SYSTEM SCHEMATIC DIAGRAM

Door Sensor → input → ECU 1 → CAN BUS ⇄ ECU 2 → OUT PUT → Buzzer

Light sensor → input → ECU 1

Speed sensor → input → ECU 1

ECU 2 → OUTPUT → Light RL & LL

STATIC DESIGN OF ECU 1

APP

- Door State
- Speed State
- Light State

Service

- COM Manager
- Device Manager

HAL

- Speed Sensor
- Door Sensor
- Light Switch

MCAL

- CAN
- ADC
- GPIO
- Timer

OS
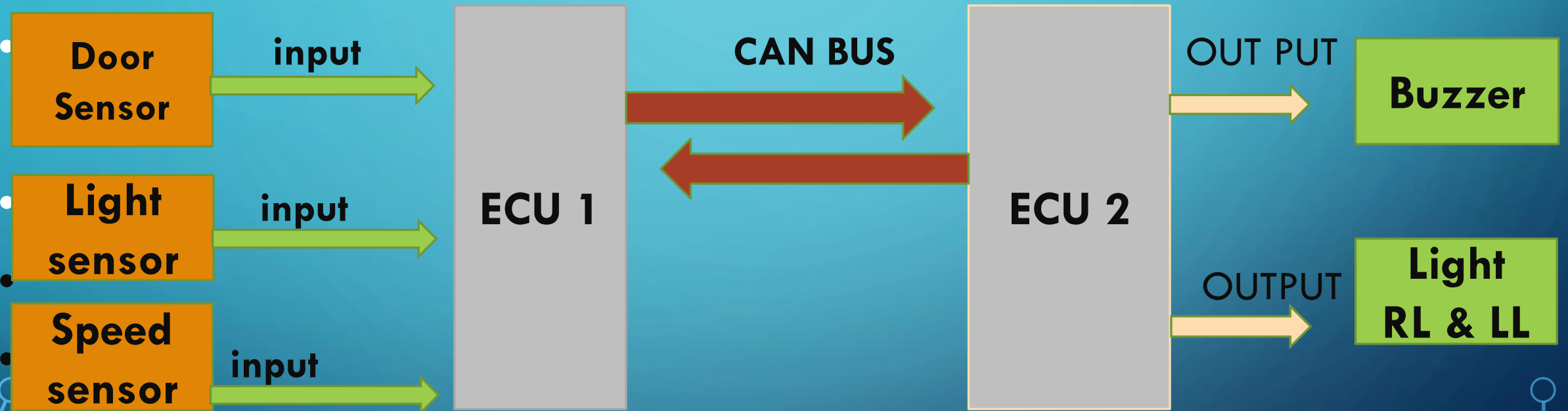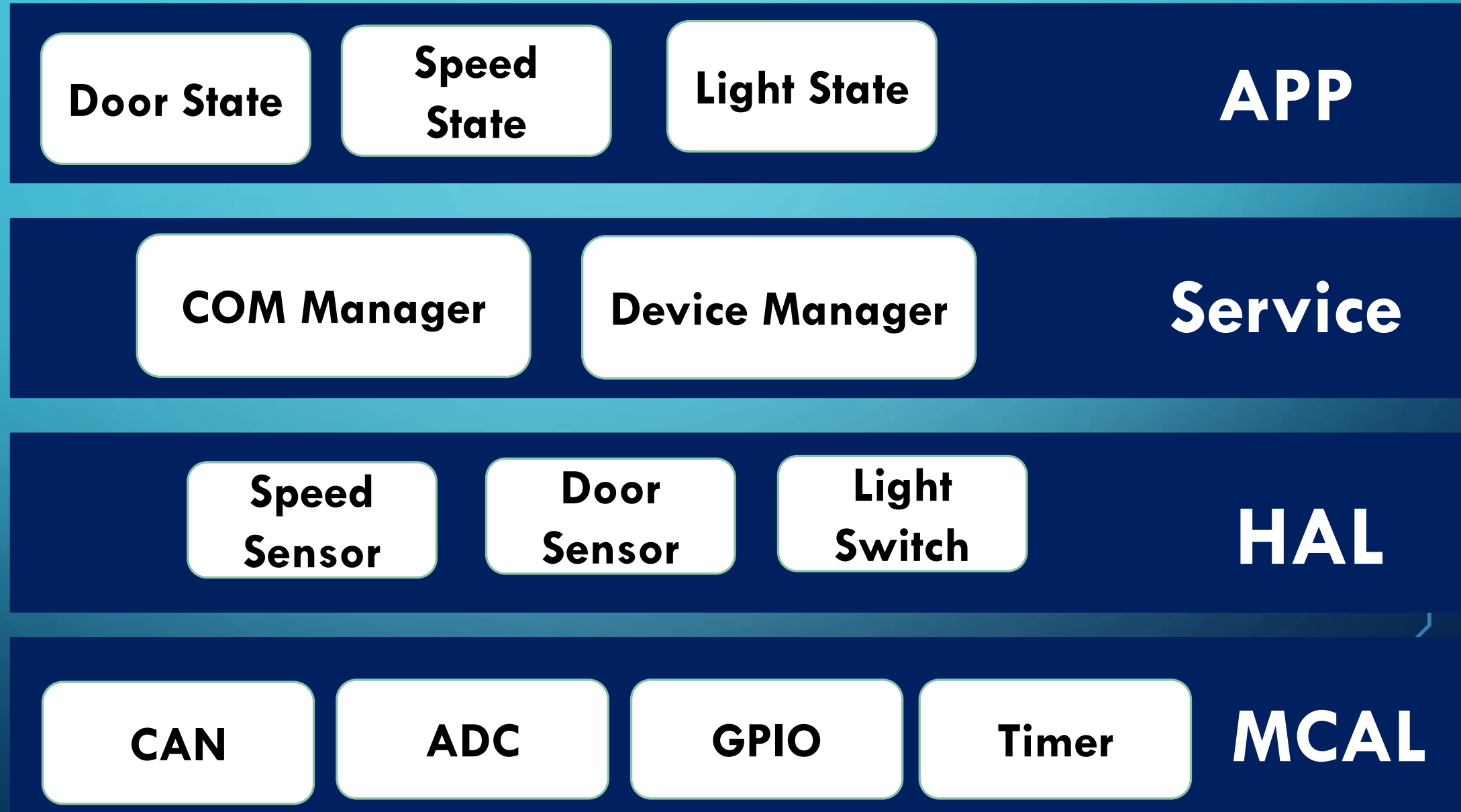
ALL APIS:

- 1- GPIO APIS for ecu1 and ecu2:

Function Name	CPIO_Init
Function prototype	Void GPIO_Init(char Pin_Num, char Port)
Input parameters	GPIO_pin_numper & GPIO_port
Output parameters	NONE
Return	void
Description	Initializing DIO module

Function Name	CPIO_Get_value
Function prototype	char GPIO_Get_value(char Pin_Num, char Port)
Input parameters	GPIO_pin_numper & GPIO_port
Output parameters	Pin_value
Return	character
Description	Getting the value on the pin

Function Name	**CPIO_Set_Pin**
Function prototype	void GPIO_Set_Pin(char Pin_Num, char Port)
Input parameters	GPIO_pin_numper & GPIO_port
Output parameters	None
Return	void
Description	Write on pin high or low

Name	**Port**
prototype	Typedef enum & range from portA to portF
Description	Decimal number for port

Name	**Pin_Num**
prototype	Typedef enum & range from 0 to 7
Description	Decimal number for pin_number

2-TIMERS APIS FOR ECU1 AND ECU2

Function Name	Timer_Init
Function prototype	Void Timer_Init(TimerCfg_t Timer)
Input parameters	TimerCfg_t
Output parameters	NONE
Return	void
Description	Initializing timer

Function Name	Timer_Start
Function prototype	Void Timer_Start(char timer_value)
Input parameters	timer value number of ticks
Output parameters	NONE
Return	void
Description	Starting the timer to count

Function Name	Timer_stop
Function prototype	Void Timer_stop (void)
Input parameters	None
Output parameters	NONE
Return	void
Description	Stopping the timer from counting

Name	TimerCfg_t
prototype	Typedef struct TimerCfg_t
Description	Struct contains all configurations

3-ADC APIS:

Function Name	ADC Init
Function prototype	Void ADC_Init(char ADC_Channel,*ADC_Config)
Input parameters	1- ADC_Channel to indicate which pin used as an ADC 2- array that include struct for every channel that include the configurations set required to initialize ADC
Output parameters	NONE
Return	void
Description	Initializing ADC timer

Function Name	ADC Get Value
Function prototype	Char ADC_Read(char ADC_Channel)
Input parameters	ADC_Channel to indicate which pin used as an ADC
Output parameters	ADC_read_value
Return	Char ADC_value
Description	Getting value from ADC Register and return it

3-ADC APIS:

Name	ADCCfg_t
prototype	Typedef struct ADCCfg_t
Description	Struct contains all configurations

Name	ADC_Channel
prototype	Typedef enum ADC_Channel
Description	Decimal value of channel

4-CAN APIS IN BOTH ECUS:

Function Name	CAN_Init
Function prototype	Void CAN_Init(CANCfg_t CAN)
Input parameters	CANCfg_t >>struct
Output parameters	none
Return	void
Description	Initializing CAN module

Function Name	CAN_Send
Function prototype	Void CAN_Send(char data)
Input parameters	Data >> data send from mcu to another
Output parameters	none
Return	void
Description	Sending data from ecu to another

Function Name	CAN_Recieve
Function prototype	Char CAN_Recieve(void)
Input parameters	none
Output parameters	Value received
Return	char
Description	receiving data from ecu

Name	CANCfg_t
prototype	Typedef struct CANCfg_t
Description	Struct contains all configurations

DOOR SENSOR APIS :

Function Name	Door_Sensor
Function prototype	Void Door_Sensor(void)
Input parameters	none
Output parameters	none
Return	void
Description	Initializing door sensor module

Function Name	Door_Sensor_Read
Function prototype	Char Door_Sensor_Read(void)
Input parameters	none
Output parameters	The value that sensor measure
Return	char
Description	Get the state of the door sensor

LIGHT SWITCH APIS :

Function Name	**Light_Sw_Init**
Function prototype	Void Light_Sw_Init(void)
Input parameters	none
Output parameters	none
Return	void
Description	Initializing light switch sensor module

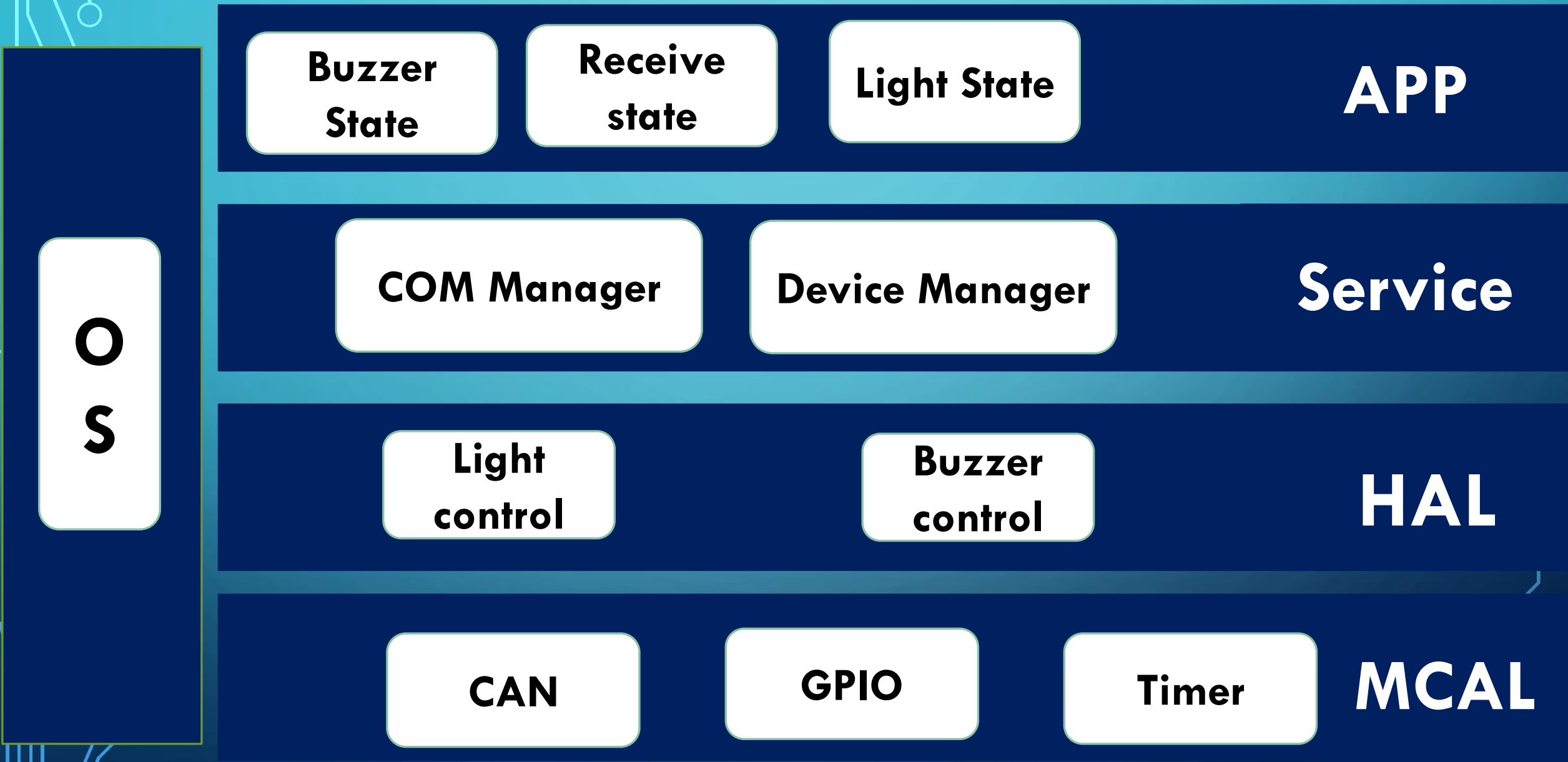|	
Function Name	**Light Sw_Read**
Function prototype	Char Light Sw_Read(void)
Input parameters	none
Output parameters	The value that Light_switch measure
Return	char
Description	Get the state of the Light_switch

SPEED SENSOR APIS :

Function Name	Speed_Sensor_Init
Function prototype	Void Speed_Sensor_Init(void)
Input parameters	none
Output parameters	none
Return	void
Description	Initializing speed sensor module

Function Name	Speed_Sensor_Read
Function prototype	Char Speed_Sensor_Read(void)
Input parameters	none
Output parameters	The value that Speed_Sensor_Read measure
Return	char
Description	Get the state of the Speed_Sensor_Read

STATIC DESIGN OF ECU 2

APP

- Buzzer State
- Receive state
- Light State

Service

- COM Manager
- Device Manager

HAL

- Light control
- Buzzer control

MCAL

- CAN
- GPIO
- Timer

OS

LIGHT APIS RIGHT AND LEFT:

Function Name	**Light_On**
Function prototype	Void Light_On(char PinNum,char Port)
Input parameters	1-PinNum >> pin numper from 0 to 7 2-port >>port from portA to port F
Output parameters	none
Return	void
Description	Make light on for(right & left) by writing high on the pin

Function Name	**Light_OFF**
Function prototype	Void Light_OFF(char PinNum,char Port)
Input parameters	1-PinNum >> pin numper from 0 to 7 2-port >>port from portA to port F
Output parameters	none
return	void
Description	Make light off for(right & left) by writing high on the pin

BUZZER APIS :

Function Name	Buzzer_On
Function prototype	Void Buzzer_On(char PinNum,char Port)
Input parameters	1-PinNum >> pin numper from 0 to 7 2-port >>port from portA to port F
Output parameters	none
return	void
Description	Turn on the buzzer

Function Name	Buzzer_OFF
Function prototype	Void Buzzer_OFF(char PinNum,char Port)
Input parameters	1-PinNum >> pin numper from 0 to 7 2-port >>port from portA to port F
Output parameters	none
return	void
Description	Turn of the buzzer

BUZZER AND LIGHTS ENUMS:

Name	PinNum
prototype	Typedef enum PinNum range 0 to 7
Description	Decimal number for pin_number

Name	PORT
prototype	Typedef enum PORT rnge PORTA to PORTF
Description	Decimal number for port