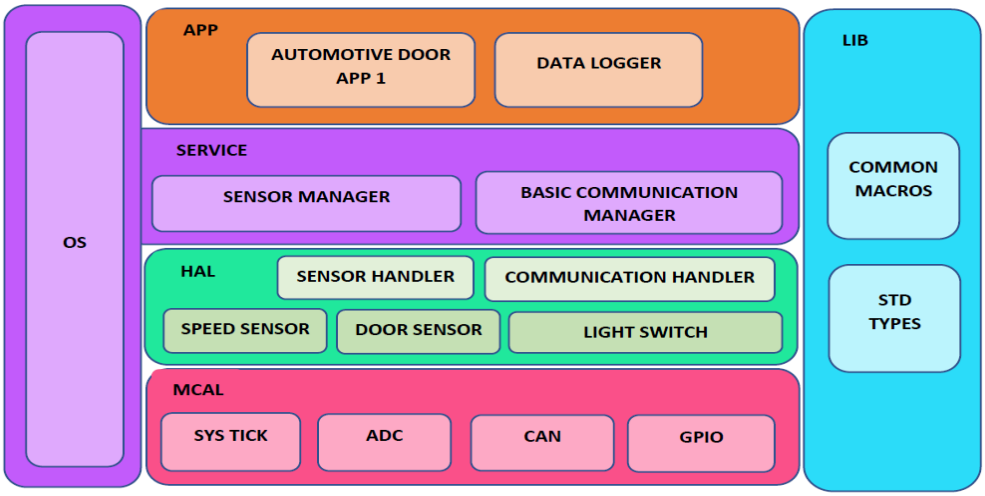
Static Design Analysis:

ECU 1:

1- Layered Architecture with ECU modules and components:



2- APIs and typedefs:

GPIO module:

API	`ERROR_STATE GPIO_init (void);`		
Description	Initialize the GPIO with the required configuration		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	ERROR-STATE

API	`void GPIO_write (uint32 a_pinId, uint8 a_value);`		
Description	Write the required GPIO pin with the required value		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Pin number – pin value	**Return**	None

API	`STD_VALUE GPIO_read (uint32 a_pinId);`		
Description	Write the required GPIO pin with the required value		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Pin number	**Return**	STD_VALUE

ADC Module :

API	`void ADC_init (void);`		
Description	Initialize the ADC with the required configuration		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Pin number - pin value	**Return**	None

API	`uint32 ADC_readChannel (uint8 a_chId);`		
Description	Write the required GPIO pin with the required value		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	channel number	**Return**	uint32

CAN Module :

API	`ERROR_STATE CAN_init (void);`		
Description	Initialize CAN bus with the required configuration		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	ERROR_STATE

API	`void CAN_transmit (uint8 a_canPinId, uint64 a_message);`		
Description	Initialize CAN bus with the required configuration		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Can Pin number - Message	**Return**	None

Speed Sensor Module:

API	`ERROR_STATE SpeedSensor_init (void);`		
Description	Initialize the speed sensor pin via ADC		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	ERROR_STATE

API	Uint16 SpeedSensor_getSpeed (void);		
Description	Get the speed from the speed sensor via ADC		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	Car speed

Door Sensor Module:

API	ERROR_STATE DoorSensor_init (void);		
Description	Initialize the door sensor pin via GPIO		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	ERROR_STATE

API	uint8 DoorSensor_getStatus (void);		
Description	Initialize the door sensor pin via GPIO		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	uint8

Light Switch Module:

API	ERROR_STATE LightSwitch_init (void);		
Description	Initialize the door sensor pin via GPIO		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	ERROR_STATE

API	uint8 LightSwitch_getStatus (void);		
Description	Read the light swich status		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	uint8

Sensor handler Module:

API	`uint32 Sensor_handler (uint8 a_sensorId);`		
Description	choose which sensor to read directly from hardware		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Sensor name	**Return**	Sensor value

Communication handler module:

API	`void BCM_handler (uint64 a_handlerMessage, uint8 a_bus);`		
Description	Choose which bus to send the message to and send it to hardware directly		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Message – the bus sent to	**Return**	None

Sensor manager Module:

API	`uint32 Sensor_manager (uint8 a_sensorId);`		
Description	Allow the application to choose which sensor to get the reading from		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Sensor name	**Return**	Sensor value

Basic Communication manager Module:

API	`Void BCM_mananger (uint64 a_ManagerMessage, uint8 a_bus);`		
Description	Allow the application to choose which bus to send the message to		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Message – the bus sent to	**Return**	None

Data logger Module:

API	`void DataLogger_saveData (uint64 a_data);`		
Description	Save the required data sent to it		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Data to be saved	**Return**	None

Automotive Door App1 Module :

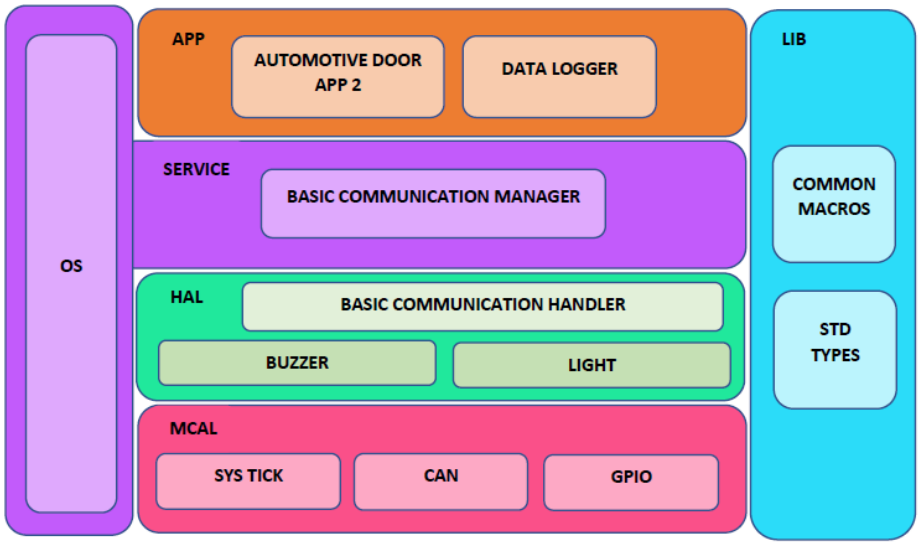
API	`void SendDoorState_task (void);`		
Description	Send the door sensor state to ECU2 via CAN bus		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	None

API	`void SendSpeed_task (void);`		
Description	Send the speed sensor value to ECU2 via CAN bus		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	None

API	`void SendLightSwitchState_task (void);`		
Description	Send the light switch state to ECU2 via CAN bus		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	None

ECU 2:

1- Layered Architecture with ECU modules and components:



2- APIs and typedefs:

GPIO module:

API	ERROR_STATE GPIO_init (void);		
Description	Initialize the GPIO with the required configuration		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	ERROR-STATE

API	void GPIO_write (uint16 a_pinId, uint8 a_value);		
Description	Write the required GPIO pin with the required value		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Pin number – pin value	**Return**	None

API	STD_VALUE GPIO_read (uint16 a_pinId);		
Description	Write the required GPIO pin with the required value		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Pin number	**Return**	STD_VALUE

CAN Module :

API	ERROR_STATE CAN_init (void);		
Description	Initialize CAN bus with the required configuration		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	ERROR_STATE

API	Uint64 CAN_receive (uint8 a_canPinId);		
Description	Initialize CAN bus with the required configuration		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Can Pin number	**Return**	message

Buzzer Module:

API	`ERROR_STATE BUZZER_init (void);`		
Description	Initialize the buzzer via GPIO		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	ERROR_STATE

API	`void BUZZER_on (void);`		
Description	Set the buzzer on via GPIO		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	None

API	`void BUZZER_off (void);`		
Description	Set the buzzer off via GPIO		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	None

Communication handler module:

API	`uint64 BCM_handler (uint8 a_bus);`		
Description	Choose which bus to read the message from, directly from hardware		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	the bus to read from	**Return**	message

Basic Communication manager Module:

API	`uint64 BCM_mananger (uint8 a_bus);`		
Description	Allow the application to choose which bus to read the message from		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	the bus to read from	**Return**	message

Light Module:

API	ERROR_STATE LIGHT_init (void);		
Description	Initialize the light via GPIO		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	ERROR_STATE

API	void LIGHT_on (void);		
Description	Set the light on via GPIO		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	None

API	void LIGHT_off (void);		
Description	Set the light off via GPIO		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	None

Data logger Module:

API	void DataLogger_saveData (uint64 a_data);		
Description	Save the required data sent to it		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	Data to be saved	**Return**	None

Automotive Door App2 Module:

API	void ReceiveMessage_task (void);		
Description	Receive the message periodically to take actions		
Sync/Async	Synchronous	**Reentrancy**	Non-reentrant
Parameters	None	**Return**	None

- typedef unsigned long uint32
- typedef unsigned short uint16
- typedef unsigned char uint8
- typedef unsigned long long uint64