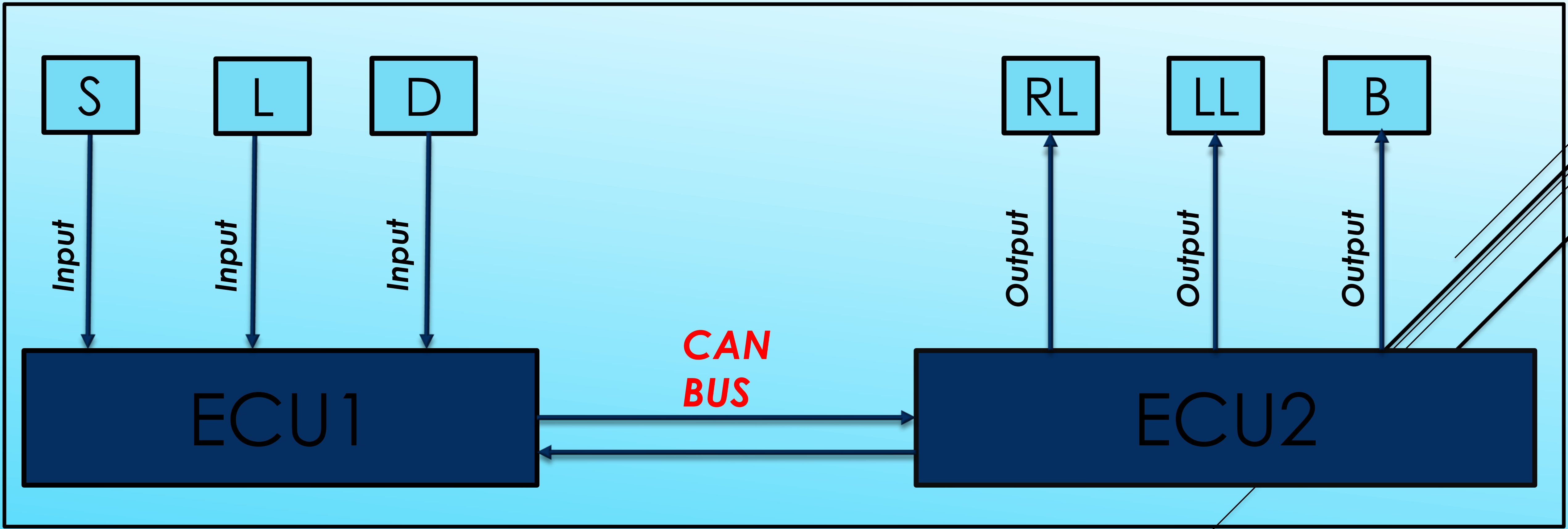
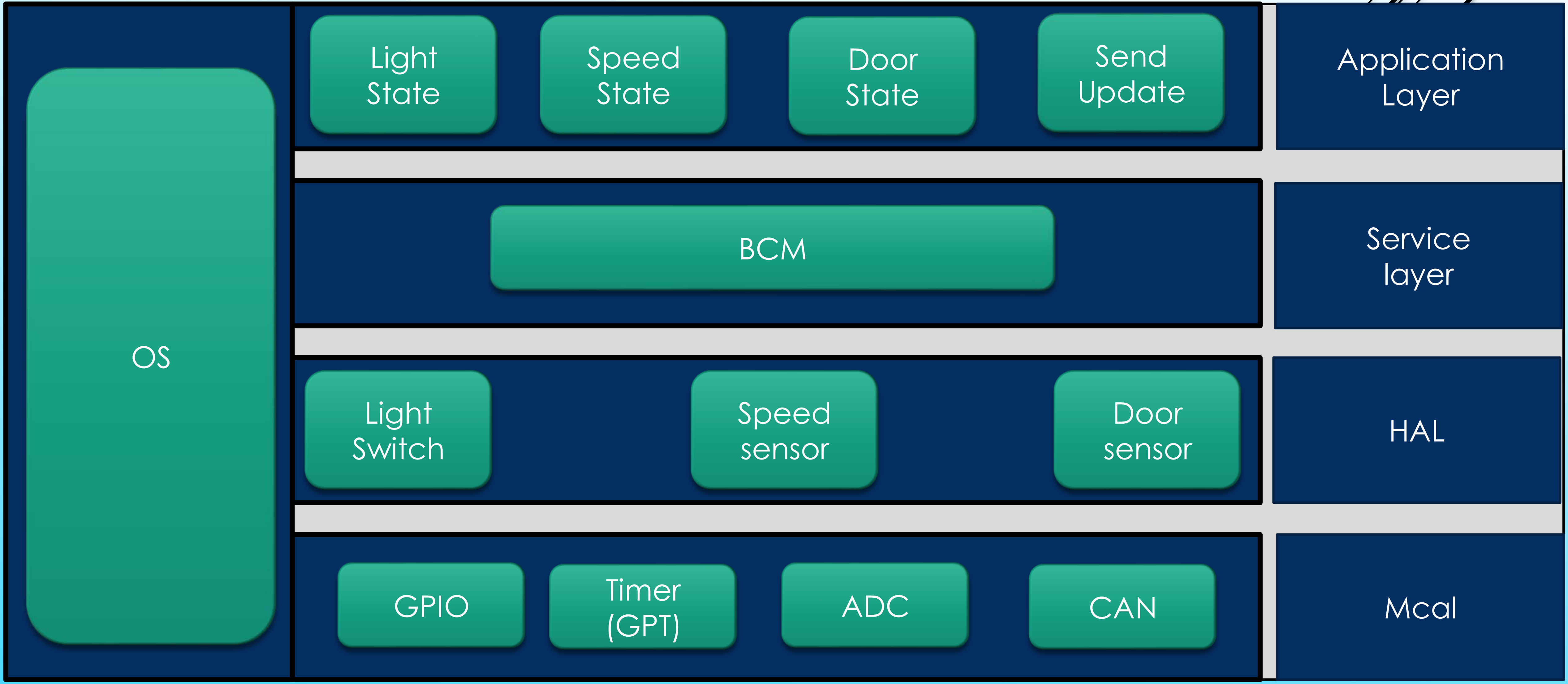
Full System Diagram



ECU_1


OS
Light State
Speed State
Door State
Send Update
Application Layer
BCM
Service layer
Light Switch
Speed sensor
Door sensor
HAL
GPIO
Timer (GPT)
ADC
CAN
Mcal

GPIO APIs	
1	*void* ***GPIO_Init*** *(const GPIO_Config * Config_ptr)*
2	*GPIO_Level* ***GPIO_ReadChannel*** *(GPIO_channel_Type Channel_ID)*
3	*void* ***GPIO_WriteChannel*** *(GPIO_channel Type Channel_ID, GPIO_LevelType level)*

detailed description for the used typedefs

1	GPIO_channel_Type	
	type	unsigned char
	Description	this typedef will be used to select the required pin to be read from or wrie on it

2	GPIO_LevelType	
	type	unsigned char
	Description	this typedef will be used to store a boolen value (0 or 1) in order to determine the selcted pin status (0 -> Low) (1 -> High) or to write a specific level on a specific pin

3	GPIO_Config	
	type	Structure
	Description	This structure is used to configure the port and the pins before using the read & write fnctions this structure is passed to the GPIO_Init as pointer to structure containg the following : (Pin Mode - Pin initial value - pin direction - pin internal resistance attach - Pin alternative function)

Function name	***GPIO_Init***
arguments	(* Config_ptr) : this is a pointer ro structure containg all the required configuration for the required pins
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to intialize the specfied pins throgh the passed pointer to structure in order to operate as required for example : (input - output - internal pull up - internal pull down - alternative function - etc.....)

Function name	*GPIO_ReadChannel*
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) typdef which is used to select the Pin required
Return vlaues	GPIO_Level: This a variable used to return the Pin level (High - Low) of the required pin to be read
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to target a spesific pin in the port then retun it's logic level (High or Low)

Function name	*GPIO_WriteChannel*
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) typdef which is used to select the Pin required level: This a variable from the (GPIO_LevelType) typdef which is used to modfiy the pin mode either High or Low
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to target a spesific pin in the port then write a spesific Logic on it (High - Low)

GPT APIs	
1	*void **GPT_Init** (const GPIO_Config * GPT_Config_ptr)*
2	*void **GPT_Start** (GPT_value Start_value)*
3	*void **GPT_Stop** (GPT_value End_value)*

detailed description for the used typedefs

1	GPT_value	
	type	unsigned long int
	Description	This variable is used to store the vlae of the staring value and end value of the timer to operate a required

2	*GPT_Config_ptr*	
	type	*Structure*
	Description	*This structure is used to configure the pins required in the GPT before using the start and stop fnctions this structure is passed to the GPT_Init as pointer to structure containg the following : (GPT Mode - GPT resulaion - GPT counter type - GPT Prescular factor - etc.......)*

Function name	*GPT_Init*
arguments	(* Config_ptr) : this is a pointer ro structure containg all the required configuration for the required pins and modes to set the GPT
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	ASynchronous
function description	This function is used to intialize the specfied GPT pins throgh the passed pointer to structure in order to operate as required for example : (GPT Mode - GPT resulaion - GPT counter type - GPT Prescular factor - etc.......)

Function name	***GPT_Start***
arguments	Start_value : This a variable from the (GPT_value) typdef which is used to determine the intial strating value of the timer
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	ASynchronous
function description	This function is used to start the GPT wit the suitable starting value

Function name	***GPT_End***
arguments	Start_value : This a variable from the (GPT_value) typdef which is used to determine the End value of the timer
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	ASynchronous
function description	This function is used to start the GPT wit the suitable Ending value

CAN APIs	
1	*void **CAN_Init** (const GPIO_Config * CAN_Config_ptr)*
2	*void **CAN_SetBaudrate** (unsigned char can_handler, uusigned short int Baudrate)*
3	*void **CAN_SendData** (CAN_data data_sent)*
4	*CAN_data **CAN_ReadData** (void)*

detailed description for the used typedefs

1	CAN_data	
	type	unsigned int
	Description	This variable is used to store the vlae of the data sent or read using the CAN operating functions

2	*CAN_Config_ptr*	
	type	*Structure*
	Description	*This structure is used to configure the pins required in the CAN pins before using the operating functions* *this structure is passed to the CAN_Init as pointer to structure containg all the user defined configurations*

Function name	*CAN_Init*
arguments	(* Config_ptr) : this is a pointer ro structure containg all the required configuration for the required pins and modes to set the CAN driver
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to intialize the specfied CAN driver throgh the passed pointer to structure in order to operate as required

Function name	*CAN_SetBaudrate*
arguments	can_handler : This a variable from the (unsigned char) which is used to determine the can handler used Baudrate: This a variable from the (unsigned int) which is used to accurately specify the Baud rate between the two ECUs
Return vlaues	None
Reentrancy	Reentrant
Sync/Async	Synchronous
function description	This functio should set the baud rate configuration of the CAN controller. Depending on necessary baud rate modifications the controller would use.

Function name	*CAN_SendData*
arguments	data_sent : This a variable from the (CAN_data) typdef which is used to store the data transferred
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to send the data throgh the CAN communication protcol after the configurations

Function name	*CAN_ReadData*
arguments	None
Return vlaues	Read_data : This a variable from the (CAN_data) typdef which is used to store the data Read from the other ECU
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to Read the data throgh the CAN communication protcol after the configurations

ADC APIs	
1	void **ADC_Init** (const GPIO_Config * ADC_Config_ptr)
2	ADC_data **ADC_Read** (GPIO_channel_Type Channel_ID)

detailed description for the used typedefs

1	ADC_data	
	type	unsigned int
	Description	This variable is used to store the value of the data read using the ADC
2	*CAN_Config_ptr*	
	type	*Structure*
	Description	*This structure is used to configure the pins required in the ADC pins and ome of operation before using the operating functions this structure is passed to the ADC_Init as pointer to structure containg all the user defined configurations*
3	*Channel_ID*	
	type	*GPIO_channel_Type*
	Description	*This variable is the same used in the GPIO driver which will determine which Pin shall be targeted for reading the analogsignals*

Function name	*ADC_Init*
arguments	(* Config_ptr) : this is a pointer ro structure containg all the required configuration for the required pins and modes to set the ADC driver
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to intialize the specfied ADC driver throgh the passed pointer to structure in order to operate as required

Function name	*ADC_Read*
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which is used to determine which pin shall be read the analog signl from
Return vlaues	ADC_data which is an unsigned int value to be stored as it is the analog reading
Reentrancy	Reentrant
Sync/Async	Synchronous
function description	This functio shall be used to in order determine which pin will be tragted for analog signal reading then the function will return the reading

DOOR APIs

1	*void **DOOR_Init** (GPIO_channel_Type Channel_ID)*
2	*unsigned char **Door_Read** (GPIO_channel_Type Channel_ID)*

Function name	*Door_Init*
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which will be sent to the conguration structure in the GPIO driver to configure the seleted pin to work as required
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to intialize the selected pin to be configured in the GPIO

Function name	*Door_Read*
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which is used to determine which pin will be read from
Return vlaues	unsigned char which will clearify the door status 0 -> opened 1 -> closed
Reentrancy	Reentrant
Sync/Async	Synchronous
function description	This functio shall be used to select the pin used for the door status reading and collect the digital reading this function will call GPIO_Level GPIO_ReadChannel (GPIO_channel_Type Channel_ID) from the GPIO driver

Lights APIs	
1	void **Lights_Init** (GPIO_channel_Type Channel_ID)
2	unsigned char **Lights_Read** (GPIO_channel_Type Channel_ID)

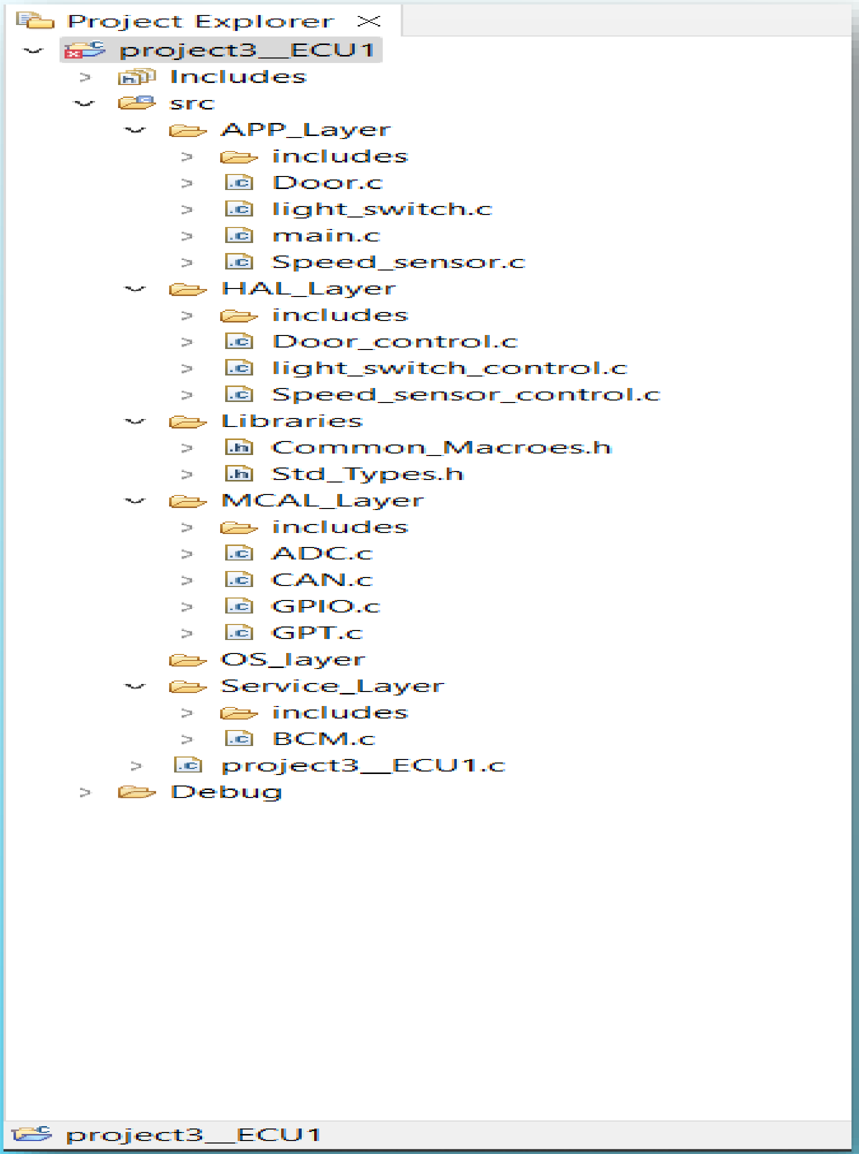
Function name	*Lights_Init*
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which will be sent to the conguration structure in the GPIO driver to configure the seleted pin to work as required
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to intialize the selected pin to be configured in the GPIO

Function name	***Lights_Read***
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which is used to determine which pin will be read from
Return vlaues	unsigned char which will clearify the Light switch status 0 -> opened 1 -> closed
Reentrancy	Reentrant
Sync/Async	Synchronous
function description	This functio shall be used to select the pin used for the door status reading and collect the digital reading this function will call GPIO_Level GPIO_ReadChannel (GPIO_channel_Type Channel_ID) from the GPIO driver

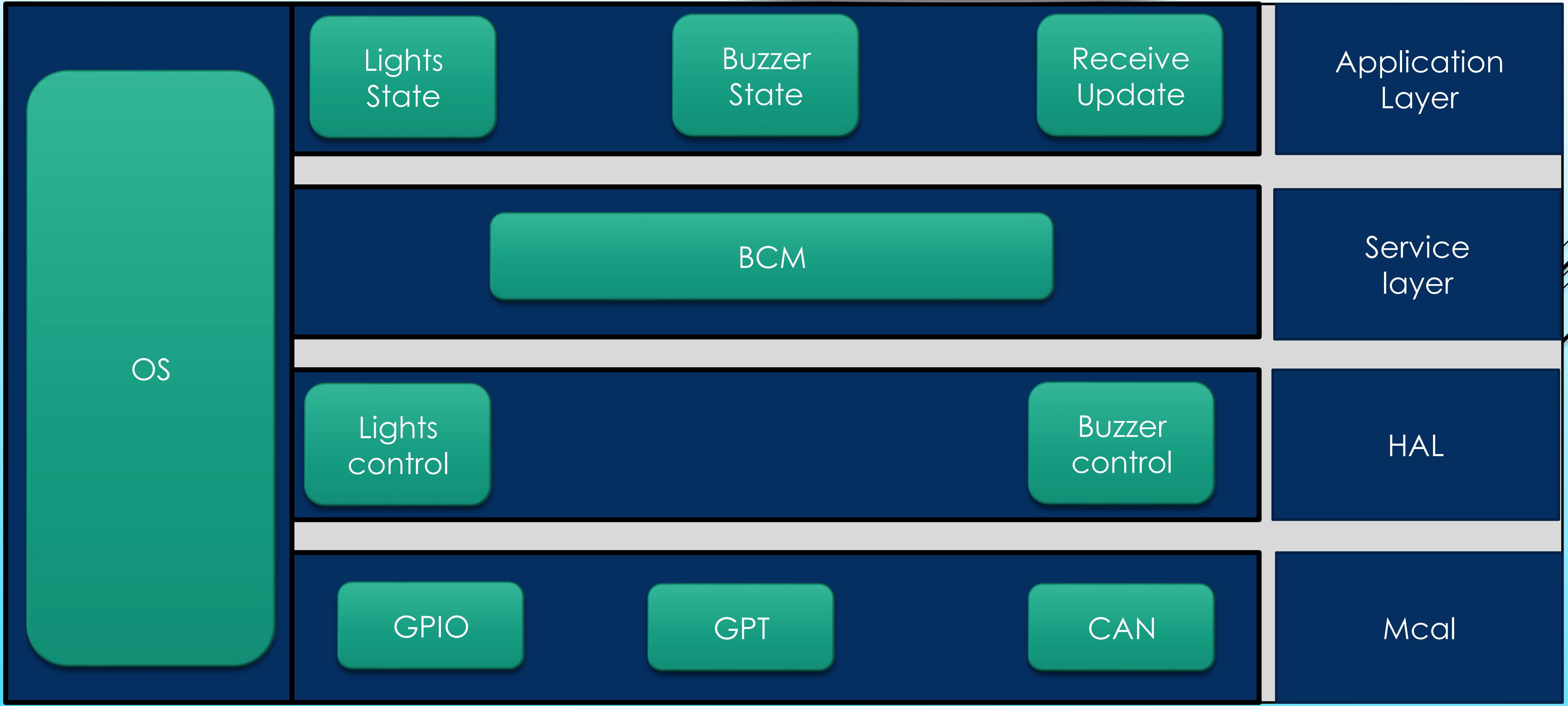
Speed sensor APIs	
1	*void* ***Speed_Init*** *(GPIO_channel_Type ADC_Channel_ID)*
2	*unsigned int* ***Speed_Read*** *(GPIO_channel_Type ADC_Channel_ID)*

Function name	*Speed_Init*
arguments	ADC_Channel_ID : This a variable from the (GPIO_channel_Type) which will be sent to the conguration structure in the GPIO driver to configure the seleted pin to work as required
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to intialize the selected pin to be configured in the GPIO

Function name	***Speed_Read***
arguments	ADC_Channel_ID : This a variable from the (GPIO_channel_Type) which is used to determine which pin will be read from
Return vlaues	unsigned char which will store the analog readings of the speed sensor
Reentrancy	Reentrant
Sync/Async	Synchronous
function description	This functio shall be used to select the pin used for the door status reading and collect the digital reading this function will call ADC_data ADC_Read (GPIO_channel_Type Channel_ID) from the ADC driver


```
Project Explorer
project3__ECU1
  > Includes
  v src
    v APP_Layer
      > includes
      > Door.c
      > light_switch.c
      > main.c
      > Speed_sensor.c
    v HAL_Layer
      > includes
      > Door_control.c
      > light_switch_control.c
      > Speed_sensor_control.c
    v Libraries
      > Common_Macroes.h
      > Std_Types.h
    v MCAL_Layer
      > includes
      > ADC.c
      > CAN.c
      > GPIO.c
      > GPT.c
      OS_layer
    v Service_Layer
      > includes
      > BCM.c
    > project3__ECU1.c
  > Debug
project3__ECU1
```

ECU_2


OS
Lights State
Buzzer State
Receive Update
Application Layer
BCM
Service layer
Lights control
Buzzer control
HAL
GPIO
GPT
CAN
Mcal

GPIO APIs	
1	void GPIO_Init (const GPIO_Config * Config_ptr)
2	GPIO_Level GPIO_ReadChannel (GPIO_channel_Type Channel_ID)
3	void GPIO_WriteChannel (GPIO_channel Type Channel_ID, GPIO_LevelType level)

detailed description for the used typedefs

1	GPIO_channel_Type	
	type	unsigned char
	Description	this typedef will be used to select the required pin to be read from or wrie on it

2	GPIO_LevelType	
	type	unsigned char
	Description	this typedef will be used to store a boolen value (0 or 1) in order to determine the selcted pin status (0 -> Low) (1 -> High) or to write a specific level on a specific pin

3	*GPIO_Config*	
	type	*Structure*
	Description	*This structure is used to configure the port and the pins before using the read & write fnctions this structure is passed to the GPIO_Init as pointer to structure containg the following : (Pin Mode - Pin initial value - pin direction - pin internal resistance attach - Pin alternative function)*

Function name	*GPIO_Init*
arguments	(* Config_ptr) : this is a pointer ro structure containg all the required configuration for the required pins
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to intialize the specfied pins throgh the passed pointer to structure in order to operate as required for example : (input - output - internal pull up - internal pull down - alternative function - etc…..)

Function name	*GPIO_ReadChannel*
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) typdef which is used to select the Pin required
Return vlaues	GPIO_Level: This a variable used to return the Pin level (High - Low) of the required pin to be read
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to target a spesific pin in the port then retun it's logic level (High or Low)

Function name	***GPIO_WriteChannel***
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) typdef which is used to select the Pin required level: This a variable from the (GPIO_LevelType) typdef which is used to modfiy the pin mode either High or Low
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to target a spesific pin in the port then write a spesific Logic on it (High - Low)

CAN APIs	
1	*void* ***CAN_Init*** *(const GPIO_Config * CAN_Config_ptr)*
2	*void* ***CAN_SetBaudrate*** *(unsigned char can_handler, uusigned short int Baudrate)*
3	*void* ***CAN_SendData*** *(CAN_data data_sent)*
4	*CAN_data* ***CAN_ReadData*** *(void)*

detailed description for the used typedefs

1	CAN_data	
	type	unsigned int
	Description	This variable is used to store the vlae of the data sent or read using the CAN operating functions

2	CAN_Config_ptr	
	type	Structure
	Description	This structure is used to configure the pins required in the CAN pins before using the operating functions this structure is passed to the CAN_Init as pointer to structure containg all the user defined configurations

Function name	*CAN_Init*
arguments	(* Config_ptr) : this is a pointer ro structure containg all the required configuration for the required pins and modes to set the CAN driver
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to intialize the specfied CAN driver throgh the passed pointer to structure in order to operate as required

Function name	*CAN_SetBaudrate*
arguments	can_handler : This a variable from the (unsigned char) which is used to determine the can handler used Baudrate: This a variable from the (unsigned int) which is used to accurately specify the Baud rate between the two ECUs
Return vlaues	None
Reentrancy	Reentrant
Sync/Async	Synchronous
function description	This functio should set the baud rate configuration of the CAN controller. Depending on necessary baud rate modifications the controller would use.

Function name	***CAN_SendData***
arguments	data_sent : This a variable from the (CAN_data) typdef which is used to store the data transferred
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to send the data throgh the CAN communication protcol after the configurations

Function name	*CAN_ReadData*
arguments	None
Return vlaues	Read_data : This a variable from the (CAN_data) typdef which is used to store the data Read from the other ECU
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function is used to Read the data throgh the CAN communication protcol after the configurations

Right light APIs	
1	void **R_light_ON** (GPIO_channel_Type Channel_ID)
2	void **R_light_OFF** (GPIO_channel_Type Channel_ID)

Function name	*R_light_ON*
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which will be used to select the digital pin for the right light
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function shall be used to turn on the Right light in the ECU2 to do this it shall call void GPIO_WriteChannel (GPIO_channel Type Channel_ID, GPIO_LevelType level) from the GPIO driver

Function name	***R_light_OFF***
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which will be used to select the digital pin for the right light
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function shall be used to turn OFF the Right light in the ECU2 to do this it shall call void GPIO_WriteChannel (GPIO_channel Type Channel_ID, GPIO_LevelType level) from the GPIO driver

Left light APIs

1	*void **L_light_ON** (GPIO_channel_Type Channel_ID)*
2	*void **L_light_OFF** (GPIO_channel_Type Channel_ID)*

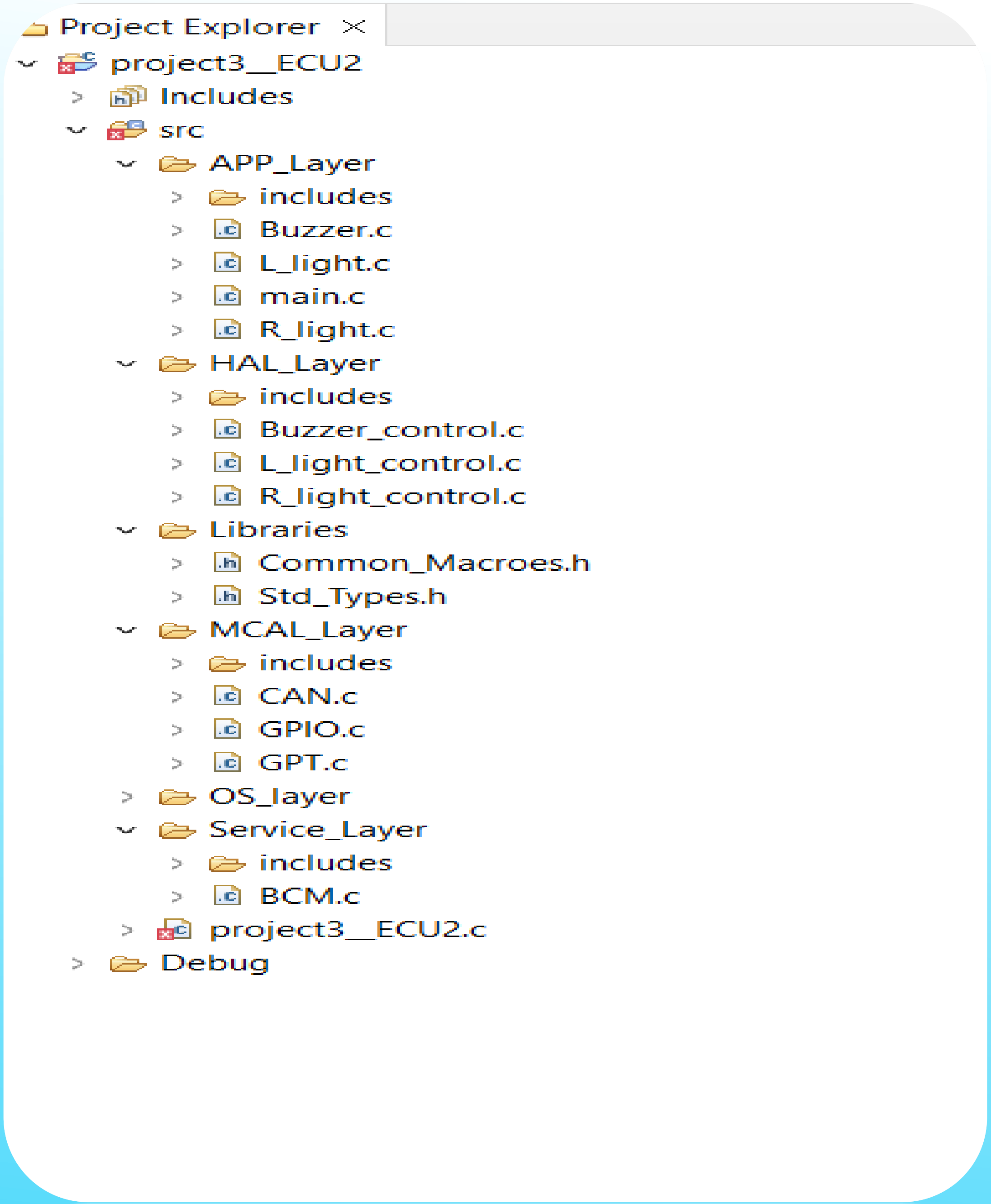
Function name	***L_light_ON***
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which will be used to select the digital pin for the left light
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function shall be used to turn on the Left light in the ECU2 to do this it shall call void GPIO_WriteChannel (GPIO_channel Type Channel_ID, GPIO_LevelType level) from the GPIO driver

Function name	***L_light_OFF***
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which will be used to select the digital pin for the left light
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function shall be used to turn OFF the Left light in the ECU2 to do this it shall call void GPIO_WriteChannel (GPIO_channel Type Channel_ID, GPIO_LevelType level) from the GPIO driver

Buzzer APIs	
1	void **Buzzer_ON** (GPIO_channel_Type Channel_ID)
2	void **Buzzer_OFF** (GPIO_channel_Type Channel_ID)

Function name	***Buzzer_ON***
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which will be used to select the digital pin for the Buzzer in ECU2
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function shall be used to turn on the Buzzer in the ECU2 to do this it shall call void GPIO_WriteChannel (GPIO_channel Type Channel_ID, GPIO_LevelType level) from the GPIO driver

Function name	***Buzzer_OFF***
arguments	Channel_ID : This a variable from the (GPIO_channel_Type) which will be used to select the digital pin for the Buzzer
Return vlaues	None
Reentrancy	None reentrant
Sync/Async	Synchronous
function description	This function shall be used to turn OFF the Buzzer in the ECU2 to do this it shall call void GPIO_WriteChannel (GPIO_channel Type Channel_ID, GPIO_LevelType level) from the GPIO driver



```
Project Explorer
project3__ECU2
  Includes
  src
    APP_Layer
      includes
      Buzzer.c
      L_light.c
      main.c
      R_light.c
    HAL_Layer
      includes
      Buzzer_control.c
      L_light_control.c
      R_light_control.c
    Libraries
      Common_Macroes.h
      Std_Types.h
    MCAL_Layer
      includes
      CAN.c
      GPIO.c
      GPT.c
    OS_layer
    Service_Layer
      includes
      BCM.c
    project3__ECU2.c
  Debug
```