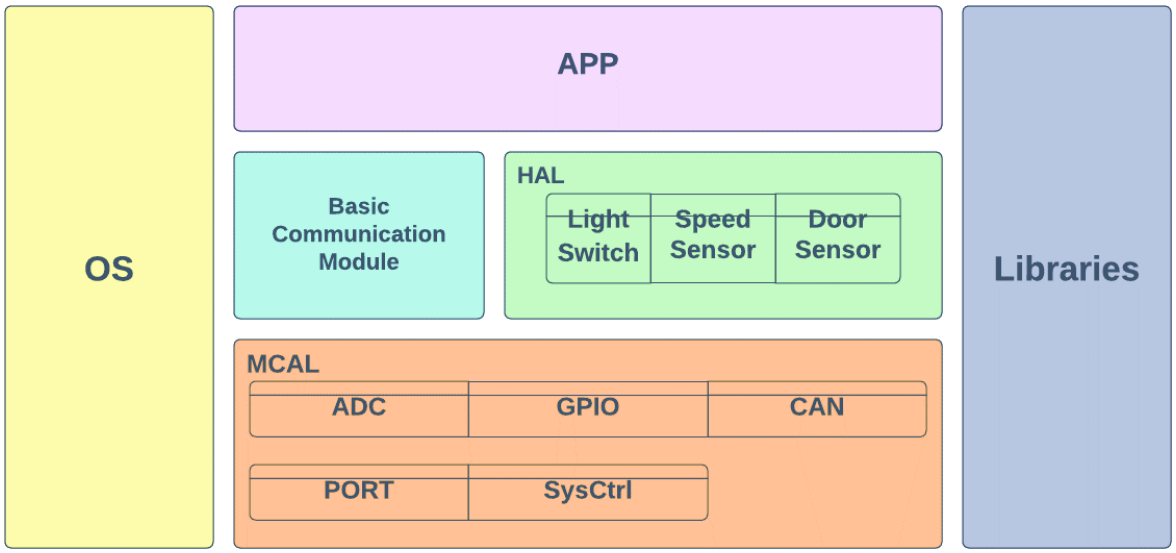
Automotive Door Control System Design

Mohamed Etewa Abdelrazik

Static design analysis

I. ECU 1

a-Layered Architecture:



B-ECU 1 Components:

1) Door Sensor

2) Light Switch

3) Speed Sensor

C-ECU 1 Modules:

MCAL Layer	HAL Layer
1) General Purpose Input Output Module 2) Analog-to-Digital Converter 3) Controller Area Network Module 4) Port Module 5) System Control Module	1) Light Switch Module 2) Speed Sensor Module 3) Door Sensor Module
Service Layer	
1) Operating System	2) Basic Communication Module

D- ECU 1 APIs:

Port Module:

Function Name:		void PORT_Init (const Port_ConfigType * Port_ConfigArray)
Arguments :	Input:	Name : Port_ConfigArray
		Type : Pointer to Port_ConfigType Port_ConfigType is an unsigned char
		Range : Array size is hardware dependent as each element represents a pin Each element range is hardware dependent as well. We may assume 0-15 (the number of possible functionalities) as an example for illustration.
		Macros : which represent each pin possible functionalities according to data sheet For ex : PA0_DIO , PA1_GPT , PA2_ADC , PA3_CAN_TX , etc ..
		Description : Specifies each pin configuration
	Output:	None
Return :		None
Synchronous: Yes		Reentrant: Yes
Description:	This function sets Initializes each Pin with its desired functionality	

SysCtrl Module:

Function Name:		void SysCtrl_MicrocontrollerInit (void)
Arguments :	Input:	Macros from SysCtrl_Configure.h header file
		Range : each configuration Macro has a range which is data sheet dependent
		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_ChannelType ChannelId);
Arguments :	Input:	Name : ChannelId
		Type : GPIO_ChannelType (An enum of microcontroller GPIO channels)
		Range : 0-Number of GPIO Channels (Hardware dependent)
		Variable / Macro : Macro
		Description : Indicates which GPIO channel to read from
	Output:	Type : GPIO_LevelType (An enum representing High/Low levels)
		Range : 0-1
		Variable / Macro : Variable
		Description : Indicates GPIO channel current level
Return :		GPIO_LevelType
Synchronous: Yes		Reentrant: No
Description:	This function receives input level from specified Pin Used typedefs GPIO_ChannelType : Specifies which channel to read from GPIO_LevelType : Specifies channel level (High/Low)	

ADC Module:

Function Name:		void ADC_Init(void);
Arguments :	Input:	Macros from ADC_Configure.h header file
		Range : each configuration Macro has a range which is data sheet dependent
		Description : Specifies ADC configurations
	Output:	None
Return :		None
Synchronous: Yes		Reentrant: Yes
Description:	This function Initializes necessary configurations for Analog-to-Digital Converter Module	

Function Name:		u8 ADC_StartConversion(ADC_ChannelType ChannelId);
Arguments :	Input:	Name : ChannelId
		Type : ADC_ChannelType
		Range : 0-Number of ADC channels (HW Dependent)
		Variable / Macro : Macro
		Description : Indicates which ADC channel to read from
	Output:	Type : unsigned char (u8)
		Range : 0-255
		Variable / Macro : Variable
		Description : Converted Digital Data
Return :		u8
Synchronous: Yes		Reentrant: No
Description:	This function receives input level from specified Pin Used typedefs ADC_ChannelType : Specifies which channel to read signal from	

CAN Module:

Function Name:		void CAN1_Init(void);
Arguments :	Input:	Range : each configuration has a different range
		Variable / Macro : Macros
		Description : CAN1 Module Configurations
	Output:	None
Return :		None
Synchronous: Yes		Reentrant: Yes
Description:	This function Initializes necessary configurations for CAN Module	

Function Name:		void CAN1_TransmitMessage(void);
Arguments :	Input:	Passed by writing over TxMailBox
		Type : unsigned char
		Range : 0-255
		Variable / Macro : Variable
		Description : Message content
	Output :	None
Return :		None
Synchronous: Yes		Reentrant: No
Description:	This function Transmits a message to CAN Transceiver	

Light Switch Module:

Function Name:		LightSwitch_StateType LightSwitch_getState(void);
Arguments :	Input:	None
	Output:	Name : -
		Type : LightSwitch_StateType (High/Low)
		Range : 0-1
		Variable / Macro : Variable
		Description : Light Switch Current state

Return :	LightSwitch_StateType
Synchronous: Yes	Reentrant: Yes
Description:	This function gets the current light switch state Used Typedefs LightSwitch_StateType : Specifies switch level (ON/OFF)

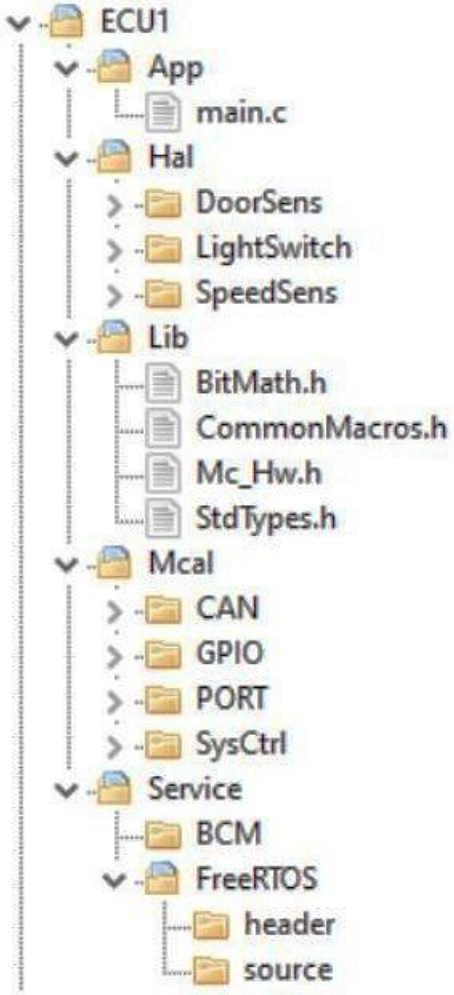
Speed Sensor Module:

Function Name:		u32 SpeedSens_getSpeed(void);
Arguments :	Input:	None
	Output:	Name : -
		Type : unsigned integer
		Range : 0-4294967295
		Variable / Macro : Variable
		Description : Speed Sensor Current value
Return :		u32
Synchronous: Yes		Reentrant: No
Description:		This function gets the digital form of a speed sensor

Door Sensor Module:

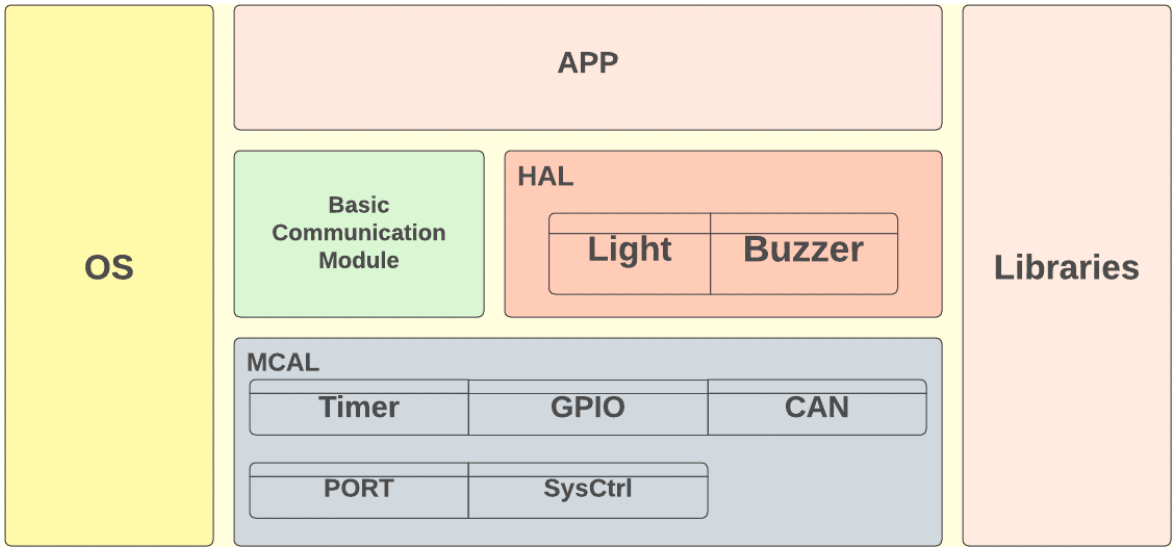
Function Name:		DoorSens_StateType DoorSens_getState(void);
Arguments :	Input:	None
	Output:	Name : -
		Type : DoorSens_StateType (Open/Closed)
		Range : 0-1
		Variable / Macro : Variable
		Description : Door Current state
Return :		DoorSens_StateType
Synchronous: Yes		Reentrant: No
Description:		This function gets the current light switch state Used Typedefs: DoorSens_StateType : Specifies Door state (Open/Closed)

E-Folder Structure:



II. ECU 2

A-Layered Architecture:



B-ECU 2 Components:

1) Right Light

2) Left Light

3) Buzzer

C-ECU 2 Modules:

MCAL Layer	HAL Layer
1) General Purpose Input Output Module 2) General Purpose Timers Module 3) Controller Area Network Module 4) Port Module 5) System Control Module	1) Lights Module 2) Buzzer Module
Service Layer	
1) Operating System	2) Basic Communication Module

D-APIs:

- **Port Module:** void PORT_Init (const u8 PinConfig)
- **SysCtrl Module:** void SysCtrl_MicrocontrollerInit (void)
- **General Purpose Input Output Module:** GPIO_LevelType GPIO_ReadChannel(GPIO_ChannelType ChannelId);
- **CAN Module:** void CAN1_Init(void)

General Purpose Timers Module:

Function Name:		**void GPT_Init (Gpt_ConfigType * GPT_ConfigArray)**
Arguments :	**Input:**	**Name : GPT_ConfigArray**
		Type : Array of Gpt_ConfigType Gpt_ConfigType is a structure which represents each pin name and configurations
		Range : Array size is hardware dependant as each element represents a GPT channel.
		Range : 0-4294967295
		Macros : which represent each channel configurations
		Description : Specifies each GPT channel configuration
	Output:	**None**
Return :		**Void**
Synchronous: Yes		**Reentrant: No**
Description:	**This function initializes the microcontroller timer with desired configurations Used typedefs 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 :	Input:	Name : Channel
		Type : Gpt_ChannelType
		Range : 0-Number of GPT Channels (HW dependant)
		Variable / Macro : Macro
		Description : Specifies which GPT channel to start
	Input:	Name: Ticks
		Type : Gpt_ValueType (unsigned integer)
		Range : 0-4294967295
		Variable / Macro : Variable
		Description : Specifies the number of ticks desired
	Output:	None
Return :		Void
Synchronous: Yes		Reentrant: No
Description:	This function starts the specified timer with desired number of ticks Used typedefs Gpt_ChannelType : Contains all the channel IDs Gpt_ValueType : unsigned integer	

Function Name:		void GPT_StopTimer(Gpt_ChannelType Channel);
Arguments :	Input:	Name : Channel
		Type : Gpt_ChannelType
		Range : 0-Number of GPT Channels (HW dependant)
		Variable / Macro : Macro
		Description : Specifies which GPT channel to stop
	Output:	None
Return :		Void
Synchronous: Yes		Reentrant: No
Description:	This function stops the specified timer with Used typedefs Gpt_ChannelType : Contains all the channel IDs	

General Purpose Input Output Module:

Function Name:		void GPIO_WriteChannel (GPIO_ChannelType ChannelId, GPIO_LevelType Level)
Arguments :	Input:	Name : ChannelId
		Type : Gpt_ChannelType
		Range : 0-Number of GPT Channels (HW dependant)
		Variable / Macro : Macro
		Description : Specifies which GPIO channel to write over
	Output:	Name: Level
		Type : GPIO_LevelType (High/Low)
		Range : 0-1
		Variable / Macro : Variable
		Description : Sets GPIO Channel level
Return :		Void
Synchronous: Yes		Reentrant: Yes
Description:	This function sets specified Output Pin value as desired Used typedefs GPIO_ChannelType : Specifies which channel to write over GPIO_LevelType : Specifies desired level (High/Low)	

CAN Module:

Function Name:		U8 CAN1_ReceiveMessage(void);
Arguments :	Input:	None
	Output:	Name : -
		Type : unsigned char (U8)
		Range : 0-255
		Variable / Macro : Variable
		Description : Received Data
Return :		U8
Synchronous: Yes		Reentrant: No
Description:	This function Receives a message from CAN Transceiver	

Buzzer Module:

Function Name:		void Buzzer_SetBuzzerON(void);
Arguments :	Input:	None
	Output:	None
Return :		None
Synchronous: Yes		Reentrant: No
Description:	This function Turns the buzzer on	

Function Name:		void Buzzer_SetBuzzerOFF(void);
Arguments :	Input:	None
	Output:	None
Return :		None
Synchronous: Yes		Reentrant: No
Description:	This function Turns the buzzer off	

Lights Module:

Function Name:		void Light_SetLightON(void);
Arguments :	Input:	None
	Output:	None
Return :		None
Synchronous: Yes		Reentrant: No
Description:	This function Turns the Lights on	

Function Name:		void Lights_SetLightsOFF(void);
Arguments :	Input:	None
	Output:	None
Return :		None
Synchronous: Yes		Reentrant: No
Description:	This function Turns the Lightsoff	

E-Folder Structure: