        ctr-=19;
}elseif(ctr ==0&& flag ==1){
        flag =0;
        hmi_dashboard->rootContext()->setContextProperty("speed",ctr);
        hmi_dashboard->rootContext()-
>setContextProperty("rpms",ctr*52.63);
        hmi_dashboard->rootContext()-
>setContextProperty("fuel",ctr/2.99);
        hmi_dashboard->rootContext()-
>setContextProperty("battery",ctr/12.25);
```

```
}
}

/*****************************************************************/
/* Name:    hmi_state_default_event
*  Brief:   Function to set default vvalues on Dashboard screen when
enter to
*           HMI_STATE_DEFAULT or others.
*  Author:  Raul Camacho
*  Param:   QtQuick2ApplicationViewer*
*  Return:  void
*/
void hmi_state_default_event(QtQuick2ApplicationViewer *hmi_dashboard){


DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING(__FUNCTION__),DLT_STRING("fun
ction called"));

/* Set values to Dashboard screen */
    hmi_dashboard->rootContext()->setContextProperty("battery",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("env_temp",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("speed",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("fuel",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("efficiency",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("odometer",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("rpms",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_1",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_1_warning",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_2",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_2_warning",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_3",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_3_warning",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_4",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_4_warning",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("gear",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("compass",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("high_beam",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_beam",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("optical_horn",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("turn_right",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("turn_left",DISABLED);
```

```
    hmi_dashboard->rootContext()-
>setContextProperty("hazard_warning_light",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("abs_break",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("airbag",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("key_status",ENABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_battery",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("low_fuel",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("check_engine",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("seat_belt",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("hand_brake",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("sattelital_notification",DISABLED);
    hmi_dashboard->rootContext()->setContextProperty("oil",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("door_warning_light",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("motor_temperature_warning",DISABLED);
    hmi_dashboard->rootContext()-
>setContextProperty("shutdown",DISABLED);
}

/*********************************************************************/
/* Name:    hmi_state_runtime_event
*  Brief:   Function to show runtime values (hmi_valid_data - CAN) on
Dashboard screen when
*           enter to HMI_STATE_RUNTIME.
*  Author:  Raul Camacho
*  Param:   QtQuick2ApplicationViewer*
*  Return:  void
*/
void hmi_state_runtime_event(tshared_memory* data,
QtQuick2ApplicationViewer *hmi_dashboard){


DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING(__FUNCTION__),DLT_STRING("fun
ction called"));

/* Set values to Dashboard screen */
    hmi_dashboard->rootContext()->setContextProperty("battery",data-
>battery);
    hmi_dashboard->rootContext()->setContextProperty("env_temp",data-
>env_temp);
    hmi_dashboard->rootContext()->setContextProperty("speed",data-
>speed);
    hmi_dashboard->rootContext()->setContextProperty("fuel",data->fuel);
    hmi_dashboard->rootContext()->setContextProperty("efficiency",data-
>efficiency);
    hmi_dashboard->rootContext()->setContextProperty("odometer",data-
>odometer);
    hmi_dashboard->rootContext()->setContextProperty("rpms",data->rpms);
```

```
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_1",data->tire_presure_1);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_1_warning",data->low_presure_1_warning);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_2",data->tire_presure_2);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_2_warning",data->low_presure_2_warning);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_3",data->tire_presure_3);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_3_warning",data->low_presure_3_warning);
    hmi_dashboard->rootContext()-
>setContextProperty("tire_presure_4",data->tire_presure_4);
    hmi_dashboard->rootContext()-
>setContextProperty("low_presure_4_warning",data->low_presure_4_warning);
    hmi_dashboard->rootContext()->setContextProperty("gear",data->gear);
    hmi_dashboard->rootContext()->setContextProperty("compass",data-
>compass);
    hmi_dashboard->rootContext()->setContextProperty("high_beam",data-
>high_beam);
    hmi_dashboard->rootContext()->setContextProperty("low_beam",data-
>low_beam);
    hmi_dashboard->rootContext()->setContextProperty("optical_horn",data-
>optical_horn);
    hmi_dashboard->rootContext()->setContextProperty("turn_right",data-
>turn_right);
    hmi_dashboard->rootContext()->setContextProperty("turn_left",data-
>turn_left);
    hmi_dashboard->rootContext()-
>setContextProperty("hazard_warning_light",data->hazard_warning_light);
    hmi_dashboard->rootContext()->setContextProperty("abs_break",data-
>abs_break);
    hmi_dashboard->rootContext()->setContextProperty("airbag",data-
>airbag);
    hmi_dashboard->rootContext()->setContextProperty("key_status",data-
>key_status);
    hmi_dashboard->rootContext()->setContextProperty("low_battery",data-
>low_battery);
    hmi_dashboard->rootContext()->setContextProperty("low_fuel",data-
>low_fuel);
    hmi_dashboard->rootContext()->setContextProperty("check_engine",data-
>check_engine);
    hmi_dashboard->rootContext()->setContextProperty("seat_belt",data-
>seat_belt);
    hmi_dashboard->rootContext()->setContextProperty("hand_brake",data-
>hand_brake);
    hmi_dashboard->rootContext()-
>setContextProperty("sattelital_notification",data-
>sattelital_notification);
    hmi_dashboard->rootContext()->setContextProperty("oil",data->oil);
    hmi_dashboard->rootContext()-
>setContextProperty("door_warning_light",data->door_warning_light);
    hmi_dashboard->rootContext()-
>setContextProperty("motor_temperature_warning",data-
>motor_temperature_warning);
```

```
    hmi_dashboard->rootContext()-
>setContextProperty("shutdown",DISABLED);
}
/*****************************************************************************/
```

10.5.7 main.c

```
/*****************************************************************************/
/*  File:       main.cpp                                                     */
/*  Brief:      HMI Dashboard main function                                  */
/*  Author:     Raul Camacho                                                 */
/*  Version:    1.0                                                          */
/*  Date:       30/Aug/2015                                                  */
/*****************************************************************************/

/*****************************************************************************/
/*  HEADER FILE INCLUDES                                                     */
/*****************************************************************************/
#include <QtGui/QGuiApplication>
#include "qtquick2applicationviewer.h"
#include <QTimer>
#include <dlt/dlt.h>
#include "hmi_dashboard.h"
#include "hmi_shared_memory.h"
#include "hmi_states_values.h"
/*****************************************************************************/

/*****************************************************************************/
/*  MACROS                                                                   */
/*****************************************************************************/
/* Creating logging context - DLT logs */
DLT_DECLARE_CONTEXT(hmi_context);
/*****************************************************************************/

/*****************************************************************************/
/*  FUNCTION DEFINITIONS                                                     */
/*****************************************************************************/

/*****************************************************************************/
/* Name:    main
*  Brief:   Main HMI Dashboard function
*  Author:  Raul Camacho
*  Param:   void
*  Return:  int
*/
int main(int argc,char*argv[])
{
/* Register the application and the context to DLT */
    DLT_REGISTER_APP("CPI","Cluster PI");
    DLT_REGISTER_CONTEXT(hmi_context,"HMI","HMI Context");

    DLT_LOG(hmi_context,DLT_LOG_INFO,DLT_STRING("HMI Dashboard
Initialization"));

    QGuiApplication app(argc, argv);

/* Declare Window Object */
    Window Dashboard;
```

```
/* Declare Timer Object */
    QTimer Timer;

/* Set 100 ms timeout */
    Timer.start(100);

/* Connect timeout signal to Update_Can_Data function - call function
every 100 ms */
    app.connect(&Timer, SIGNAL(timeout()),&Dashboard,
SLOT(Update_Can_Data()));

return app.exec();

/* Unregister the application and the context - DLT */
    DLT_UNREGISTER_CONTEXT(hmi_context);
DLT_UNREGISTER_APP();
}
/*****************************************************************************/
```

10.5.8 main.qml

```
/*****************************************************************************/
/*  File:       main.qml                                                     */
/*  Brief:      HMI Dashboard QML file                                       */
/*  Author:     Raul Camacho                                                 */
/*  Version:    1.0                                                          */
/*  Date:       30/Aug/2015                                                  */
/*****************************************************************************/

/*****************************************************************************/
/*  MODULE IMPORT                                                            */
/*****************************************************************************/
import QtQuick 2.1
/*****************************************************************************/

/*****************************************************************************/
/*  DASHBOARD COMPONENTS                                                     */
/*****************************************************************************/
Rectangle {
    id: root
    width:1366
    height:768
    color:"black"

    FontLoader { id: digital_font; source:"./fonts/digital-7-italic.ttf"}

    Image {
        id: dashboard_panel
        z:6
        source:"images/dashboard_panel.png"
}

    Image {
        id: speed_needle
        x:377
        y:171
        z:7
        source:"images/speed_needle.png"
```

```
        transform: Rotation {
            origin.x:37;
            origin.y:213;
            angle:-142.5+(1.422* speed);
            Behavior on angle {
                NumberAnimation {duration:100}
}
}
        smooth: true
}

    Image {
        id: rpm_needle
        x:917
        y:171
        z:7
        source:"images/rpm_needle.png"
        transform: Rotation {
            origin.x:37;
            origin.y:213;
            angle:-135.2+(2.445*(rpms/100));
            Behavior on angle {
                NumberAnimation {duration:100}
}
}
        smooth: true
}

    Image {
id: fuel_needle
        x:106.5
        y:25.5
z:7
        source:"images/fuel_needle.png"
        transform: Rotation {
            origin.x:28.5;
            origin.y:96.5;
            angle:135-(2.834* fuel);
            Behavior on angle {
                NumberAnimation {duration:100}
}
}
        smooth: true
}

    Image {
        id: battery_needle
        x:1202.5
        y:25.5
        z:7
        source:"images/batery_needle.png"
        transform: Rotation {
            origin.x:28.5;
            origin.y:96.5;
            angle:-135+(11.612* battery);
            Behavior on angle {
NumberAnimation {duration:100}
```

```
}
}
        smooth: true
}

    Rectangle {
        id: compass_panel
        width:115
        height:85
        x:633
        y:25
        z:7
        color:"#093375"
        border.color:"white"
        border.width:4
        radius:15
        Text {
            id: compass_value
            text:if(compass ===0){"N"}elseif(compass
===1){"NE"}elseif(compass ===2){"E"}elseif(compass
===3){"SE"}elseif(compass ===4){"S"}elseif(compass
===5){"SW"}elseif(compass ===6){"W"}elseif(compass ===7){"NW"}
font.pointSize:68
            font.family: digital_font.name
            anchors.centerIn: compass_panel
            color:"white"
}
}

    Image {
        id: right_arrow
        x:759
        y:125
        z:7
        scale:1.3
        opacity: turn_right
        source:"images/arrow.png"
        Behavior on opacity {
              NumberAnimation {duration:100}
}
}

    Image {
        id: left_arrow
        x:573
        y:125
        z:7
        scale:1.3
        mirror: true
        opacity: turn_left
        source:"images/arrow.png"
        Behavior on opacity {
              NumberAnimation {duration:100}
}
}

    Image {
```

```
        id: fuel_static_indicator
        x:-160
        y:-100
        z:7
        scale:0.125
        opacity:0.75
        source:"images/fuel.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: fuel_warning_signal
        x:-160
        y:-100
        z:7
        scale:0.125
        opacity: low_fuel
        source:"images/fuel_warning.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: battery_static_indicator
        x:1017
y:-105
        z:7
        scale:0.095
        opacity:0.7
        source:"images/battery.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: battery_warning_signal
        x:1017
        y:-105
        z:7
        scale:0.095
        opacity: low_battery
        source:"images/battery_warning.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: high_beam_signal
        x:660
        y:170
        z:7
        scale:1.05
```

```
        opacity: high_beam
        source:"images/high_beam.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: low_beam_signal
        x:646
        y:225
        z:7
        scale:0.8
        opacity: low_beam
        source:"images/low_beam.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: hand_brake_signal
        x:75
        y:250
        z:7
        scale:0.9
        opacity: hand_brake
        source:"images/handbrake.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: check_engine_signal
        x:67
        y:333
        z:7
        scale:0.8
        opacity: check_engine
        source:"images/check_engine.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: abs_break_signal
        x:74
        y:400
        z:7
        scale:0.9
        opacity: abs_break
        source:"images/abs_break.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
```

```
}

    Image {
        id: airbag_signal
        x:74
        y:464
        z:7
        scale:0.8
        opacity: airbag
        source:"images/airbag.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: key_status_signal
        x:71
        y:543
        z:7
        scale:0.9
        opacity: key_status
        source:"images/key_status.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: oil_signal
        x:1220
y:260
        z:7
        scale:0.85
        opacity: oil
        source:"images/oil.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: motor_temperature_signal
        x:1225
y:308
        z:7
        scale:0.75
        opacity: motor_temperature_warning
        source:"images/motor_temperature.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: seatbelt_signal
        x:1228
```

```
        y:375
        z:7
        scale:0.75
        opacity: seat_belt
        source:"images/seatbelt.png"
        Behavior on opacity {
NumberAnimation { duration:100}
}
}

    Image {
        id: door_warning_light_signal
        x:1186
        y:403
        z:7
        scale:0.57
        opacity: door_warning_light
        source:"images/door_warning_light.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: sattelital_notification_signal
        x:1228
        y:520
        z:7
        scale:0.85
        opacity: sattelital_notification
        source:"images/sattelital_notification.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: hazard_warning_light_signal
        x:651
        y:465
        z:7
        scale:1.2
        opacity: hazard_warning_light
        source:"images/hazard_warning_light.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Rectangle {
        id: clock_panel
        width:150
        height:38
        x:865
        y:25
        z:7
        color:"#093375"
```

```
        border.color:"white"
        border.width:2
        radius:8
        Text {
            id: clock_value
            text: Qt.formatDateTime(new Date(),"hh:mm:ss AP")
            font.pointSize:20
            font.family: digital_font.name
            anchors.centerIn: clock_panel
            color:"white"
}
        Timer {
            id: clock_timer
            interval:1000
            repeat: true
            running: true
            onTriggered:
{
                clock_value.text =  Qt.formatTime(new Date(),"hh:mm:ss
AP")
}
}
}

    Rectangle {
        id: env_temp_panel
        width:150
height:38
        x:350
        y:25
        z:7
        color:"#093375"
        border.color:"white"
        border.width:2
        radius:8
        Text {
            id: env_temp_value
            text: env_temp +"  C"
            font.pointSize:20
            font.family: digital_font.name
            anchors.centerIn: env_temp_panel
            color:"white"
}
}

    Rectangle {
        id: gear_panel
        width:45
        height:45
        x:if(gear ===0){518}elseif(gear ===1){582}elseif(gear
===2){646}elseif(gear ===3){710}elseif(gear ===4){767}elseif(gear
===5){821}
y:621
        z:7
        color:"#288de9"
border.color:"green"
border.width:1.5
```

```
        radius:10
        opacity:.75
        Behavior on x {
NumberAnimation { duration:300}
}

}

    Image {
        id: tire_pressure_front_panel
        x:280
        y:630
        z:7
        scale:0.9
        opacity:1
        source:"images/tire_pressure.png"
}

    Rectangle {
        id: tire_pressure_front_left
        width:120
        height:40
        x:155
        y:645
        z:7
color:"#093375"
        border.color:"white"
        border.width:2
        radius:8
        Text {
            id: tire_pressure_front_left_value
            text: tire_presure_1 +" PSI "
            font.pointSize:18
            font.family: digital_font.name
            anchors.right: tire_pressure_front_left.right
            anchors.verticalCenter: parent.verticalCenter
            color:"white"
}
}

    Rectangle {
        id: tire_pressure_front_left_indicator
        width:10
        height:10
        x:160
        y:650
        z:8
        color:"white"
        border.color:"white"
        border.width:1
        radius:5
}

    Image {
        id: tire_pressure_front_left_warning
        x:152
        y:630
```

```
        z:8
        scale:0.60
        opacity: low_presure_1_warning
        source:"images/tire_pressure_warning.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Rectangle {
        id: tire_pressure_front_right
        width:120
        height:40
        x:357
        y:645
        z:7
        color:"#093375"
        border.color:"white"
        border.width:2
        radius:8
        Text {
            id: tire_pressure_front_right_value
            text:" "+ tire_presure_2 +" PSI "
            font.pointSize:18
            font.family: digital_font.name
            anchors.left: tire_pressure_front_right.left
            anchors.verticalCenter: parent.verticalCenter
            color:"white"
}
}

    Rectangle {
        id: tire_pressure_front_right_indicator
width:10
        height:10
        x:462
        y:650
        z:8
        color:"white"
        border.color:"white"
        border.width:1
        radius:5
}

    Image {
        id: tire_pressure_front_right_warning
        x:407
        y:630
        z:8
        scale:0.60
        opacity: low_presure_2_warning
        source:"images/tire_pressure_warning.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}
```

```
    Image {
        id: tire_pressure_back_panel
        x:1030
        y:630
        z:7
        scale:0.9
        opacity:1
        source:"images/tire_pressure.png"
}

    Rectangle {
        id: tire_pressure_back_left
width:120
        height:40
        x:905
        y:645
        z:7
        color:"#093375"
        border.color:"white"
        border.width:2
        radius:8
        Text {
            id: tire_pressure_back_left_value
            text: tire_presure_3 +" PSI "
            font.pointSize:18
            font.family: digital_font.name
            anchors.right: tire_pressure_back_left.right
            anchors.verticalCenter: parent.verticalCenter
            color:"white"
}
}

    Rectangle {
        id: tire_pressure_back_left_indicator
        width:10
        height:10
        x:910
        y:670
        z:8
        color:"white"
        border.color:"white"
        border.width:1
        radius:5
}

    Image {
        id: tire_pressure_back_left_warning
        x:904
        y:630
        z:8
        scale:0.60
        opacity: low_presure_3_warning
        source:"images/tire_pressure_warning.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}
```

```
    Rectangle {
        id: tire_pressure_back_right
        width:120
        height:40
        x:1111
        y:645
        z:7
        color:"#093375"
        border.color:"white"
        border.width:2
        radius:8
        Text {
            id: tire_pressure_back_right_value
            text:" "+ tire_presure_4 +" PSI "
            font.pointSize:18
            font.family: digital_font.name
            anchors.left: tire_pressure_back_right.left
            anchors.verticalCenter: parent.verticalCenter
            color:"white"
}
}

    Rectangle {
        id: tire_pressure_back_right_indicator
        width:10
        height:10
        x:1216
        y:670
        z:8
        color:"white"
        border.color:"white"
        border.width:1
        radius:5
}

    Image {
        id: tire_pressure_back_right_warning
        x:1161
        y:630
        z:8
        scale:0.60
        opacity: low_presure_4_warning
        source:"images/tire_pressure_warning.png"
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}

    Image {
        id: optical_horn_signal
        x:660
        y:170
        z:8
        scale:1.05
        opacity: optical_horn
        source:"images/high_beam.png"
```

```
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}
    Rectangle {
        id: efficiency_panel
        width:125
        height:38
        x:621
        y:545
        z:7
        color:"black"
        border.color:"white"
        border.width:2
        radius:8
        Text {
            id: efficiency_value
            text: efficiency +" KM/L"
            font.pointSize:20
            font.family: digital_font.name
            anchors.centerIn: efficiency_panel
            color:"white"
}
}
    Rectangle {
        id: odometer_panel
        width:160
        height:38
        x:604
        y:580
        z:7
        color:"black"
        border.color:"white"
        border.width:2
        radius:8
        Text {
            id: odometer_value
            text: odometer +" KM"
            font.pointSize:20
            font.family: digital_font.name
            anchors.centerIn: odometer_panel
            color:"white"
}
}
    Rectangle {
        id: black_panel
        width:1366
        height:768
        color:"black"
        z:8
        opacity: shutdown
        Behavior on opacity {
              NumberAnimation { duration:100}
}
}
}
/*********************************************************************/
```

10.6 Middleware Source Code

10.6.1 main.c

```
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include "menu_options.h"
#include "shared_memory.h"

int main(void)
{
	char menu_option  =' ';

	init_shared_memory();

	while(menu_option !='Q'){
		system("clear");

		display_main_menu();

		menu_option = getchar();

		change_option_value(menu_option);
	}
	deinit_shared_memory();
	return0;
}
/*****************************************************************************/
```

10.6.2 menu_options.c

```
/*****************************************************************************/
#include "menu_options.h"
void display_main_menu(void){
	printf("-------------------------------------------------------------
\n");
	printf("\t    Can Data Emulation - HMI Test Client \n\n");
	printf("-------------------------Main Menu----------------------
\n");
	printf("\t q) HIGH BEAM \n");
	printf("\t w) LOW BEAM \n");
	printf("\t e) OPTICAL HORN \n");
	printf("\t r) TURN RIGHT \n");
	printf("\t t) TURN LEFT \n");
	printf("\t y) HAZARD WARINING LIGHT \n");
	printf("\t u) ABS BREAK \n");
	printf("\t i) AIRBAG \n");
	printf("\t o) KEY STATUS \n");
	printf("\t p) LOW BATTERY \n");
	printf("\t a) LOW FUEL \n");
	printf("\t s) CHECK ENGINE \n");
	printf("\t d) SEATBELT \n");
	printf("\t f) HANDBRAKE \n");
	printf("\t g) SATTELITAL NOTIFICATION \n");
	printf("\t h) OIL \n");
	printf("\t j) DOOR WANING LIGHT \n");
	printf("\t k) MOTOR TEMPERATURE WARNING \n");
```

```
	printf("\t l) BATTERY \n");
	printf("\t z) ENV TEMP \n");
	printf("\t x) SPEED \n");
	printf("\t c) FUEL \n");
	printf("\t v) EFFICIENCY \n");
	printf("\t b) RPMs \n");
	printf("\t n) TIREPRESURE_1 \n");
	printf("\t m) LOW PRESURE 1 Warning \n");
	printf("\t 1) TIREPRESURE_2 \n");
	printf("\t 2) LOW PRESURE 2 Warning \n");
	printf("\t 3) TIREPRESURE_3 \n");
	printf("\t 4) LOW PRESURE 3 Warning \n");
	printf("\t 5) TIREPRESURE_4 \n");
	printf("\t 6) LOW PRESURE 4 Warning \n");
	printf("\t 7) ODOMETER\n");
	printf("\t 8) GEAR \n");
	printf("\t 9) COMPAS \n");
	printf("-----------------------------------------------\n");
	printf("\t Q) Exit \n");
	printf("-----------------------------------------------\n");
	printf("\t Select option to change:");
}

uint8_t display_boolean_sub_menu(void){

	uint8_t new_value;
	char option;

	printf("-----------------------------------------------\n");
	printf("\t    Boolean Options \n\n");
	printf("-------------------------Sub-Menu---------------\n");
	printf("\t 0) OFF \n");
	printf("\t 1) ON \n");
	printf("-----------------------------------------------\n");
	printf("\t Select value:");

	getchar();
	scanf("%c",&option);


	if(option =='0'){
		new_value = OFF;
	}else{
		new_value = ON;
	}
	return new_value;
}

float display_float_sub_menu(void){

	float new_value;

	printf("-----------------------------------------------\n");
	printf("\t      Float Options \n\n");
	printf("---------------------Sub-Menu------------------\n");
	printf("\t Before enter the new value, please check \n");
	printf("\t range values for each option. \n");
```

```
	printf("-----------------------------------------------------\n");
	printf("\t Enter new value:");

	getchar();
	scanf("%f",&new_value);

	return new_value;
}

int display_int_sub_menu(void){

	int new_value;

	printf("-----------------------------------------------------\n");
	printf("\t      Integer Options \n\n");
	printf("----------------------Sub-Menu---------------------\n");
	printf("\t Before enter the new value, please check \n");
	printf("\t range values for each option. \n");
	printf("-----------------------------------------------------\n");
	printf("\t Enter new value:");

	getchar();
	scanf("%i",&new_value);

	return new_value;
}

void change_option_value(char option){

	uint8_t     boolean_value;
	float       float_value;
	int         int_value;

	switch(option){
		case'q':
			boolean_value = display_boolean_sub_menu();
			getchar();
			write_uint8_value(option, boolean_value);
			break;
		case'w':
			boolean_value = display_boolean_sub_menu();
			getchar();
			write_uint8_value(option, boolean_value);
			break;
		case'e':
			boolean_value = display_boolean_sub_menu();
			getchar();
			write_uint8_value(option, boolean_value);
			break;
		case'r':
			boolean_value = display_boolean_sub_menu();
			getchar();
			write_uint8_value(option, boolean_value);
			break;
		case't':
			boolean_value = display_boolean_sub_menu();
			getchar();
```

```
            write_uint8_value(option, boolean_value);
            break;
        case'y':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case'u':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case'i':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case'o':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case'p':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case'a':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case's':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case'd':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case'f':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case'g':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
        case'h':
            boolean_value = display_boolean_sub_menu();
            getchar();
            write_uint8_value(option, boolean_value);
            break;
```

```
case'j':
      boolean_value = display_boolean_sub_menu();
      getchar();
      write_uint8_value(option, boolean_value);
      break;
case'k':
      boolean_value = display_boolean_sub_menu();
      getchar();
      write_uint8_value(option, boolean_value);
      break;
case'l':
      float_value = display_float_sub_menu();
      getchar();
      write_float_value(option, float_value);
      break;
case'z':
      float_value = display_float_sub_menu();
      getchar();
      write_float_value(option, float_value);
      break;
case'x':
      float_value = display_float_sub_menu();
      getchar();
      write_float_value(option, float_value);
      break;
case'c':
      float_value = display_float_sub_menu();
      getchar();
      write_float_value(option, float_value);
      break;
case'v':
      float_value = display_float_sub_menu();
      getchar();
      write_float_value(option, float_value);
      break;
case'b':
      int_value = display_int_sub_menu();
      getchar();
      write_int_value(option, int_value);
      break;
case'n':
      int_value = display_int_sub_menu();
      getchar();
      write_int_value(option, int_value);
      break;
case'm':
      boolean_value = display_boolean_sub_menu();
      getchar();
      write_uint8_value(option, boolean_value);
      break;
case'1':
      int_value = display_int_sub_menu();
      getchar();
      write_int_value(option, int_value);
      break;
case'2':
      boolean_value = display_boolean_sub_menu();
```

```
                getchar();
                write_uint8_value(option, boolean_value);
                break;
            case'3':
                int_value = display_int_sub_menu();
                getchar();
                write_int_value(option, int_value);
                break;
            case'4':
                boolean_value = display_boolean_sub_menu();
                getchar();
                write_uint8_value(option, boolean_value);
                break;
            case'5':
                int_value = display_int_sub_menu();
                getchar();
                write_int_value(option, int_value);
                break;
            case'6':
                boolean_value = display_boolean_sub_menu();
                getchar();
                write_uint8_value(option, boolean_value);
                break;
            case'7':
                int_value = display_int_sub_menu();
                getchar();
                write_int_value(option, int_value);
                break;
            case'8':
                int_value = display_int_sub_menu();
                getchar();
                write_int_value(option, int_value);
                break;
            case'9':
                int_value = display_int_sub_menu();
                getchar();
                write_int_value(option, int_value);
                break;
            default:
                break;
        }
}
/*****************************************************************************/
```

10.6.3 menu_options.h

```
/*****************************************************************************/
#ifndef MENU_OPTIONS_H_
#define MENU_OPTIONS_H_

#include <stdio.h>
#include <stdint.h>
#include "types_definitions.h"
#include "shared_memory.h"


void display_main_menu(void);
uint8_t display_boolean_sub_menu(void);
```

```
float display_float_sub_menu(void);
int display_int_sub_menu(void);
void change_option_value(char option);

#endif /* MENU_OPTIONS_H_ */
/*****************************************************************************/
```

10.6.4 shared_memory.c

```
/*****************************************************************************/
#include "shared_memory.h"

tshared_memory     can_data;
tshared_memory * ptr_can_data;

/* Data types for Semaphore and Shared Memory*/
key_t Key;
int Id_Shared_Memory;
int Id_Semaphore;
struct sembuf Operation;

#define FILEKEY         "/bin/ls"
#define KEY             10
#define BYTES           sizeof(can_data)
#define SEMAPHORES      1

void init_shared_memory(void){

      /* Calculate Key for Shared Memory and Semaphore */
      Key = ftok(FILEKEY, KEY);
      /* Check if error */
      if(Key ==-1)
      {
            printf ("Error with Key for Shared Memory and Semaphore \n");
      return;
      }

      /* Create the Semaphore */
      Id_Semaphore = semget (Key, SEMAPHORES,0777| IPC_CREAT);
      /* Check if error */
      if(Id_Semaphore ==-1)
      {
            printf ("Error with Shared Memory ID \n");
                  return;
      }

      /* Create the Shared Memory */
      Id_Shared_Memory = shmget (Key, BYTES,0777| IPC_CREAT);
      /* Check if error */
      if(Id_Shared_Memory ==-1)
      {
            printf ("Error with Shared Memory ID \n");
      return;
      }

      /* Semaphore Initialization*/
      semctl(Id_Semaphore,0, SETVAL,1);
```

```
	/* Point to Shared Memory */
	ptr_can_data =(tshared_memory *)shmat
(Id_Shared_Memory,(char*)0,0);
	/* Check if error */
	if(ptr_can_data ==NULL)
	{
		printf ("Error reserving shared memory \n");
	return;
	}

	/* Set values for Semaphore */
	Operation.sem_num =0;
	Operation.sem_flg =0;
	Operation.sem_op =1;
}


void deinit_shared_memory(void){
	/* Free the shared memory */
	shmdt ((char*)ptr_can_data);
	shmctl (Id_Shared_Memory, IPC_RMID,(struct shmid_ds *)NULL);
	return;
}

void write_uint8_value(char option,uint8_t value){
	switch(option){
			case'q':
				/* Change Semaphore to RED */
				Operation.sem_op =-1;
				/* Set the Semaphore to RED */
				semop (Id_Semaphore,&Operation,1);
				/* Write Value */
				ptr_can_data->high_beam = value;
				/* Change Semaphore to GREEN */
				Operation.sem_op =1;
				/* Set Semaphore to GREEN */
				semop (Id_Semaphore,&Operation,1);
				break;
			case'w':
				/* Change Semaphore to RED */
				Operation.sem_op =-1;
				/* Set the Semaphore to RED */
				semop (Id_Semaphore,&Operation,1);
				/* Write Value */
				ptr_can_data->low_beam = value;
				/* Change Semaphore to GREEN */
				Operation.sem_op =1;
				/* Set Semaphore to GREEN */
				semop (Id_Semaphore,&Operation,1);
				break;
			case'e':
				/* Change Semaphore to RED */
				Operation.sem_op =-1;
				/* Set the Semaphore to RED */
				semop (Id_Semaphore,&Operation,1);
				/* Write Value */
				ptr_can_data->optical_horn = value;
```

```
            /* Change Semaphore to GREEN */
            Operation.sem_op =1;
            /* Set Semaphore to GREEN */
            semop (Id_Semaphore,&Operation,1);
            break;
      case'r':
            /* Change Semaphore to RED */
            Operation.sem_op =-1;
            /* Set the Semaphore to RED */
            semop (Id_Semaphore,&Operation,1);
            /* Write Value */
            ptr_can_data->turn_right = value;
            /* Change Semaphore to GREEN */
            Operation.sem_op =1;
            /* Set Semaphore to GREEN */
            semop (Id_Semaphore,&Operation,1);
            break;
      case't':
            /* Change Semaphore to RED */
            Operation.sem_op =-1;
            /* Set the Semaphore to RED */
            semop (Id_Semaphore,&Operation,1);
            /* Write Value */
            ptr_can_data->turn_left = value;
            /* Change Semaphore to GREEN */
            Operation.sem_op =1;
            /* Set Semaphore to GREEN */
            semop (Id_Semaphore,&Operation,1);
            break;
      case'y':
            /* Change Semaphore to RED */
            Operation.sem_op =-1;
            /* Set the Semaphore to RED */
            semop (Id_Semaphore,&Operation,1);
            /* Write Value */
            ptr_can_data->hazard_warning_light = value;
            /* Change Semaphore to GREEN */
            Operation.sem_op =1;
            /* Set Semaphore to GREEN */
            semop (Id_Semaphore,&Operation,1);
            break;
      case'u':
            /* Change Semaphore to RED */
            Operation.sem_op =-1;
            /* Set the Semaphore to RED */
            semop (Id_Semaphore,&Operation,1);
            /* Write Value */
            ptr_can_data->abs_break = value;
            /* Change Semaphore to GREEN */
            Operation.sem_op =1;
            /* Set Semaphore to GREEN */
            semop (Id_Semaphore,&Operation,1);
            break;
      case'i':
            /* Change Semaphore to RED */
            Operation.sem_op =-1;
            /* Set the Semaphore to RED */
```

```
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->airbag = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'o':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /*Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->key_status = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'p':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->low_battery = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'a':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->low_fuel = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case's':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->check_engine = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'd':
```

```
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->seat_belt = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'f':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->hand_break = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'g':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->satelital_notification = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'h':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->oil = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'j':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->door_warning_light = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
```

```
        semop (Id_Semaphore,&Operation,1);
        break;
case'k':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->motor_temperature_warning = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'm':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->low_presure_1_warning = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'2':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->low_presure_2_warning = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'4':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->low_presure_3_warning = value;
        /* Change Semaphore to GREEN */
        Operation.sem_op =1;
        /* Set Semaphore to GREEN */
        semop (Id_Semaphore,&Operation,1);
        break;
case'6':
        /* Change Semaphore to RED */
        Operation.sem_op =-1;
        /* Set the Semaphore to RED */
        semop (Id_Semaphore,&Operation,1);
        /* Write Value */
        ptr_can_data->low_presure_4_warning = value;
```

```
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                default:
                        break;
            }
}

void write_int_value(char option,int value){
      switch(option){
                case'b':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->rpms =(uint16_t)value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                case'n':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->tire_presure_1 =(uint8_t)value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                case'1':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->tire_presure_2 =(uint8_t)value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                case'3':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->tire_presure_3 =(uint8_t)value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
```

```
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  case'5':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->tire_presure_4 =(uint8_t)value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  case'7':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->odometer =(uint32_t)value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  case'8':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->gear =(uint8_t)value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  case'9':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->compas =(uint8_t)value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  default:
                        break;
            }
}
```

```
void write_float_value(char option,float value){
      switch(option){
                  case'l':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->battery = value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  case'z':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->env_temp = value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  case'x':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->speed = value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  case'c':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->fuel = value;
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  case'v':
                        /* Change Semaphore to RED */
                        Operation.sem_op =-1;
                        /* Set the Semaphore to RED */
                        semop (Id_Semaphore,&Operation,1);
                        /* Write Value */
                        ptr_can_data->efficiency = value;
```

```
                        /* Change Semaphore to GREEN */
                        Operation.sem_op =1;
                        /* Set Semaphore to GREEN */
                        semop (Id_Semaphore,&Operation,1);
                        break;
                  default:
                        break;
            }
}
/*****************************************************************************/
```

10.6.5 shared_memory.h

```
/*****************************************************************************/
#ifndef SHARED_MEMORY_H_
#define SHARED_MEMORY_H_
#include <stdio.h>
#include <stdint.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/sem.h>
#include "types_definitions.h"
void init_shared_memory(void);
void deinit_shared_memory(void);
void write_uint8_value(char option,uint8_t value);
void write_int_value(char option,int value);
void write_float_value(char option,float value);
#endif /* SHARED_MEMORY_H_ */
/*****************************************************************************/
```

10.6.6 types_definitions.h

```
/*****************************************************************************/
#ifndef TYPES_DEFINITIONS_H_
#define TYPES_DEFINITIONS_H_

#define ON  1
#define OFF 0

typedefstruct{
      float       battery;
      float       env_temp;
      float       speed;
      float       fuel;
      float       efficiency;
      uint32_t    odometer;
      uint16_t    rpms;
      uint8_t           tire_presure_1;
      uint8_t           low_presure_1_warning;
      uint8_t     tire_presure_2;
      uint8_t     low_presure_2_warning;
      uint8_t           tire_presure_3;
      uint8_t           low_presure_3_warning;
      uint8_t     tire_presure_4;
      uint8_t     low_presure_4_warning;
      uint8_t     gear;
      uint8_t     compas;
```

```
        uint8_t     high_beam;
        uint8_t     low_beam;
        uint8_t     optical_horn;
        uint8_t     turn_right;
        uint8_t     turn_left;
        uint8_t     hazard_warning_light;
        uint8_t     abs_break;
        uint8_t     airbag;
        uint8_t     key_status;
        uint8_t     low_battery;
        uint8_t     low_fuel;
        uint8_t     check_engine;
        uint8_t     seat_belt;
        uint8_t     hand_break;
        uint8_t     satelital_notification;
        uint8_t     oil;
        uint8_t     door_warning_light;
        uint8_t     motor_temperature_warning;
} tshared_memory;

#endif /* TYPES_DEFINITIONS_H_ */
/*****************************************************************************/
```

11 Bibliography

[1] A. SIG, "Automotive SPICE® Process Reference Model," 2010.

[2] Buildroot, "The Buildroot user manual," Buildroot, 2015. [Online]. Available: http://buildroot.uclibc.org/downloads/manual/manual.html. [Accessed 2015].

[3] T. Q. Company, "Qt Documentation," The Qt Company, 2015. [Online]. Available: http://doc.qt.io/qt-4.8/index.html. [Accessed 2015].

[4] T. Q. Company, "Qt Documentation," The Qt Company, 2015. [Online]. Available: http://doc.qt.io/qt-5/index.html. [Accessed 2015].

[5] G. Alliance, "GENIVI Diagnostic Log and Trace," GENIVI, 2016. [Online]. Available: http://projects.genivi.org/diagnostic-log-trace/. [Accessed 2016].

[6] M. Bats, "mbats," GitHub, 10 November 2011. [Online]. Available: https://github.com/mbats/eclipse-buildroot-bundle/wiki. [Accessed 2015].