Title: Electronic instrument cluster
SW Component Cluster_EA V1.5

History				
Issue status (Index)	**Maturity/Date** (draft/invalid/valid) (dd-mmm-yyyy)	**Author** Department	**Check/Release** Department	**Description**
1.0	Draft 08-Jan-2016	Edgar Escayola	Adrián Zacarías	Creation of the document
1.1	Draft 11-Jan-2016	Adrián Zacarías	Edgar Escayola	Addition of the architecture and deployment diagram.
1.2	Draft 12-Jan-2016	Edgar Escayola	Adrián Zacarías	Addition of the functional decomposition diagram and names of the functions.
1.3	Draft 14-Jan-2016	Edgar Escayola	Adrián Zacarías	Addition of the definition of each of the functions.
1.4	Draft 14-Jan-2016	Adrián Zacarías	Edgar Escayola	Addition of the definition of the functions from the application.
1.5	Draft 15-Jan-2016	Adrián Zacarías	Edgar Escayola	Traceability updated.

Table of Contents

1 Purpose

This document has been created to describe the design specifications of the application Cluster_EA. It consists on the implementation of the CAN protocol in the TRK-MPC5606B development board and an application, which is a cluster with the following characteristics: speedometer, fuel gauge, indicators LEDs, and odometer.

Req. Id. 1.1

2 References

N°	Document name	Reference	Revision
1	Traceability Matrix – Cluster_EA	Final_Project/Documents/Requirements/Traceability Matrix – Cluster_EA.xls	1.1
2	MPC5607B Microcontroller Reference Manual	Final_Project/Documents/MPC5607BRM_Reference_Manual.pdf	7.2
3	Quick Start Guide TRK-MPC5606B	Final_Project/Documents/Quick_Start_Guide.pdf	3
5	ID_Cluster_Requirements	Final_Project/Documents/ID_Cluster_Requirements.docx	1.0

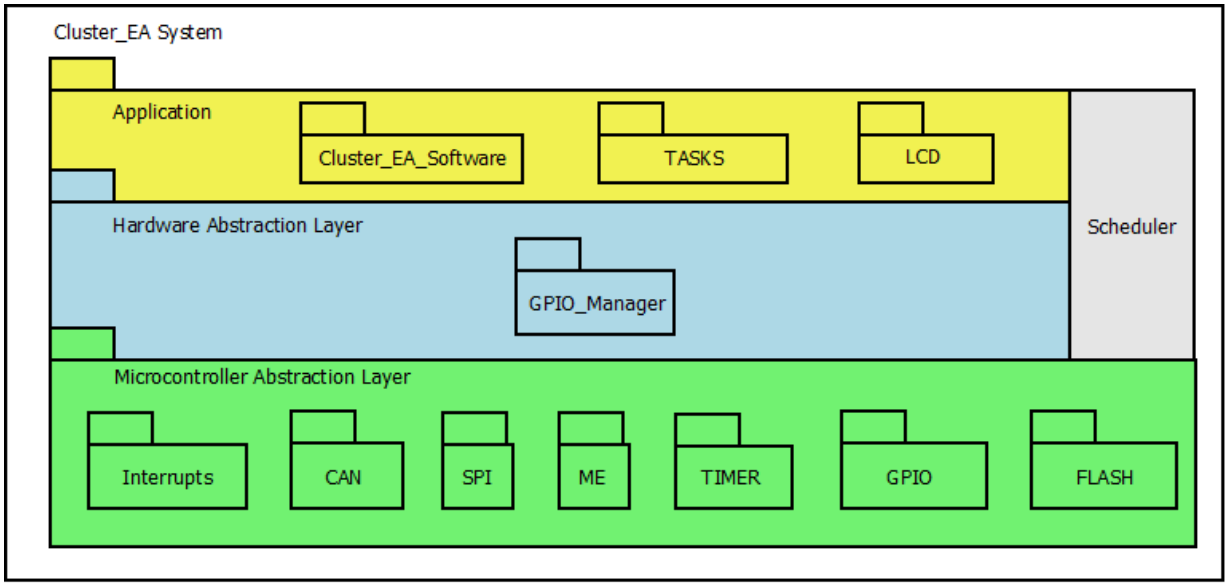
3 Realization constraints and targets

3.1 TRK-MPC5606B's features

- MPC5606B MCU (144-pin LQFP).
- On-board JTAG connection via open source OSBDM circuit using the MPC9S08JM MCU.
- MCZ3390S5EK system basis chip with advanced power management and integrated CAN transceiver and LIN 2.0 interface.
- CAN interface.
- LIN interface with 1.3, 2.0, 2.1, and J2602 protocol versions supported.
- Analog interface with potentiometer.
- High-efficency green LEDs.
- 4 PushButtons.
- Serial communication interface.
- External power 9V DC to 12V DC regulated down to 5V DC.

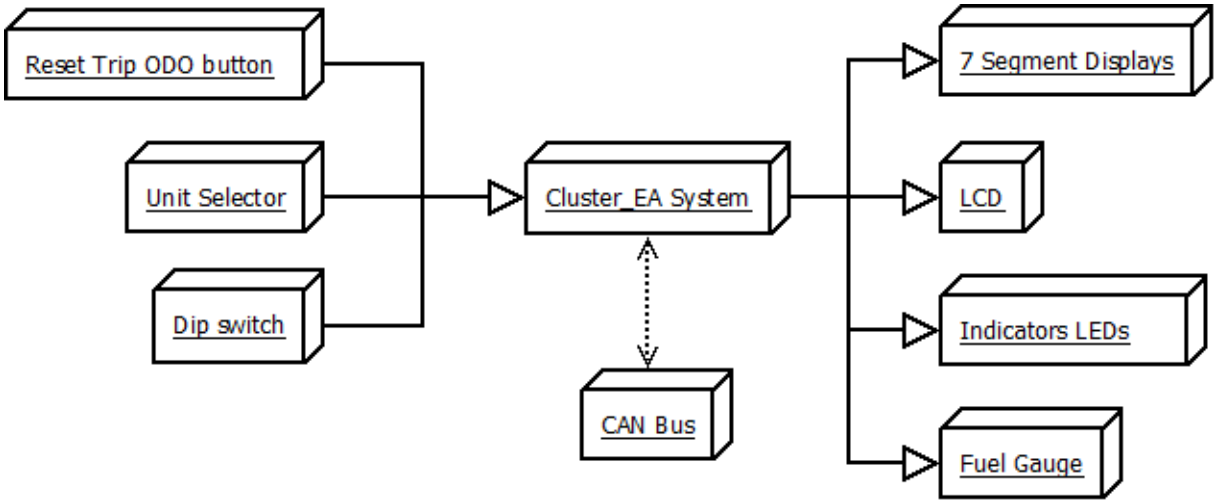
4 SW Conceptual design

4.1 Architecture design



The architecture diagram is divided in 3 layers: Application abstraction layer, hardware abstraction layer, and microcontroller abstraction layer. The following modules are positioned at the microcontroller abstraction layer: Interrupts, CAN, SPI, ME, TIMER, GPIO, and FLASH. Each of them handles the registers needed for its functions. The next layer is the hardware abstraction layer, which only has the GPIO_Manager module. This module handles the hardware used without affecting the registers, only using the modules from the lower layer. The application layer consists on the following modules: Cluster_EA_Software, TASKS, and LCD. This modules use the only module in the hardware abstraction layer in order to manage all the functions of the system. Parallel to the application and hardware layer, there is the scheduler, which interact with modules from each of the layers.

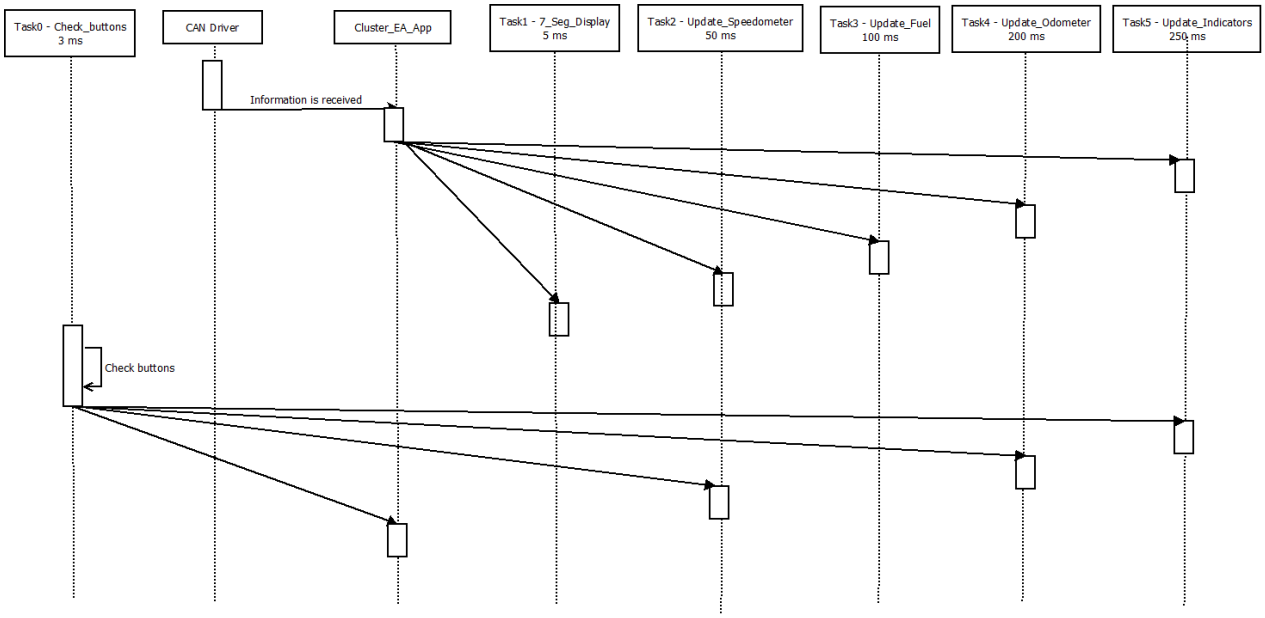
4.2 Deployment Diagram



The system has three inputs, four outputs, and a communication bus. The inputs are the following: a DIP switch to select the ignition and battery status, a button to reset the trip odometer, and another button for selecting the unit of the speedometer between miles or kilometers per hour. The outputs are the following: three seven segments display for displaying the speed, an LCD for displaying the odometer and trip odometer, indicators LEDs and a LED bar for the fuel gauge. The indicators are five LEDs for the emergency break, high beams, fuel reserve, unbuckled seat belt and opened doors.

Req. Id. 2.2, 2.3, 2.7, 2.9, 2.14, 3.2, 4.1, 4.10, 4.12

4.3 Sequence diagram

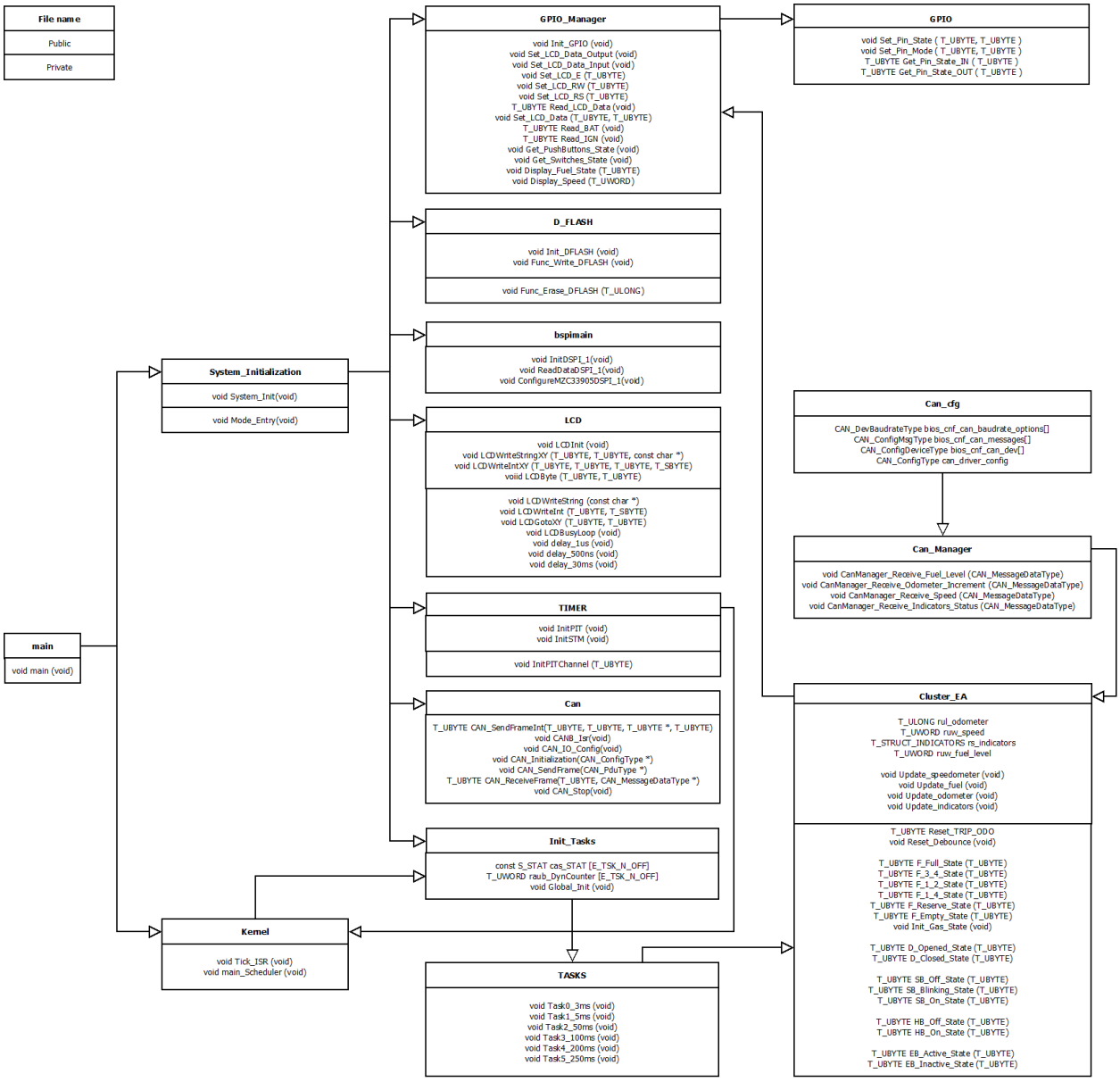


This sequence diagram illustrates the functionality of the periodic tasks of the system. Whenever a valid CAN ID with valid information is received, the corresponding variables are updated, then the tasks gets affected by those variables. A three milliseconds task which checks the status of the buttons. In case that there is a valid button press, the following tasks are affected the next time they will run: Update speedometer, update odometer or update indicators.

Req. Id.2.5, 2.6, 2.10

5 SW Component internal breakdown

5.1 Functional Decomposition



File	Description
Main	Main module that runs the main function. It is divided in system's initialization and execution of the scheduler.
System_Initialization	The function of this module is to call the functions that initializes the mode of operation, peripherals, and the scheduler.
Cluster_EA	It is in charge of executing the main application of the program. It contains the functions that update the speed, fuel gauge, odometer, and indicators.
Kernel	The scheduler is being executed here. It handles the main configurations and the tick interrupt.
TASKS	This module contains the periodic tasks that are executed by the scheduler.
Init_Tasks	This module contains the global initializations which are needed for the correct execution of the scheduler.
CAN	This module contains the CAN driver. It consists of an initialization of the CAN controller and an interrupt for reception.
TIMER	This file contains the configurations that must be done to achieve the periodic interrupt that gives the Ticks to the scheduler.
GPIO	This module handles the registers needed to configure ports and change state of pins.
GPIO_Manager	This module configures and handles the ports needed by the application.
D_FLASH	This module contains the driver needed to use the flash memory.
Bspimain	This module is in charge of configuring the transceiver needed for CAN communication.
LCD	The main function of this module is to control the Hitachi LCD HD44780 in 4 bit mode.
CAN_Manager	This module contains the callback functions of the IDs received through CAN.
Can_cfg	The configuration of the CAN driver is defined in this module. The IDs and callback functions are linked here.

Function Description and Dynamic Behavior

5.2 TASKS

5.2.1 Function void Task0_3ms (void)

Description	This task is executed every 3 milliseconds. Its main function is to check the state of the push buttons.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.2.2 Function void Task1_5ms (void)

Description	This task is executed every 5 milliseconds. Its main function is to refresh the 7 segments displays.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.2.3 Function void Task2_50ms (void)

Description	This task is executed every 50 milliseconds. Its main function is to update the speedometer.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

Req. Id. 2.5

5.2.4 Function void Task3_100ms (void)

Description	This task is executed every 100 milliseconds. Its main function is to update the fuel level.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.2.5 Function void Task4_200ms (void)

Description	This task is executed every 200 milliseconds. Its main function is to update the value of the odometers.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

Req. Id. 2.10

5.2.6 Function void Task5_250ms (void)

Description	This task is executed every 250 milliseconds. Its main function is to update the state of the indicators.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.3 Init_Tasks

5.3.1 Function void Global_Init (void)

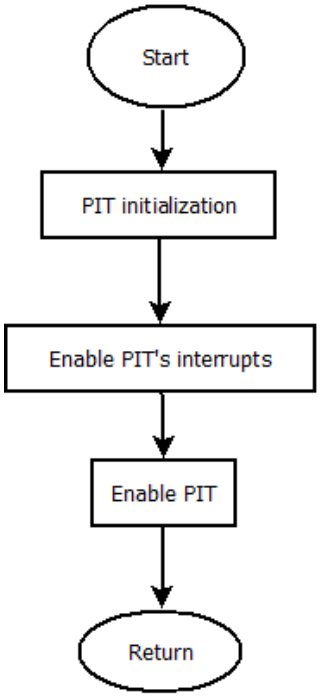
Description	This function initializes the scheduler.
Parameter 1	Void
Return Value	Void
Precondition	This function should be called before executing the scheduler.
Post condition	The scheduler's functionalities can be used.
Error Conditions	Does not apply.

5.4 TIMER

5.4.1 Function void InitPIT (void)

Description	PIT is initialized.
Parameter 1	Void
Return Value	Void
Precondition	This function is called in the beginning of the main program to initialize the PIT.
Post condition	The interrupts every 5ms will be generated.
Error Conditions	Does not apply

Dynamic Behavior
Activity diagram



5.4.2 Function void InitPITChannel (T_UBYTE)

Description	This function configures the given channel of the PIT timer.
Parameter 1	T_UBYTE PIT channel which must be configured.
Parameter 2..n	Does not apply
Return Value	Void
Precondition	This function should be called in the beginning of the main application.
Post condition	The initialized channel of the PIT will be ready to use.
Error Conditions	Does not apply.

5.4.3 Function void InitSTM (void)

Description	STM is initialized.
Parameter 1	Void
Return Value	Void
Precondition	This function is called in the beginning of the main program to initialize the STM.
Post condition	The STM counter will be incrementing its value at all times.
Error Conditions	Does not apply

5.4.4 Function void Clear_STM (void)

Description	The STM counter is cleared.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	The STM counter will be equal to 0.
Error Conditions	Does not apply

5.5 GPIO_Manager

5.5.1 Function void Init_GPIO (void)

Description	This function initializes all the pins that are going to be used in the application.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply
Post condition	The LED will be able to be changed.
Error Conditions	Does not apply

5.5.2 Function void Set_LCD_Data_Output (void)

Description	This function sets as outputs all the data pins of the LCD.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.5.3 Function void Set_LCD_Data_Input (void)

Description	This function sets as inputs all the data pins of the LCD.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.5.4 Function void Set_LCD_E (T_UBYTE)

Description	This function turns on or off the enable pin of the LCD depending on the parameter 1.
Parameter 1	Receives the new state of the enable pin of the LCD.
Return Value	Void
Precondition	The enable pin shall be configured as an output.
Post condition	The enable pin takes the state given by the parameter.
Error Conditions	Does not apply

5.5.5 Function void Set_LCD_RW (T_UBYTE)

Description	This function turns on or off the read/write pin of the LCD depending on the parameter.
Parameter 1	Receives the new state of the read/write pin of the LCD.
Return Value	Void
Precondition	The read/write pin shall be configured as an output.
Post condition	The read/write pin takes the state given by the parameter.
Error Conditions	Does not apply

5.5.6 Function void Set_LCD_RS (T_UBYTE)

Description	This function turns on or off the register select pin of the LCD depending on the parameter.
Parameter 1	Receives the new state of the register select pin of the LCD.
Return Value	Void
Precondition	The register select pin shall be configured as an output.
Post condition	The register select pin takes the state given by the parameter.
Error Conditions	Does not apply

5.5.7 Function T_UBYTE Read_LCD_Data (void)

Description	This function reads the data pins of the LCD.
Parameter 1	Void
Return Value	Returns a T_UBYTE with the value of the data pins.
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.5.8 Function void Set_LCD_Data (T_UBYTE, T_UBYTE)

Description	This function sets a value to a data bit.
Parameter 1	Receives a value from 1 to 4 to select the data bit that shall be changed.
Parameter 2	Receives the value that shall take the selected data bit.
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.5.9 Function T_UBYTE Read_BAT (void)

Description	This function reads the value of the battery switch.
Parameter 1	Void
Return Value	Returns the value of the battery switch.
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.5.10 Function T_UBYTE Read_IGN (void)

Description	This function reads the value of the ignition switch.
Parameter 1	Void
Return Value	Returns the value of the ignition switch.
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.5.11 Function void Get_PushButtons_State (void)

Description	This function gets the state of the push buttons and saves it into global variables for processing.
Parameter 1	Void
Return Value	Void
Precondition	The push buttons shall be configured as inputs.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.5.12 Function void Get_Switches_State (void)

Description	This function gets the state of the switches and saves it into global variables for processing.
Parameter 1	Void
Return Value	Void
Precondition	The switches shall be configured as inputs.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.5.13 Function void Set_Bar_Led (T_UBYTE)

Description	This function sets the fuel level value to a led bar.
Parameter 1	Receives a value to set the led bar level.
Return Value	Void
Precondition	The led bar pins shall be configured as outputs.
Post condition	Does not apply.
Error Conditions	Does not apply

5.5.14 Function void Display_Speed (T_UWORD)

Description	This function shows the speed in 7 segment displays.
Parameter 1	Receives a variable with the speed.
Return Value	Void
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.6 GPIO

5.6.1 Function void Set_Pin_State (T_UBYTE, T_UBYTE)

Description	Changes the logic level of the output pin selected.
Parameter 1	T_UBYTE. Corresponds to the pin number that should be affected.
Parameter 2	T_UBYTE. It can receive either 0 or 1 to turn the pin off or on respectively.
Return Value	Void
Precondition	The mode of the pin selected should be OUTPUT.
Post condition	The logic level for the selected pin, will be the one selected in the second parameter.
Error Conditions	Does not apply

5.6.2 Function void Set_Pin_Mode (T_UBYTE, T_UBYTE)

Description	This function changes the pin mode of the selected pin to the selected mode.
Parameter 1	T_UBYTE. Corresponds to the pin number that should be affected.
Parameter 2	T_UBYTE. It can receive a value according to the following definitions: 0 -> OUTPUT, 1 -> INPUT, 2 -> LIN_TX, 3 -> LIN_RX.
Return Value	Void
Precondition	Does not apply
Post condition	The selected pin will act as the selected pin mode.
Error Conditions	Does not apply

5.6.3 Function T_UBYTE Get_Pin_State_IN (T_UBYTE)

Description	This function returns the state of the given pin previously configured as an input.
Parameter 1	T_UBYTE. Corresponds to the pin number which state is unknown.
Return Value	T_UBYTE. Corresponds to the logic state of the pin. It can be either 0 or 1, meaning off or on, respectively.
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.6.4 Function T_UBYTE Get_Pin_State_OUT (T_UBYTE)

Description	This function returns the state of the given pin previously configured as an output.
Parameter 1	T_UBYTE. Corresponds to the pin number which state is unknown.
Return Value	T_UBYTE. Corresponds to the logic state of the pin. It can be either 0 or 1, meaning off or on, respectively.
Precondition	Does not apply
Post condition	Does not apply
Error Conditions	Does not apply

5.7 KERNEL

5.7.1 Function void Tick_ISR (void)

Description	This function is the one that handles the clock Ticks in order to trigger the tasks. This interrupt runs periodically every 5 milliseconds according to the configuration of the PIT.
Parameter 1	Void
Return Value	Void
Precondition	This function is called when a PIT interrupt is generated.
Post condition	It will literally interrupt the flow of the program to implement its code.
Error Conditions	Does not apply

5.7.2 Function void main_Scheduler (void)

Description	This function contains the main function of the scheduler which controls the timing for each of the tasks.
Parameter 1	Void
Return Value	Void
Precondition	The Global_Init function should be executed before.
Post condition	Does not apply
Error Conditions	Does not apply

5.8 Main

5.8.1 Function void main (void)

Description	This function runs first by default. It is divided in system initialization and execution of the scheduler.
Parameter 1	Void
Parameter 2	Does not apply.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.9 System_Initialization

5.9.1 Function void System_Init (void)

Description	This function calls the functions that initializes the mode of operation, peripherals, and the scheduler.
Parameter 1	Void
Parameter 2	Does not apply.
Return Value	Void
Precondition	This should be the first function called in the main program.
Post condition	It will be possible to use the microcontroller with the configurations done.
Error Conditions	Does not apply.

5.9.2 Function void ModeEntry (void)

Description	It initializes the mode of operation.
Parameter 1	Void
Parameter 2	Does not apply.
Return Value	Void
Precondition	This should be the first function called in the main program.
Post condition	It will be possible to use the microcontroller with the configurations done.
Error Conditions	Does not apply.

5.10 Cluster_EA

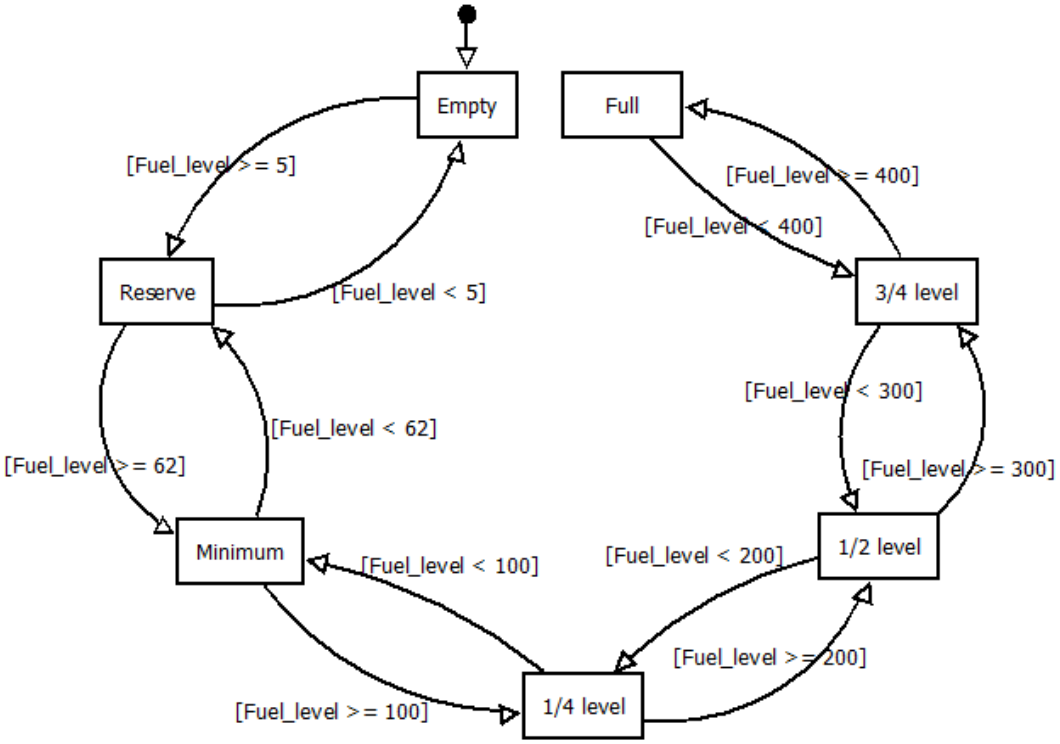
5.10.1 Function void Update_speedometer (void)

Description	Function that updates the speedometer.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.10.2 Function void Update_fuel (void)

Description	Function that updates the fuel level. It holds the state machine for the fuel level.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

State diagram



Req. Id. 3.3

5.10.3 Function void Update_odometer (void)

Description	Function that updates the value of the odometer.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

Req. Id. 2.11, 2.12, 2.13, 2.15, 2.16, 2.17, 2.18

5.10.4 Function void Update_indicators (void)

Description	Function that updates the state of the indicators.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

Req. Id. 4.2, 4.4, 4.5, 4.6, 4.8

5.10.5 Function T_UBYTE F_Full_State (T_UBYTE lub_Data)

Description	Function that
Parameter 1	Receives the state variable.
Return Value	Returns the state variable.
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.10.6 Function T_UBYTE F_3_4_State (T_UBYTE)

Description	Fuel gauge three quarters state.
Parameter 1	Receives the state variable.
Return Value	Returns the state variable.
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.10.7 Function T_UBYTE F_1_2_State (T_UBYTE)

Description	Fuel gauge half state.
Parameter 1	Receives the state variable.
Return Value	Returns the state variable.
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.10.8 Function T_UBYTE F_1_4_State (T_UBYTE)

Description	Fuel gauge one quarter state.
Parameter 1	Receives the state variable.
Return Value	Returns the state variable.
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.10.9 Function T_UBYTE F_Reserve_State (T_UBYTE)

Description	Fuel gauge reserve state.
Parameter 1	Receives the state variable.
Return Value	Returns the state variable.
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.10.10 Function T_UBYTE F_Empty_State (T_UBYTE)

Description	Fuel gauge empty state.
Parameter 1	Receives the state variable.
Return Value	Returns the state variable.
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.10.11 Function T_UBYTE F_Minimum_State(T_UBYTE)

Description	Fuel gauge minimum state.
Parameter 1	Receives the state variable.
Return Value	Returns the state variable.
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.10.12 Function void Init_Gas_State (void)

Description	Function that handles the 4 seconds initialization of the fuel gauge.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

Req. Id. 3.4, 3.5

5.11 CAN

5.11.1 Function T_UBYTE CAN_SendFrameInt (T_UBYTE, T_UBYTE, T_UBYTE *, T_UBYTE)

Description	Function that sends a message through CAN.
Parameter 1	Receives a structure with the following information: message buffer number, data length code, and an array with data.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.11.2 Function void CANB_Isr (void)

Description	Function that handles all the interrupts generated for CAN, including the reception of information.
Parameter 1	Void
Return Value	Void
Precondition	It shall be configured in the initialization as the interrupt handler for CAN.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.11.3 Function void CAN_IO_Config (void)

Description	Function that configures the pins used by CAN.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.11.4 Function void CAN_Initialization (CAN_ConfigType *)

Description	Function that initializes the CAN driver.
Parameter 1	Receives a structure with the CAN configuration.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.11.5 Function void CAN_SendFrame (CAN_PduType *)

Description	Function that sends a message through CAN.
Parameter 1	Receives a structure with the following information: message buffer number, data length code, and an array with data.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.11.6 Function T_UBYTE CAN_ReceiveFrame (T_UBYTE, CAN_MessageDataType *)

Description	Function that receives a message through CAN.
Parameter 1	Receives the message buffer number
Parameter 2	Receives a structure with information of the message data type.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.11.7 Function void CAN_Stop (void)

Description	Function that stops the CAN driver.
Parameter 1	Does not apply.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.12 CAN_Manager

5.12.1 Function void CanManager_Receive_Fuel_Level (CAN_MessageDataType)

Description	Callback function for the 0x103 ID that handles the information received through CAN.
Parameter 1	Receives a structure with the following information: message buffer number, data length code, and an array with data.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

Req. Id. 5.3, 5.6

5.12.2 Function void CanManager_Receive_Odometer_Increment (CAN_MessageDataType)

Description	Callback function for the 0x102 ID that handles the information received through CAN.
Parameter 1	Receives a structure with the following information: message buffer number, data length code, and an array with data.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

Req. Id. 5.2

5.12.3 Function void CanManager_Receive_Speed (CAN_MessageDataType)

Description	Callback function for the 0x101 ID that handles the information received through CAN.
Parameter 1	Receives a structure with the following information: message buffer number, data length code, and an array with data.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

Req. Id. 2.4, 5.1, 5.5

5.12.4 Function void CanManager_Receive_Indicators_Status (CAN_MessageDataType)

Description	Callback function for the 0x104 ID that handles the information received through CAN.
Parameter 1	Receives a structure with the following information: message buffer number, data length code, and an array with data.
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

Req. Id. 5.4, 5.7

5.13 D_FLASH

5.13.1 Function void Init_DFLASH (void)

Description	Function that initializes the FLASH module.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	The flash module can be used after the initialization.
Error Conditions	Does not apply.

5.13.2 Function void Func_Write_DFLASH (void)

Description	Function that writes information in a defined address of the flash.
Parameter 1	Void
Return Value	Void
Precondition	The flash module had to be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.13.3 Function void Func_Erase_DFLASH (T_ULONG)

Description	Function that erases a defined block of address of the flash.
Parameter 1	Void
Return Value	Void
Precondition	The flash module had to be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.14 Bspimain

5.14.1 Function void InitDSPI_1 (void)

Description	Function that initializes the SPI module.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.14.2 Function void ReadDataDSPI_1 (void)

Description	Function that reads the received data through SPI.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.14.3 Function void ConfigureMZC33905DSPI_1 (void)

Description	Function that configures the transceiver MZC33905 through SPI.
Parameter 1	Void
Return Value	Void
Precondition	The SPI module must be initialized.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15 LCD

5.15.1 Function void LCDInit (void)

Description	Function that initializes the LCD module. This must be called before calling LCD related functions.
Parameter 1	Void
Return Value	Void
Precondition	Does not apply.
Post condition	After this function the LCD can be used.
Error Conditions	Does not apply.

5.15.2 Function void LCDWriteStringXY (T_UBYTE, T_UBYTE, const char *)

Description	Function that receives the position and the string to be printed.
Parameter 1	Receives the x value for the position.
Parameter 2	Receives the y value for the position.
Parameter 3	Receives an array of characters to be printed in the LCD.
Return Value	Void
Precondition	The LCD must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15.3 Function void LCDWriteIntXY (T_UBYTE, T_UBYTE, T_UWORD, T_SBYTE)

Description	Function that receives the position and value to be printed.
Parameter 1	Receives the x value for the position.
Parameter 2	Receives the y value for the position.
Parameter 3	Receives the variable that is going to be printed.
Parameter 4	Receives the number of digits of the variable. If a -1 is sent, it calculates the length.
Return Value	Void
Precondition	The LCD must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15.4 Function void LCDByte (T_UBYTE, T_UBYTE)

Description	Function to send a byte to the LCD in 4bit mode.
Parameter 1	Receives the message to be sent.
Parameter 2	Selects if it is a command or a message (0 or 1 respectively).
Return Value	Void
Precondition	The LCD must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15.5 Function void LCDWriteString (const char *)

Description	Function that receives a string to print it in the LCD.
Parameter 1	Receives an array of characters to be printed in the LCD.
Return Value	Void
Precondition	The LCD must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15.6 Function void LCDWriteInt (T_ULONG, T_SBYTE)

Description	Function that receives a number and a field length to print it in the LCD.
Parameter 1	Receives the variable that is going to be printed.
Parameter 2	Receives the number of digits of the variable. If a -1 is sent, it calculates the length.
Return Value	Void
Precondition	The LCD must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15.7 Function void LCDGotoXY (T_UBYTE, T_UBYTE)

Description	Sets the position of the cursor to a specific part of the screen.
Parameter 1	Receives the x value for the position.
Parameter 2	Receives the y value for the position.
Return Value	Void
Precondition	The LCD must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15.8 Function void LCDBusyLoop (void)

Description	Function that checks whether the LCD is in busy state.
Parameter 1	Void
Return Value	Void
Precondition	The LCD must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15.9 Function void delay_1us (void)

Description	Function that makes a delay of one microsecond.
Parameter 1	Void
Return Value	Void
Precondition	STM timer must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15.10 Function void delay_500ns (void)

Description	Function that makes a delay of 500 nanoseconds.
Parameter 1	Void
Return Value	Void
Precondition	STM timer must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.

5.15.11 Function void delay_30ms (void)

Description	Function that makes a delay of 30 miliseconds.
Parameter 1	Void
Return Value	Void
Precondition	STM timer must be initialized before.
Post condition	Does not apply.
Error Conditions	Does not apply.