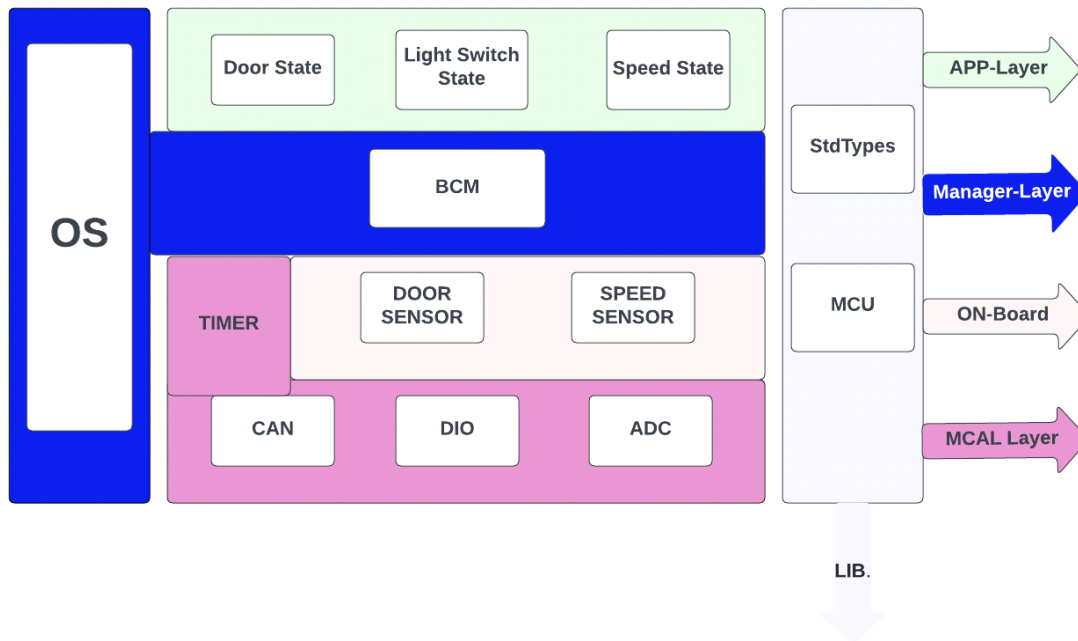


# Static Design

## 1-layered architecture

### a)ECU1



## 2-Provide full detailed APIs for each module as well as a detailed description for the used typedefs

### a)TIMER

Final-project-3 > ECU\_1 > src > MCAL > Inc > Timers\_API.h

```
1  typedef enum {
2
3      GPT_16_32_bit_Timer_0      ,
4      GPT_16_32_bit_Timer_1      ,
5      GPT_16_32_bit_Timer_2      ,
6      GPT_16_32_bit_Timer_3      ,
7      GPT_16_32_bit_Timer_4      ,
8      GPT_16_32_bit_Timer_5      ,
9      GPT_32_64_bit_Wide_Timer_0  ,
10     GPT_32_64_bit_Wide_Timer_1  ,
11     GPT_32_64_bit_Wide_Timer_2  ,
12     GPT_32_64_bit_Wide_Timer_3  ,
13     GPT_32_64_bit_Wide_Timer_4  ,
14     GPT_32_64_bit_Wide_Timer_5
15
16 }Gpt_ChannelType;
17
18 typedef uint32 Gpt_ValueType;
19
20 typedef enum
21 {
22     GPT_MODE_NORMAL,
23     GPT_MODE_SLEEP
24
25 }Gpt_ModeType;
26
27 typedef enum
28 {
29     GPT_PREDEF_TIMER_1US_16BIT,
30     GPT_PREDEF_TIMER_1US_24BIT,
31     GPT_PREDEF_TIMER_1US_32BIT,
32     GPT_PREDEF_TIMER_100US_32BIT
33 }Gpt_PrefDefTimerType;
34
35 typedef uint32 Gpt_ChannelTickFrequency;
36 typedef uint32 GptChannelTickValueMax;
37
38 typedef enum
39 {
40     GPT_CH_MODE_PERIODIC,
41     GPT_CH_MODE_ONESHOT
42 }ChannelMode;
```

inal-project-3 > ECU\_1 > src > MCAL > Inc > Timers\_API.h

```
44  typedef void(*GptNotification)(void);
45
46  typedef struct
47  {
48      Gpt_ChannelType          channel;
49      Gpt_ValueType            channelTickFreq;
50      GptChannelTickValueMax    channelTickMaxValue;
51      ChannelMode               channelMode;
52      GptNotification           gptNotification;
53  }Gpt_ConfigType;
54
55  extern const Gpt_ConfigType Gpt_Config[];
56
57  extern DIO_LevelType Timer_Flag ;
58
59
60  ****
61  * \Syntax      : Gpt_Init( const Gpt_ConfigType* ConfigPtr)
62  * \Description : Initialization of Timer
63  * \Sync\Async  : Synchronous
64  * \Reentrancy  : Non Reentrant
65  * \Parameters (in) : ConfigPtr
66  * \Parameters (out): void
67  * \Return value:  : void
68  * \Arguments Type :None
69  * \Arguments Range:None
70  * \Arguments size :None
71  *****/
72
73  void Gpt_Init( const Gpt_ConfigType* ConfigPtr);
74
75  ****
76  * \Syntax      : Gpt_DisableNotification( Gpt_ChannelType Channel )
77  * \Description : Disable Interrupt for a specific timer
78  * \Sync\Async  : Synchronous
79  * \Reentrancy  : Non Reentrant
80  * \Parameters (in) : Channel
81  * \Parameters (out): void
82  * \Return value:  : void
83  * \Arguments Type :Channel-> typedef enum
84  * \Arguments Range:Channel-> 0-11
85  * \Arguments size :12
```

Final-project-3 > ECU\_1 > src > MCAL > Inc > Timers\_API.h

```
86  ******/
87
88  void Gpt_DisableNotification( Gpt_ChannelType Channel );
89
90  *****
91  * \Syntax      : Gpt_EnableNotification( Gpt_ChannelType Channel )
92  * \Description : Enable Interrupt for a specific timer
93  * \Sync\Async  : Synchronous
94  * \Reentrancy  : Non Reentrant
95  * \Parameters (in) : Channel
96  * \Parameters (out): void
97  * \Return value: : void
98  * \Arguments Type :Channel-> typedef enum
99  * \Arguments Range:Channel-> 0-11
100 * \Arguments size :12
101 ******/
102
103 void Gpt_EnableNotification( Gpt_ChannelType Channel );
104
105 *****
106 * \Syntax      : Gpt_GetTimeElapsed( Gpt_ChannelType Channel )
107 * \Description : GetTimeElapsed of specific timer
108 * \Sync\Async  : Synchronous
109 * \Reentrancy  : Non Reentrant
110 * \Parameters (in) : Channel
111 * \Parameters (out): void
112 * \Return value: : void
113 * \Arguments Type :Channel-> typedef enum
114 * \Arguments Range:Channel-> 0-11
115 * \Arguments size :12
116 ******/
117
118 Gpt_ValueType Gpt_GetTimeElapsed( Gpt_ChannelType Channel );
119
120 *****
121 * \Syntax      : Gpt_GetTimeRemaining( Gpt_ChannelType Channel )
122 * \Description : GetTimeRemaining of specific timer
123 * \Sync\Async  : Synchronous
124 * \Reentrancy  : Non Reentrant
125 * \Parameters (in) : Channel
126 * \Parameters (out): void
127 * \Return value: : void
```

```

Final-project-3 > ECU_1 > src > MCAL > Inc > C Timers_APL.h
125 * \Parameters (in) : Channel
126 * \Parameters (out): void
127 * \Return value: : void
128 * \Arguments Type :Channel-> typedef enum
129 * \Arguments Range:Channel-> 0-11
130 * \Arguments size :12
131 *****/
132
133 Gpt_ValueType Gpt_GetTimeRemaining( Gpt_ChannelType Channel );
134
135 *****
136 * \Syntax : Gpt_StartTimer( Gpt_ChannelType Channel, Gpt_ValueType Value )
137 * \Description : Start a specific Timer by setting tick count
138 * \SyncAsync : Synchronous
139 * \Reentrancy : Non Reentrant
140 * \Parameters (in) : Channel,Value
141 * \Parameters (out): void
142 * \Return value: : void
143 * \Arguments Type :Channel-> typedef enum ,Value-> typedef uint32
144 * \Arguments Range:Channel-> 0-11 , Value-> Norange
145 * \Arguments size :12 , uint32 (16bit)
146 *****/
147
148 void Gpt_StartTimer( Gpt_ChannelType Channel, Gpt_ValueType Value );
149
150 *****
151 * \Syntax : Gpt_StopTimer( Gpt_ChannelType Channel )
152 * \Description : Stop a specific Timer
153 * \SyncAsync : Synchronous
154 * \Reentrancy : Non Reentrant
155 * \Parameters (in) : Channel
156 * \Parameters (out): void
157 * \Return value: : void
158 * \Arguments Type :Channel-> typedef enum
159 * \Arguments Range:Channel-> 0-11
160 * \Arguments size :12
161 *****/
162
163 void Gpt_StopTimer( Gpt_ChannelType Channel );
164

```

## **b)DIO**

Final-project-3 > ECU\_1 > src > MCAL > Inc > C DIO\_APIS.h

```
1
2  typedef enum {
3
4      PIN0                ,
5      PIN1                ,
6      PIN2                ,
7      PIN3                ,
8      PIN4                ,
9      PIN5                ,
10     PIN6                ,
11     PIN7
12
13 }DIO_ChannelType;
14
15 typedef enum {
16
17     PortA                ,
18     PortB                ,
19     PortC                ,
20     PortD                ,
21     PortE                ,
22     PortF
23
24 }DIO_PortType;
25
26 typedef enum {
27
28     LOW                  ,
29     HIGH
30
31 }DIO_LevelType;
32
33
34 typedef struct {
35
36
37
38     DIO_PortType  Port_num        ;
39     DIO_ChannelType Chann_n      ;
40
41 }Channel_Id_Types;
42
```

Final-project-3 > ECU\_1 > src > MCAL > Inc > C DIO\_APIS.h

```
43
44     typedef uint32  DIO_PortLevelType ;
45
46
47
48     *****
49     * \Syntax      : DIO_LevelType Dio_ReadChannel(Channel_Id_Types ChannelId)
50     * \Description  : Read PIN by its Pin number and return its value
51     * \Sync\Async   : Synchronous
52     * \Reentrancy   : Non Reentrant
53     * \Parameters (in) : ChannelId
54     * \Parameters (out): DIO_LevelType
55     * \Return value:  : DIO_LevelType
56     * \Arguments Type :ChannelId-> typedef struct
57     * \Arguments Range:Channel_Id-> 0-1000
58     * \Arguments size :14
59     *****/
60
61     DIO_LevelType Dio_ReadChannel(Channel_Id_Types ChannelId);
62
63     *****
64     * \Syntax      : void Dio_WriteChannel(Channel_Id_Types ChannelId,DIO_LevelType Level)
65     * \Description  : Write on PIN High or Low
66     * \Sync\Async   : Synchronous
67     * \Reentrancy   : Non Reentrant
68     * \Parameters (in) : ChannelId,Level
69     * \Parameters (out): void
70     * \Return value:  : void
71     * \Arguments Type :ChannelId-> typedef struct ,Level -> typedef enum
72     * \Arguments Range:Channel_Id-> 0-1000 , Level-> 0-1
73     * \Arguments size :Channel_Id->14 ,Level->2
74     *****/
75
76     void Dio_WriteChannel(Channel_Id_Types ChannelId,DIO_LevelType Level);
77
78     *****
79     * \Syntax      : DIO_PortLevelType Dio_ReadPort(DIO_PortType PortId)
80     * \Description  : Read Port and return Port
81     * \Sync\Async   : Synchronous
82     * \Reentrancy   : Non Reentrant
83     * \Parameters (in) : PortId
84     * \Parameters (out): DIO_PortLevelType
```

```

Final-project-3 > ECU_1 > src > MCAL > Inc > C DIO_API.h
84 * \Parameters (out): DIO_PortLevelType
85 * \Return value: : DIO_PortLevelType
86 * \Arguments Type :PortId-> typedef enum
87 * \Arguments Range:PortId-> 0-5
88 * \Arguments size :PortId->6
89 *****/
90
91 DIO_PortLevelType Dio_ReadPort(DIO_PortType PortId);
92
93 *****
94 * \Syntax : Dio_WritePort(DIO_PortType PortId,DIO_PortLevelType Level)
95 * \Description : Write on Port
96 * \Sync\Async : Synchronous
97 * \Reentrancy : Non Reentrant
98 * \Parameters (in) : PortId,Level
99 * \Parameters (out): void
100 * \Return value: : void
101 * \Arguments Type :PortId-> typedef enum , Level-> typedef enum
102 * \Arguments Range:PortId-> 0-5 , Level-> 0-1
103 * \Arguments size :PortId->6 , Level-> 2
104 *****/
105
106 void Dio_WritePort(DIO_PortType PortId,DIO_PortLevelType Level);
107
108 *****
109 * \Syntax : Dio_FlipChannel(Channel_Id_Types ChannelId)
110 * \Description : Flip Value on Pin
111 * \Sync\Async : Synchronous
112 * \Reentrancy : Non Reentrant
113 * \Parameters (in) :ChannelId
114 * \Parameters (out): DIO_LevelType
115 * \Return value: : DIO_LevelType
116 * \Arguments Type :ChannelId-> typedef struct
117 * \Arguments Range:Channel_Id-> 0-1000
118 * \Arguments size :14
119 *****/
120
121 DIO_LevelType Dio_FlipChannel(Channel_Id_Types ChannelId);

```

## c)ADC

```

Final-project-3 > ECU_1 > src > MCAL > Inc > C ADC_API.h
1 *****
2 * \Syntax : void ADC_init(void)
3 * \Description : Intialize ADC
4 * \Sync\Async : Synchronous
5 * \Reentrancy : Non Reentrant
6 * \Parameters (in) : void
7 * \Parameters (out): void
8 * \Return value: : void
9 * \Arguments Type : None
10 * \Arguments Range: None
11 * \Arguments size : None
12 *****/
13
14 void ADC_init(void);

```



## **d)CAN**

```
1
2 *****
3 * \Syntax      : CAN_Initialize(uint32 Channel_Id ,uint32 speed, uint32 linkingPort,uint32 interrupt)
4 * \Description : Initialisation of the channel, setting the speed, linking port and interrupt for non PnP devices
5 * \Sync\Async  : Synchronous
6 * \Reentrancy  : Non Reentrant
7 * \Parameters (in) : Channel_Id,speed,linkingPort,interrupt
8 * \Parameters (out): void
9 * \Return value: : void
10 * \Arguments Type : uint32,uint32,uint32,uint32
11 * \Arguments Range:Channel_Id-> 0-1000 , speed-> 0-200 , linkingPort ->0-100 , interrupt -> 0-100
12 * \Arguments size :uint32,uint32,uint32,uint32
13 *****/
14
15 void CAN_Initialize(uint32 Channel_Id ,uint32 speed, uint32 linkingPort,uint32 interrupt);
16
17 *****
18 * \Syntax      : CAN_Write(uint32 Channel_Id,uint32 CAN_data)
19 * \Description : Sending a CAN message
20 * \Sync\Async  : Synchronous
21 * \Reentrancy  : Non Reentrant
22 * \Parameters (in) : Channel_Id,CAN_data
23 * \Parameters (out): void
24 * \Return value: : void
25 * \Arguments Type : uint32,uint32
26 * \Arguments Range:Channel_Id-> 0-1000 , CAN_data-> 0-32
27 * \Arguments size :uint32,uint32
28 *****/
29
30 void CAN_Write(uint32 Channel_Id,uint32 CAN_data);
31
```

```
void CAN_Write(uint32 Channel_Id,uint32 CAN_data);

*****
* \Syntax      : CAN_Read(uint32 Channel_Id)
* \Description : Reading a CAN message
* \Sync\Async  : Synchronous
* \Reentrancy  : Non Reentrant
* \Parameters (in) : Channel_Id
* \Parameters (out): uint32
* \Return value: : uint32
* \Arguments Type : uint32
* \Arguments Range:Channel_Id-> 0-1000
* \Arguments size :uint32
*****/

uint32 CAN_Read(uint32 Channel_Id);
```

## **e) DOOR SENSOR**

Final-project-3 > ECU\_1 > src > OnBoard > Inc > C DoorSensor\_API.h

```
1
2
3 typedef enum {
4
5     DoorSensor_0,
6     DoorSensor_1,
7     DoorSensor_2,
8
9 }DoorSensor_Type;
10
11
12 typedef uint32 Sensor_Read ;
13
14
15
16 *****
17 * \Syntax      : DoorSensor_Init(void)
18 * \Description : DoorSensor_Initilization
19 * \Sync\Async  : Synchronous
20 * \Reentrancy  : Non Reentrant
21 * \Parameters (in) : void
22 * \Parameters (out): void
23 * \Return value: : void
24 * \Arguments Type : None
25 * \Arguments Range: None
26 * \Arguments size : None
27 ******/
28
29 void DoorSensor_Init(void);
30
31
32 *****
33 * \Syntax      : DoorSensor_ReadStatus( DoorSensor_Type Sensor_Num)
34 * \Description : Read the sensor readings every 10 ms
35 * \Sync\Async  : Synchronous
36 * \Reentrancy  : Non Reentrant
37 * \Parameters (in) : ConfigPtr
38 * \Parameters (out): Sensor_Read
39 * \Return value: : Sensor_Read
40 * \Arguments Type : Sensor_Num->typedef enum, Sensor_Read->typedef uint32
41 * \Arguments Range: Sensor_Num->0-2, Sensor_Read-> no range (Integer uint32)
42 * \Arguments size : Sensor_Num->typedef enum, Sensor_Read->typedef uint32
```

Ln 42, Col 76 (232 sele

```
*****
* \Syntax      : DoorSensor_ReadStatus( DoorSensor_Type Sensor_Num)
* \Description : Read the sensor readings every 10 ms
* \Sync\Async  : Synchronous
* \Reentrancy  : Non Reentrant
* \Parameters (in) : ConfigPtr
* \Parameters (out): Sensor_Read
* \Return value: : Sensor_Read
* \Arguments Type : Sensor_Num->typedef enum, Sensor_Read->typedef uint32
* \Arguments Range: Sensor_Num->0-2, Sensor_Read-> no range (Integer uint32)
* \Arguments size : Sensor_Num->typedef enum, Sensor_Read->typedef uint32
******/
Sensor_Read DoorSensor_ReadStatus( DoorSensor_Type Sensor_Num);
```

## **f)SPEED SENSOR**

Final-project-3 > ECU\_1 > src > OnBoard > Inc > SpeedSensor.h

```
1  typedef enum {
2
3      SpeedSensor_0,
4      SpeedSensor_1,
5      SpeedSensor_2,
6
7  }SpeedSensor_Type;
8
9
10 typedef uint32  Sensor_Read ;
11
12
13
14
15 *****
16 * \Syntax      : SpeedSensor_Init(void)
17 * \Description : SpeedSensor Initialization
18 * \Sync\Async  : Synchronous
19 * \Reentrancy  : Non Reentrant
20 * \Parameters (in) : void
21 * \Parameters (out): void
22 * \Return value: : void
23 * \Arguments Type : None
24 * \Arguments Range: None
25 * \Arguments size : None
26 *****/
27
28 void SpeedSensor_Init(void);
29
30 *****
31 * \Syntax      : SpeedSensor_ReadStatus( DoorSensor_Type Sensor_Num)
32 * \Description : Read the sensor readings every 5 ms
33 * \Sync\Async  : Synchronous
34 * \Reentrancy  : Non Reentrant
35 * \Parameters (in) : ConfigPtr
36 * \Parameters (out): Sensor_Read
37 * \Return value: : Sensor_Read
38 * \Arguments Type : Sensor_Num->typedef enum , Sensor_Read->typedef uint32
39 * \Arguments Range: Sensor_Num->0-2 , Sensor_Read-> no range (Integer uint32)
40 * \Arguments size : Sensor_Num->typedef enum , Sensor_Read->typedef uint32|
41 *****/
42
```

```
*****
* \Syntax      : SpeedSensor_ReadStatus( DoorSensor_Type Sensor_Num)
* \Description : Read the sensor readings every 5 ms
* \Sync\Async  : Synchronous
* \Reentrancy  : Non Reentrant
* \Parameters (in) : ConfigPtr
* \Parameters (out): Sensor_Read
* \Return value: : Sensor_Read
* \Arguments Type : Sensor_Num->typedef enum , Sensor_Read->typedef uint32
* \Arguments Range: Sensor_Num->0-2 , Sensor_Read-> no range (Integer uint32)
* \Arguments size : Sensor_Num->typedef enum , Sensor_Read->typedef uint32|
*****/

Sensor_Read SpeedSensor_ReadStatus( DoorSensor_Type Sensor_Num);
```

## g)OS

Final-project-3 > ECU\_1 > src > Service > Inc > OS\_API.h

```
38 void OS_voidDeleteTask(u8 Copy_u8ID);
39
40
41 *****
42 * \Syntax      : OS_voidSuspendTask(u8 Copy_u8ID, u8 Copy_u8SuspendTime)
43 * \Description  : SuspendTask
44 * \Sync\Async   : Synchronous
45 * \Reentrancy   : Non Reentrant
46 * \Parameters (in) : Copy_u8ID,Copy_u8SuspendTime
47 * \Parameters (out): void
48 * \Return value:  : void
49 * \Arguments Type : u8,u8
50 * \Arguments Range:Copy_u8ID-> 0-1000 , Copy_u8SuspendTime-> Integer(no range)
51 * \Arguments size :u8,u8
52
53 ******/
54
55 void OS_voidSuspendTask(u8 Copy_u8ID, u8 Copy_u8SuspendTime);
56
57
58
59 *****
60 * \Syntax      : OS_voidStartScheduler(void)
61 * \Description  : StartScheduler
62 * \Sync\Async   : Synchronous
63 * \Reentrancy   : Non Reentrant
64 * \Parameters (in) : void
65 * \Parameters (out): void
66 * \Return value:  : void
67 * \Arguments Type : None
68 * \Arguments Range: None
69 * \Arguments size : None
70 ******/
71
72 void OS_voidStartScheduler(void);
73
```

```

41 *****
42 * \Syntax      : OS_voidSuspendTask(u8 Copy_u8ID, u8 Copy_u8SuspendTime)
43 * \Description : SuspendTask
44 * \Sync\Async  : Synchronous
45 * \Reentrancy  : Non Reentrant
46 * \Parameters (in) : Copy_u8ID, Copy_u8SuspendTime
47 * \Parameters (out): void
48 * \Return value:  : void
49 * \Arguments Type : u8,u8
50 * \Arguments Range: Copy_u8ID-> 0-1000 , Copy_u8SuspendTime-> Integer(no range)
51 * \Arguments size : u8,u8
52 *****
53 ******/
54
55 void OS_voidSuspendTask(u8 Copy_u8ID, u8 Copy_u8SuspendTime);
56
57
58
59 *****
60 * \Syntax      : OS_voidStartScheduler(void)
61 * \Description : StartScheduler
62 * \Sync\Async  : Synchronous
63 * \Reentrancy  : Non Reentrant
64 * \Parameters (in) : void
65 * \Parameters (out): void
66 * \Return value:  : void
67 * \Arguments Type : None
68 * \Arguments Range: None
69 * \Arguments size : None
70 ******/
71
72 void OS_voidStartScheduler(void);
73
74

```

Final-project-3 > ECU\_1 > src > Service > Inc > C OS\_API.h

```
73
74
75 *****
76 * \Syntax      : OS_voidResumeTask(u8 Copy_u8ID)
77 * \Description : ResumeTask
78 * \Sync\Async  : Synchronous
79 * \Reentrancy  : Non Reentrant
80 * \Parameters (in) : Copy_u8ID
81 * \Parameters (out): void
82 * \Return value:  : void
83 * \Arguments Type : u8
84 * \Arguments Range:Copy_u8ID-> 0-1000
85 * \Arguments size :u8
86 *****/
87
88 void OS_voidResumeTask(u8 Copy_u8ID);
89
90
91 *****
92 * \Syntax      : OS_u8GetTaskState(u8 Copy_u8ID)
93 * \Description : GetTaskState
94 * \Sync\Async  : Synchronous
95 * \Reentrancy  : Non Reentrant
96 * \Parameters (in) : Copy_u8ID
97 * \Parameters (out): void
98 * \Return value:  : void
99 * \Arguments Type : u8
100 * \Arguments Range:Copy_u8ID-> 0-1000
101 * \Arguments size :u8
102 *****/
103
104 u8 OS_u8GetTaskState(u8 Copy_u8ID);
105
```

## **h)BCM**

```

1
2 *****
3 * \Syntax      : BCM_init(void)
4 * \Description : Initialize BCM
5 * \Sync\Async  : Synchronous
6 * \Reentrancy  : Non Reentrant
7 * \Parameters (in) : void
8 * \Parameters (out): void
9 * \Return value: : void
10 * \Arguments Type : None
11 * \Arguments Range: None
12 * \Arguments size : None
13 ******/
14
15 void BCM_init(void);

```

## **I) DOOR STATE**

```

Final-project-3 > ECU_1 > src > APP > Inc > DoorState_API.h
1
2
3 *****
4 * \Syntax      : DoorSensor_SendState_10ms( DoorSensor_Type Sensor_Num)
5 * \Description : Door state message will be sent every 10 ms to ECU 2
6 * \Sync\Async  : Synchronous
7 * \Reentrancy  : Non Reentrant
8 * \Parameters (in) : ConfigPtr
9 * \Parameters (out): void
10 * \Return value: : void
11 * \Arguments Type : Sensor_Num->typedef enum
12 * \Arguments Range: Sensor_Num->0-2
13 * \Arguments size : Sensor_Num->typedef enum
14 ******/
15
16 void DoorSensor_SendState_10ms( DoorSensor_Type Sensor_Num);

```

## **J) SPEED SENSOR STATE**



```

Final-project-3 > ECU_1 > src > APP > Inc > C SpeedState_API.h
1
2
3
4 *****
5 * \Syntax      : SpeedSensor_SendState_10ms( DoorSensor_Type Sensor_Num)
6 * \Description : Speed state message will be sent every 5 ms to ECU 2
7 * \Sync\Async  : Synchronous
8 * \Reentrancy  : Non Reentrant
9 * \Parameters (in) : ConfigPtr
10 * \Parameters (out): void
11 * \Return value: : void
12 * \Arguments Type : Sensor_Num->typedef enum
13 * \Arguments Range: Sensor_Num->0-2
14 * \Arguments size : Sensor_Num->typedef enum |
15 ******/
16
17 void SpeedSensor_SendState_10ms( DoorSensor_Type Sensor_Num);

```

## **K)LIGHTSWITCH STATE**

```

Final-project-3 > ECU_1 > src > APP > Inc > C Switch_API.h
1
2
3 *****
4 * \Syntax      : Switch_init(Channel_Id_Types Switch_Id);
5 * \Description : Initialize Switch with its port and pin number
6 * \Sync\Async  : Synchronous
7 * \Reentrancy  : Non Reentrant
8 * \Parameters (in) : Switch_Id
9 * \Parameters (out): void
10 * \Return value: : void
11 * \Arguments Type :Switch_Id-> typedef struct
12 * \Arguments Range:Switch_Id-> 0-1000
13 * \Arguments size :14
14
15 ******/
16
17 void Switch_init(Channel_Id_Types Switch_Id);
18
19 *****
20 * \Syntax      : Switch_SendState_20ms( DoorSensor_Type Sensor_Num)
21 * \Description : Light switch state message will be sent every 20 ms to ECU 2
22 * \Sync\Async  : Synchronous
23 * \Reentrancy  : Non Reentrant
24 * \Parameters (in) : ConfigPtr
25 * \Parameters (out): void
26 * \Return value: : void
27 * \Arguments Type :Switch_Id-> typedef struct
28 * \Arguments Range:Switch_Id-> 0-1000
29 * \Arguments size :14|
30 ******/
31
32 void Switch_SendState_20ms(Channel_Id_Types Switch_Id );

```

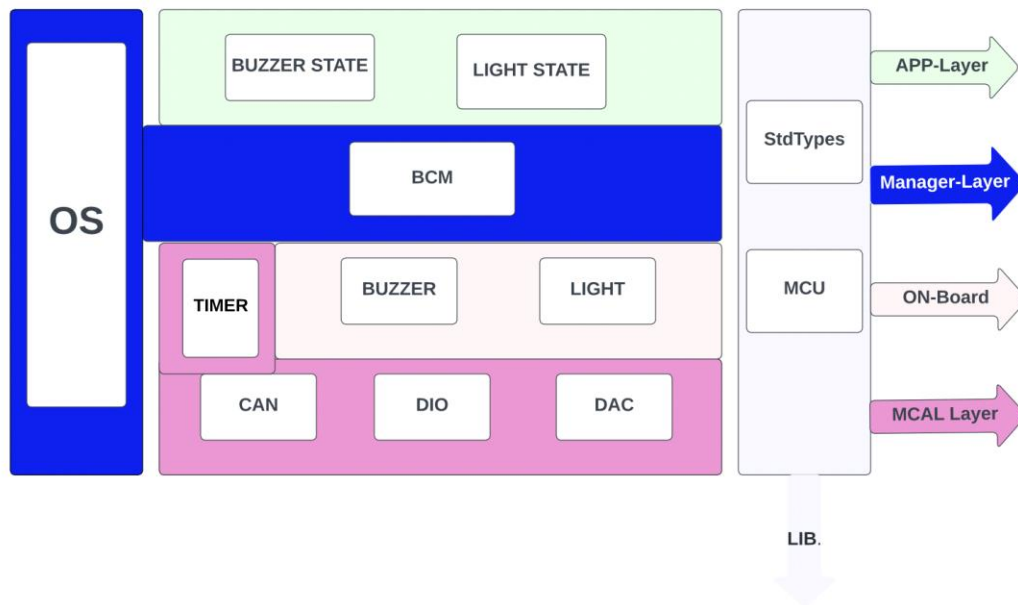
## **L)STD TYPES**



```
typedef unsigned char uint8;
typedef unsigned short int uint16;
typedef unsigned long int uint32;

typedef signed char suint8;
typedef signed short int suint16;
typedef signed long int suint32;
```

## **b)ECU2**



## **A)BUZZER**

```

Final-project-3 > ECU_2 > src > ONBOARD > Inc > C Buzzer.h
1  *****
2  * \Syntax      : $Buzzer_init(Channel_Id_Types Buzzer_Id)
3  * \Description : Intialize BUZZER with its port and pin number
4  * \Sync\Async  : Synchronous
5  * \Reentrancy  : Non Reentrant
6  * \Parameters (in) : Buzzer_Id
7  * \Parameters (out): void
8  * \Return value: : void
9  * \Arguments Type :Buzzer_Id-> typedef struct
10 * \Arguments Range:Buzzer_Id-> 0-1000
11 * \Arguments size :14
12 *****
13
14 void Buzzer_init(Channel_Id_Types Buzzer_Id);

```

## **B)LED**

```

final-project-3 > ECU_2 > src > ONBOARD > Inc > C LED_API.h
1  *****
2  * \Syntax      : LED_init(Channel_Id_Types LED_Id)
3  * \Description : Intialize LED with its port and pin number
4  * \Sync\Async  : Synchronous
5  * \Reentrancy  : Non Reentrant
6  * \Parameters (in) : LED_Id
7  * \Parameters (out): void
8  * \Return value: : void
9  * \Arguments Type :LED_Id-> typedef struct
10 * \Arguments Range:LED_Id-> 0-1000
11 * \Arguments size :14
12 *****
13
14 void LED_init(Channel_Id_Types LED_Id);

```

## **C)DAC**

```

Final-project-3 > ECU_2 > src > MCAL > Inc > C DAC_API.h
1  *****
2  * \Syntax      : void DAC_init(void)
3  * \Description : Initialize ADC
4  * \Sync\Async  : Synchronous
5  * \Reentrancy  : Non Reentrant
6  * \Parameters (in) : void
7  * \Parameters (out): void
8  * \Return value: void
9  * \Arguments Type : None
10 * \Arguments Range: None
11 * \Arguments size : None
12 *****
13
14 void DAC_init(void);

```

## **D)BUZZER STATE**

```

Final-project-3 > ECU_2 > src > APP > Inc > C Buzzer_State_API.h
1  *****
2  * \Syntax      : Buzzer_updateState(uint8 Buzzer_Num)
3  * \Description : Update Buzzer state according to door state and car state
4  * \Sync\Async  : Synchronous
5  * \Reentrancy  : Non Reentrant
6  * \Parameters (in) : Buzzer_Num
7  * \Parameters (out): void
8  * \Return value: void
9  * \Arguments Type : Buzzer_Num->uint8
10 * \Arguments Range: Buzzer_Num-> 0-1000
11 * \Arguments size : Buzzer_Num->uint8
12 *****
13
14
15 void Buzzer_updateState(uint8 Buzzer_Num);

```

## **E)LED STATE**

Final-project-3 > ECU\_2 > src > APP > Inc > C LEDState\_APL.h

```
1  *****
2  * \Syntax      : LED_updateState(uint8 LED_Num)
3  * \Description : Update Buzzer state according to door state and car state
4  * \Sync\Async  : Synchronous
5  * \Reentrancy  : Non Reentrant
6  * \Parameters (in) : LED_Num
7  * \Parameters (out): void
8  * \Return value: : void
9  * \Arguments Type : LED_Num->uint8
10 * \Arguments Range: LED_Num-> 0-1000
11 * \Arguments size : LED_Num->uint8
12 *****
13
14 void LED_updateState(uint8 LED_Num);
```

