AUTOMOTIVE DOOR CONTROL SYSTEM DESIGN

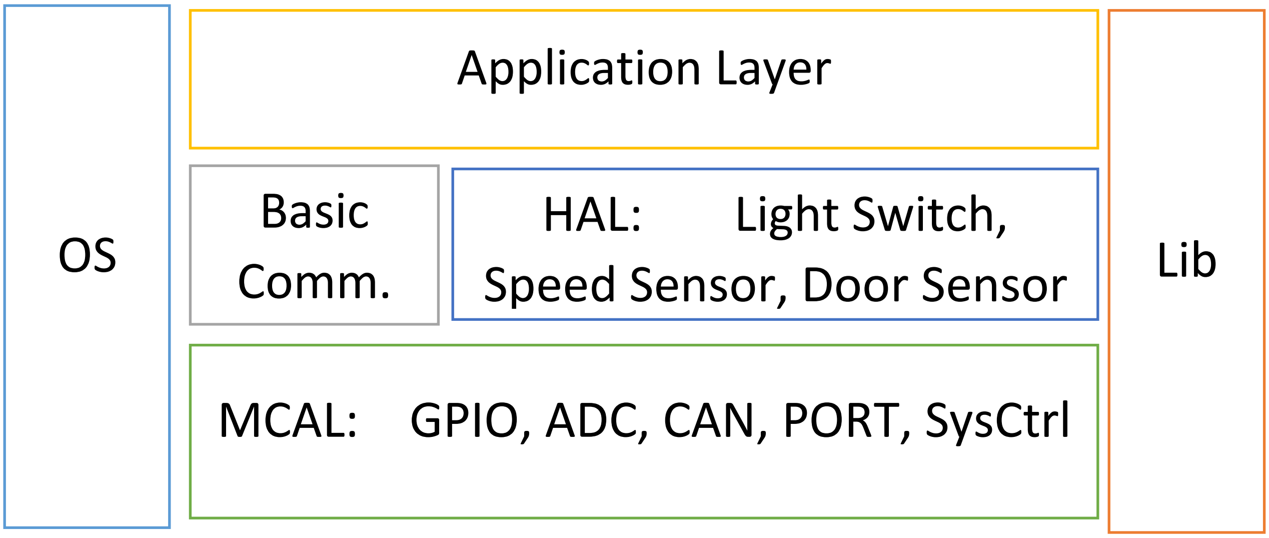
Static Design

Shiry Ezzat Hakim

shiry.hakim@gmail.com

ECU 1

Layered Architecture:



OS

Application Layer

Basic Comm.

HAL: Light Switch, Speed Sensor, Door Sensor

MCAL: GPIO, ADC, CAN, PORT, SysCtrl

Lib

ECU Components:

- Door Sensor
- Speed Sensor
- Light Switch

ECU Modules:

Service Layer	**MCAL**	**HAL**
OS	GPIO	Light Switch
Basic Comm.	ADC	Speed Sensor
	CAN	Door Sensor
	PORT	
	SysCtrl	

APIs:

Port Module:

Function Name: void PORT_Init (const Port_ConfigType * Port_ConfigArray)

Arguments: Port_ConfigArray

Type: Pointer to Port_ConfigType
Port_ConfigType is an unsigned char

Description: Specifies each pin configuration

Output: None

Return: None

Synchronous: Yes

Reentrant: Yes

Description: This function Initializes each Pin with its desired functionality

SysCtrl Module:

Function Name: void SysCtrl_MCInit (void)

Arguments: Macros of SysCtrl_Confg.h

Description: Specifies Microcontroller clock configuration

Output: None

Return: None

Synchronous: Yes

Reentrant: Yes

Description: This function Initializes necessary configurations for Microcontroller such as system clock, peripherals configurations

General Purpose Input Output Module:

Function Name: GPIO_LevelType GPIO_ReadChannel
(GPIO_ChannelTypeChannelId);

Arguments: ChannelId

Type: GPIO_ChannelType

Range: 0-Number of GPIO Channels

Description: Indicates which GPIO channel to read from

Type: GPIO_LevelType (An enum representing High/Low levels)

Range: 0-1

Description: Indicates GPIO channel current level

Return: GPIO_LevelType

Synchronous: Yes

Reentrant: No

Description: This function receives input level from specified Pin

GPIO_ChannelType: Specifies which channel to read from

GPIO_LevelType: Specifies channel level (High/Low)

ADC Module:

Function Name: void ADC_Init(void);

Range: Data sheet dependent

Description: Specifies ADC configurations

Return: None

Synchronous: Yes

Reentrant: Yes

Description: This function Initializes necessary configurations for ADC Converter

Module

Function Name: uint8_t ADC_StartConversion(ADC_ChannelType ChannelId);

Arguments: ChannelId

Type: ADC_ChannelType

Range: 0-Number of ADC channels

Description: Indicates which ADC channel to read from

Type: uint8_t

Range: 0-255

Description: Converted Digital Data

Synchronous: Yes

Reentrant: No

Description: This function receives input level from specified Pin

ADC_ChannelType: Specifies which channel to read signal from

CAN Module:

Function Name: void CAN_Init(void);

Description: CAN1 Module Configurations

Return: None

Synchronous: Yes

Reentrant: Yes

Description: This function Initializes necessary configurations for CAN Module

Function Name: void CAN_TransmitMessage(void);

Input: Passed by writing over TxMailBox

Type: uint8_t

Range: 0-255

Description: Send MSG over CAN

Return: None

Synchronous: Yes

Reentrant: No

Light Switch Module:

Function Name: LightSwitch_StateType LightSwitch_GetState(void);

Type: LightSwitch_StateType (High/Low)

Range: 0-1

Description: Light Switch Current state

Return: LightSwitch_StateType

Synchronous: Yes

Reentrant: Yes

Description: This function gets the current light switch state

LightSwitch_StateType: Specifies switch level (HIGH/LOW)

Speed Sensor Module:

Function Name: uint32_t SpeedSens_getSpeed(void);

Type: uint32_t

Range: 0-4294967295

Description: Speed Sensor Current value

Return: uint32_t

Synchronous: Yes

Reentrant: No

Description: This function gets the digital form of a speed sensor

Door Sensor Module:

Function Name: DoorSens_StateType DoorSens_getState(void);

Type: DoorSens_StateType (Open/Closed)

Range: 0-1

Description: Door Current state

Return: DoorSens_StateType

Synchronous: Yes

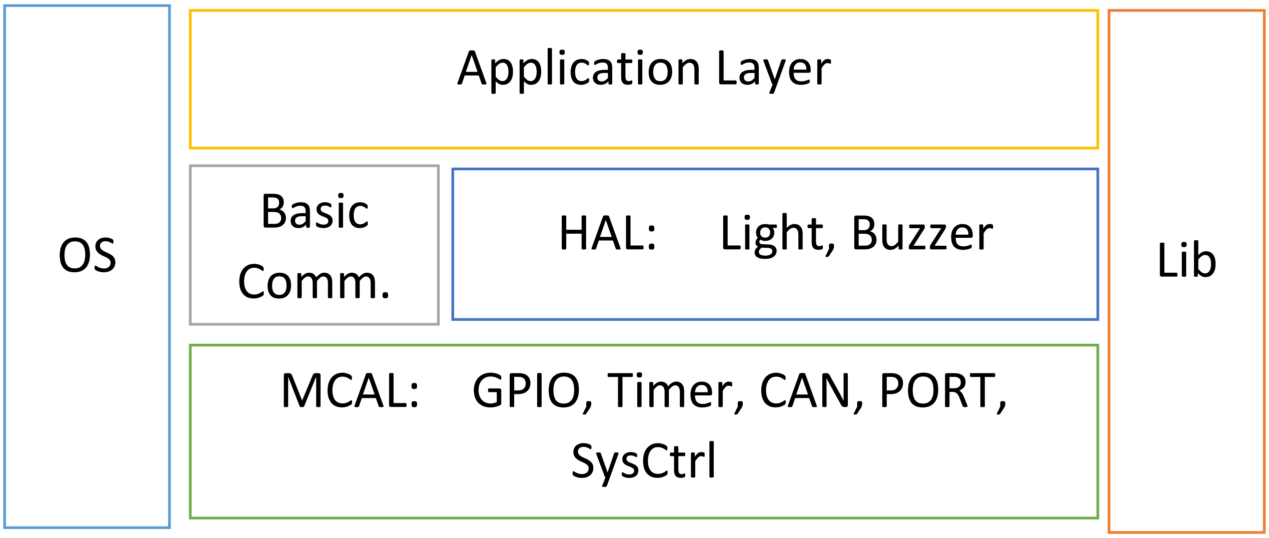
Reentrant: No

Description: This function gets the current light switch state

DoorSens_StateType: Specifies Door state (Open/Closed)

ECU 2

Layered Architecture:



ECU Components:

- Left Light
- Right Light
- Buzzer

ECU Modules:

Service Layer

OS
Basic Comm.

MCAL

GPIO
ADC
CAN
PORT
SysCtrl

HAL

Light Module
Buzzer Module

APIs:

General Purpose Timers Module:

Function Name: void GPT_Init (Gpt_ConfigType * GPT_ConfigArray)

Arguments: GPT_ConfigArray

Type: Array of Gpt_ConfigType

Gpt_ConfigType: is a structure which represents each pin name and Config.

Description: Specifies each GPT channel configuration

Return: None

Synchronous: Yes

Reentrant: No

Description: This function initializes the microcontroller timer with desired configurations

Gpt_ConfigType: Contains configurations associated with timers such as (Channel Id , Channel Mode , Channel Tick Frequency , etc..)

Function Name: void GPT_StartTimer(Gpt_ChannelType Channel, Gpt_ValueType Counts);

Arguments: Channel , Ticks

Range: 0-Number of GPT Channels

Description: Specifies which GPT channel to start

Type: Gpt_ValueType (uint32_t)

Range: 0-4294967295

Description: Specifies the number of ticks desired

Return: None

Synchronous: Yes

Reentrant: No

Description: This function starts the specified timer with desired number of ticks

Gpt_ChannelType: Contains all the channel IDs

Gpt_ValueType: uint8_tFunction Name: void GPT_StopTimer(Gpt_ChannelType Channel);

Arguments: Channel

Type: Gpt_ChannelType

Range: 0-Number of GPT Channels

Description: Specifies which GPT channel to stop

Return: None

Synchronous: Yes

Reentrant: No

Description: This function stops the specified timer with

Gpt_ChannelType: Contains all the channel IDs

Function Name: void GPT_nSecondsDelay (Gpt_ChannelType Channel ,

Gpt_ValueType TimeInSec);

Arguments: Channel

Type: Gpt_ChannelType

Range: 0-Number of GPT Channels

Description: Specifies which GPT channel to start

Input: TimeInSec

Type: Gpt_ValueType (uint32_t)

Range: 0-4294967295

Description: Specifies the number of ticks desired

Return: None

Synchronous: Yes

Reentrant: No

Description: This function is a busy wait implementation for the desired number of seconds

Gpt_ChannelType: Contains all the channel IDs

General Purpose Input Output Module:

Function Name: void GPIO_WriteChannel (GPIO_ChannelType ChannelId, GPIO_LevelType Level)

Arguments: ChannelId

Type: GPIO_ChannelType

Range: 0-Number of GPIO Channels

Description: Specifies which GPIO channel to write over

Type: GPIO_LevelType (High/Low)

Range: 0-1

Description: Sets GPIO Channel level

Return: None

Synchronous: Yes

Reentrant: Yes

GPIO_ChannelType: Specifies which channel to write over

GPIO_LevelType: Specifies desired level (High/Low)

CAN Module:

Function Name: uint8_t CAN1_ReceiveMessage(void);

Output: uint8_t

Range: 0-255

Description: Receive Data from CAN

Synchronous: Yes

Reentrant: No

Buzzer Module:

Function Name: void Buzz_SetBuzzerON(void);

Arguments: None

Return: None

Synchronous: Yes

Reentrant: No

Description: This function Turns the buzzer on

Function Name: void Buzz_SetBuzzerOFF(void);

Arguments: None

Return: None

Synchronous: Yes

Reentrant: No

Description: This function Turns the buzzer off

Lights Module

Function Name: void Lights_SetLightsON(void);

Arguments: None

Return: None

Synchronous: Yes

Reentrant: No

Description: This function Turns the Lights on

Function Name: void Lights_SetLightsOFF(void);

Arguments: None

Return: None

Synchronous: Yes

Reentrant: No

Description: This function Turns the Lights Off