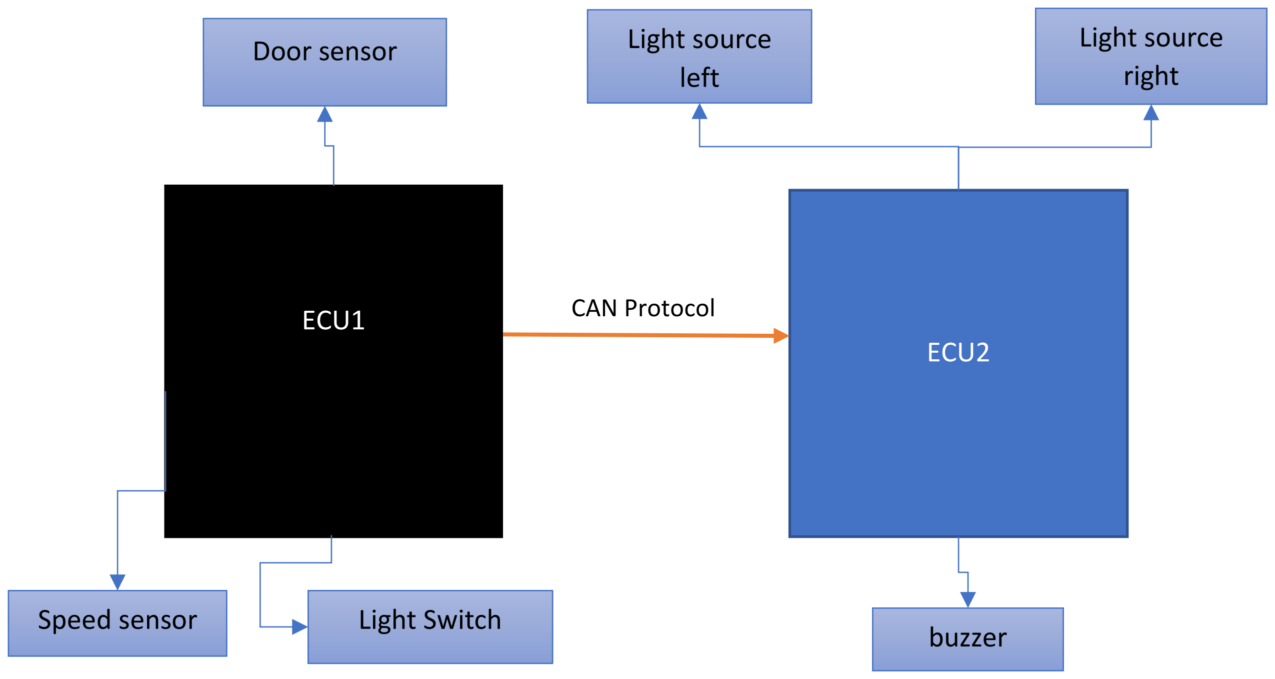
Static Design

Basic design:



Pseudocode:

1- ECU1:

MCAL Layer:

For identifying GPIO pins control a GPIO module is needed to specify input and output pins, reading, and writing to pins.
For communication (sending status massages) a CAN module is needed.
To read the speed we need to know if the motor is moving or stopped, so an ADC module is not needed.
To time the massages sending a TIMER module is needed.

On-Board Layer:

Speed sensor module is needed which uses GPIO module.
Light Switch module is needed which uses GPIO module.
Door sensor module is needed which uses GPIO module.
Communication handler module is needed to connect service layer with MCAL layer (CAN module).

Service layer:

OS to handle all the layers.
Communication manager will be needed.

Standard lib layer:

STD Types.

STD Macros.

Hardware standards.

Application layer:

Main file to make the program.

2- ECU2:

For identifying GPIO pins control a GPIO module is needed to specify input and output pins, reading, and writing to pins.

For communication (sending status massages) a CAN module is needed.

To time the massages sending a TIMER module is needed and the time for closing the light source.

On-Board Layer:

Light source module is needed which uses GPIO module.

Buzzer module is needed that uses GPIO module.

Communication handler module is needed to connect service layer with MCAL layer (CAN module).

Service layer:

OS to handle all the layers.

Communication manager will be needed.

Standard lib layer:

STD Types.

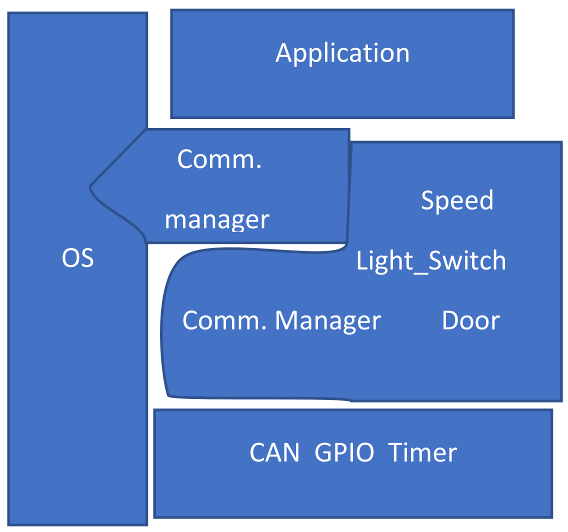
STD Macros.

Hardware standards.

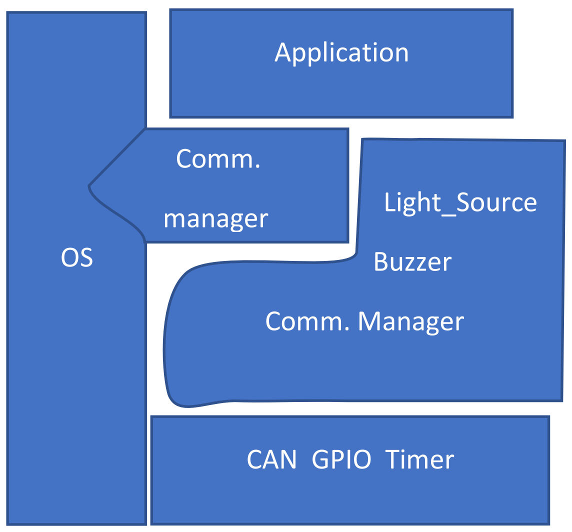
Application layer:

Main file to make the program.

Finale layer architecture:



ECU 1



ECU 2

Modules Contents:

Common Modules:

1- CAN:
APIs:

a. CAN_vidInit

Description	Initialize CAN
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	CAN_config_t *
Output	None
Return	None

b. CAN_u8SendData

Description	Send data through the CAN
Reentrancy	Non Reentrant
Synchronization	Synchronous
Type	Sender
Input	CAN_Data_t *
Output	Number of bytes sent
Return	1 (if data is sent) 0(if some data is not sent)

c. CAN_xRecieveData

Description	Receive data through the CAN
Reentrancy	Reentrant
Synchronization	Asynchronous
Type	Getter
Input	None
Output	None
Return	CAN_Data_t *

Typedefs:

a. CAN_config_t
   A structure that contains all required data to configure CAN communication.
b. CAN_Data_t

   The structure that contains the form of data sent.

2- Communication_Manager
   APIs:
   a. CANM_Init()

Description	Initiate the CAN module
Reentrancy	Reentrant
Synchronization	synchronous
Type	Init
Input	None
Output	None
Return	None

   b. CANM_SendStatus

Description	Receive data through the CAN
Reentrancy	Reentrant
Synchronization	Asynchronous
Type	Setter
Input	1- StatusSent 2- CurrentStatus
Output	None
Return	None

3- GPIO
APIs:
   a. GPIO_vidInit

Description	Initialize GPIO pins
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	None (use an array of structures inside GPIO_Config)
Output	None
Return	None

b. GPIO_u8GetPinVal

Description	Get a pin status
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Getter
Input	portNumber pinNumber
Output	None
Return	HIGH LOW

c. GPIO_vidSetPinVal

Description	Get a pin status
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Sender
Input	portNumber pinNumber Status
Output	None
Return	None

d. GPIO_vidSetPinDir

Description	Choose pin direction
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Sender
Input	portNumber pinNumber Direction
Output	None
Return	None

e. GPIO_vidInitPin

Description	Choose pin direction and value

Reentrancy	Reentrant
Synchronization	Synchronous
Type	Sender
Input	portNumber pinNumber Direction Value
Output	None
Return	None

f. GPIO_vidEnablePin

Description	Enable/Disable pin
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Setter
Input	portNumber pinNumber EN(0 -> disable, 1 -> enable)
Output	None
Return	None

Typedefs:

1- GPIO_config_t
   A structure that contains all required data to configure a GPIO pin.
2- GPIO_pinNumber
   An enum that contains the available pins.
3- GPIO_portNumber
   An enum that contains the available ports.
4- GPIO_pinValue
   An enum that contains High and Low statuses.
5- GPIO_pinDirection
   An enum that contains Output and Input.

4- Timer:

APIs:

a. Timer_vidInit

Description	Initialize Timers
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	None (use an array of structures inside Timer_Config)
Output	None
Return	None

b. Timer_vidStart

Description	Start a Timer
Reentrancy	Reentrant
Synchronization	Synchronous
Type	
Input	Time in ms (or number of ticks) timer_number
Output	Edit to active_Timer array (put 1 in the array's element that specifies the timer's number)
Return	None

c. Timer_vidStop

Description	Stop a Timer
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	Time in ms (or number of ticks) timer_number
Output	Edit to active_Timer array (put zero)
Return	None

d. Timer_ISR

Description	Timer ISR function
Reentrancy	Non-Reentrant
Synchronization	Synchronous
Type	Setter
Input	None
Output	Add 1 to tickCount
Return	None

Typedefs:

a. Timer_config_t
   A structure that contains all required data to configure a Timer pin.
b. Timer_Id
   An enum that contains all available timers.

ECU1 Modules:

1- Speed

APIs:

a. SpeedSensor_vidInit

Description	Initiate speed sensor
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	None
Output	None
Return	None

b. SpeedSensor_ u8TurnOn

Description	Turn on speed sensor
Reentrancy	Reentrant
Synchronization	Synchronous
Type	
Input	None
Output	None
Return	1(If done) 0(if not)

c. SpeedSensor_u8TurnOff

Description	Turn off speed sensor
Reentrancy	Reentrant
Synchronization	Synchronous
Type	
Input	None
Output	None
Return	1(If done) 0(if not)

d. SpeedSensor_vidGetStatus

Description	Get Motor Status
Reentrancy	Non-Reentrant
Synchronization	Synchronous
Type	Getter
Input	None
Output	MotorStatus (global variable)
Return	None

e. SpeedSensor_vidSendStatus

Description	Send Motor Status
Reentrancy	Non-Reentrant
Synchronization	Asynchronous
Type	Sender
Input	None
Output	None
Return	None

2- Light_Switch
APIs:

a. LightSwitch_vidInit

Description	Initiate Light switch
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	None
Output	None
Return	None

b. LightSwitch_vidGetStatus

Description	Get status of Light switch
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	None
Output	LightStatus
Return	None

c. LightSwitch_vidSendStatus

Description	Send Light Status
Reentrancy	Non-Reentrant
Synchronization	Asynchronous
Type	Sender
Input	None
Output	None
Return	None

3- Door

APIs:

a. DoorSensor_vidInit

Description	Initiate Door sensor
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	None
Output	None
Return	None

b. DoorSensor_ u8TurnOn

Description	Turn on Door sensor
Reentrancy	Reentrant
Synchronization	Synchronous
Type	init
Input	None
Output	None
Return	1(if ok) 0(if not ok)

c. DoorSensor_u8TurnOff

Description	Turn off Door sensor
Reentrancy	Reentrant
Synchronization	Synchronous
Type	De-init
Input	None
Output	None
Return	1(if ok) 0(if not ok)

d. DoorSensor_vidGetStatus

Description	Get Door Status
Reentrancy	Non-Reentrant
Synchronization	Synchronous
Type	Getter
Input	None
Output	DoorStatus (global variable)
Return	None

a. DoorSensor_vidSendStatus

Description	Send Door Status
Reentrancy	Non-Reentrant
Synchronization	Asynchronous
Type	Sender
Input	None
Output	None
Return	None

ECU2 Modules:

1- Light_Source
APIs:

a. LightSource_vidInit

Description	Initiate Light source
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	None
Output	None
Return	None

b. LightSource_u8ChangeStatus

Description	Change status of a Light source
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Setter
Input	The status you want to change the light source to (HIGH or LOW)
Output	None
Return	The current status of the changed Light source.

2- Buzzer

APIs:

a. Buzzer_vidInit

Description	Initiate Buzzer
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Init function
Input	None
Output	None
Return	None

b. Buzzer_u8ChangeStatus

Description	Change status of the buzzer
Reentrancy	Reentrant
Synchronization	Synchronous
Type	Setter
Input	The status you want to change the Buzzer to. (HIGH or LOW)
Output	None
Return	The current status of the buzzer source.