Automotive Door Control System DesignPart 1Static DesignName: Mohamed Abdelnasser Mehery

ECU1:

ECU1 has 7 modules.



ECU1 will send status messages periodically to ECU 2 through the CAN protocol, Status messages will be sent using Basic Communication Module (BCM) module, Door state message will be sent every 10ms to ECU 2, Light switch state message will be sent every 20ms to ECU 2, Speed state message will be sent every 5ms to ECU 2.

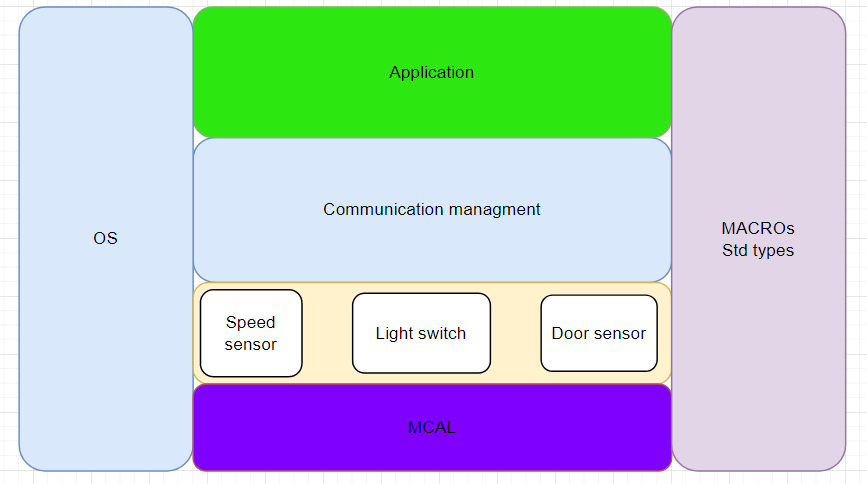
ECU2:



ECU 2 has 6 modules.

If the door is opened while the car is moving → Buzzer ON, Lights OFF, If the door is opened while the car is   
stopped → Buzzer OFF, Lights ON, If the door is closed while the lights were ON → Lights are OFF after 3 seconds  
If the car is moving and the light switch is pressed → Buzzer OFF, Lights ON, If the car is stopped and the light switch is pressed → Buzzer ON, Lights ON.

ECU1:

  
  
**DIO APIs:**

|  |  |
| --- | --- |
| **Function Name** | DIO\_Init() |
| **API Type** | Init |
| **Parameters (INPUT)** | DIO\_Port |
| DIO\_Channel |
| DIO\_PinLevel |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK: 0 |
| E\_NOK: 1 |
| **Description** | initialization the Dio module |

|  |  |
| --- | --- |
| **Function Name** | DIO\_Read() |
| **API Type** | Getter |
| **Parameters (INPUT)** | DIO\_Port |
| DIO\_Channel |
| **Parameters (OUTPUT)** | DIO\_PinLevel |
| **Return** | E\_OK: 0 |
| E\_NOK: 1 |
| **Description** | Reading the value of the channel |
| **Function Name** | DIO\_Write() |
| **API Type** | Setter |
| **Parameters (INPUT)** | DIO\_Port |
|  | DIO\_Channel |
| **Parameters (OUTPUT)** | DIO\_PinLevel |
| **Return** | E\_OK: 0 |
|  | E\_NOK: 1 |
| **Description** | Write on the channel low or high |

|  |  |
| --- | --- |
| Name | **DIO\_Port** |
| Type | typedef enum |
| Range | {Port A to PortF } |
| Description | The decimal number for Port |

|  |  |
| --- | --- |
| Name | **DIO\_Channel** |
| Type | typedef enum |
| Range | { PIN0 to PIN7} |
| Description | The decimal number for Pin |

|  |  |
| --- | --- |
| Name | **DIO\_PinLevel** |
| Type | typedef enum |
| Range | 0 | Low or Input Direction |
| 1 | High or Output Direction |
| Description | The direction of the channel or the level on it. | |

**Timer APIs:**

|  |  |
| --- | --- |
| **Function Name** | TIMER\_Inti() |
| **API Type** | Init |
| **Parameters (INPUTS)** | \* ConfigPtr | TIMER\_ConfigType |
| **Parameters (OUTPUT)** | None |  |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | initialization the timer module |  |

**for ECU 1 & ECU 2**

|  |  |
| --- | --- |
| **Function Name** | TIMER\_Start() |
| **API Type** | - |
| **Parameters (INPUTS)** | Channel | TIMER\_ChannelType |
| Value | TIMER\_ValueType |
| **Parameters (OUTPUT)** | None |  |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Start the timer channel |  |

|  |  |
| --- | --- |
| **Function Name** | TIMER\_Stop() |
| **API Type** | - |
| **Parameters (INPUTS)** | Channel | TIMER\_ChannelType |
| **Parameters (OUTPUT)** | None |  |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Stop the timer channel | |

|  |  |
| --- | --- |
| Name | **TIMER\_ChannelType** |
| Type | Uint8\_t |
| Description | The channel of the timer |

|  |  |
| --- | --- |
| Name | **TIMER\_ValueType** |
| Type | Uint8\_t |
| Description | Type for reading and setting the timer value number of ticks |

|  |  |
| --- | --- |
| Name | **TIMER\_ConfigType** |
| Type | Structure |
| Description | This structure is including the configuration |

**ADC APIs:**

|  |  |
| --- | --- |
| **Function Name** | ADC\_Init() |
| **API Type** | Init |
| **Parameters (INPUTS)** | \* ConfigPtr | ADC\_ConfigType |
| **Parameters (OUTPUT)** | None |  |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | initialization the ADC module | |

|  |  |
| --- | --- |
| **Function Name** | ADC\_Read () |
| **API Type** | Init |
| **Parameters (INPUTS)** | Channel | ADC\_ChannelType |
| **Parameters (OUTPUT)** | None |  |
|  | E\_OK | 0 |
|  | E\_NOK | 1 |
| **Description** | This API to read the value in ADC registers and return it. | |

|  |  |
| --- | --- |
| Name | **ADC\_ChannelType** |
| Type | Uint8\_t |
| Description | This the data of struct including config of ADC |

|  |  |
| --- | --- |
| Name | **ADC\_ConfigType** |
| Type | structure |

**CAN APIs:**

|  |  |
| --- | --- |
| **Function Name** | CAN\_Init() |
| **API Type** | Init |
| **Parameters (INPUTS)** | \* ConfigPtr | CAN\_ConfigType |
| **Parameters (OUTPUT)** | None |  |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Initializes the CAN Module | |

**for ECU 1 & ECU 2**

|  |  |
| --- | --- |
| **Function Name** | CAN\_Baudrate() |
| **API Type** |  |
| **Parameters (INPUTS)** | Controller | Uint8\_t |
| Baudrate | Uint16\_t |  |
| **Parameters (OUTPUT)** | None |  |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Set the baudrate to CAN Module | |

|  |  |
| --- | --- |
| **Function Name** | CAN\_SendData() |
| **API Type** | - |
| **Parameters (INPUTS)** | Data | Uint32\_t |
| **Parameters (OUTPUT)** | None |  |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Send the data by the CAN Module | |

|  |  |
| --- | --- |
| **Function Name** | CAN\_ReceiveData() |
| **API Type** | Getter |
| **Parameters (INPUTS)** | void |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Receive data from CAN Module | |

|  |  |
| --- | --- |
| Name | **CAN\_ConfigType** |
| Type | structure |
| Description | Thie Structure include the configratioin set required for initializaing the CAN |

**Door Sensor APIs:**

|  |  |
| --- | --- |
| **Function Name** | DoorSen\_Init() |
| **API Type** | Init |
| **Parameters (INPUTS)** | None |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Initializes the door sensor module | |

|  |  |
| --- | --- |
| **Function Name** | DoorSen\_ReadValue() |
| **API Type** | Getter |
| **Parameters (INPUTS)** | None |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Get the state of door sensor module | |

**Light Switch APIs:**

|  |  |
| --- | --- |
| **Function Name** | LightSW\_Init() |
| **API Type** | Init |
| **Parameters (INPUTS)** | None |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |  |
| **Description** | Initializes the Light Switch module | |

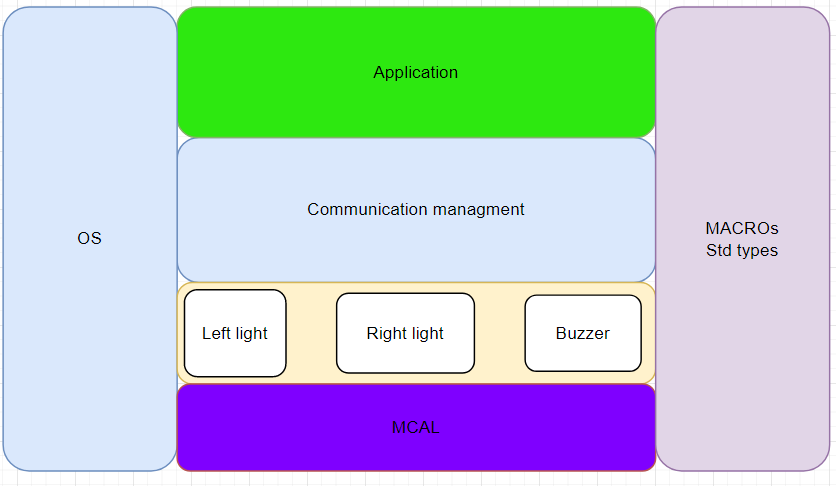
|  |  |
| --- | --- |
| **Function Name** | LightSW\_ReadValue() |
| **API Type** | Init |
| **Parameters (INPUTS)** | None |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |  |
| **Description** | Get the state of Light Switch module | |

**Speed Sensor APIs:**

|  |  |
| --- | --- |
| **Function Name** | SpeedSen\_Init() |
| **API Type** | Init |
| **Parameters (INPUTS)** | None |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Initializes the timer module | |

|  |  |
| --- | --- |
| **Function Name** | SpeedSen\_ReadValue() |
| **API Type** | Init |
| **Parameters (INPUTS)** | None |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Get the state of Speed Sensor module | |

ECU 2 :-



**Light Right(LR) APIs:**

|  |  |
| --- | --- |
| **Function Name** | LR\_Init() |
| **API Type** | - |
| **Parameters (INPUTS)** | DIO\_Port , DIO\_Pin |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Initializes the Light Right |  |

|  |  |
| --- | --- |
| **Function Name** | LR\_ON() |
| **API Type** | - |
| **Parameters (INPUTS)** | DIO\_Port , DIO\_Pin |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | make Light right on | |

|  |  |
| --- | --- |
| **Function Name** | LR\_OFF() |
| **API Type** | - |
| **Parameters (INPUTS)** | DIO\_Port , DIO\_Pin |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Make Light right off | |

**Light Left (LL) APIs:**

|  |  |
| --- | --- |
| **Function Name** | LL\_Init() |
| **API Type** | - |
| **Parameters (INPUTS)** | DIO\_Port , DIO\_Pin |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Initializes the Light lift | |

|  |  |
| --- | --- |
| **Function Name** | LL\_ON() |
| **API Type** | - |
| **Parameters (INPUTS)** | DIO\_Port , DIO\_Pin |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Make Light lift on | |

|  |  |
| --- | --- |
| **Function Name** | LL\_OFF() |
| **API Type** | - |
| **Parameters (INPUTS)** | DIO\_Port , DIO\_Pin |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Make Light lift off | |

**Buzzer (B) APIs:**

|  |  |
| --- | --- |
| **Function Name** | Buzzer\_Init() |
| **API Type** | Init |
| **Parameters (INPUTS)** | DIO\_Port , DIO\_Pin |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Initializes the Buzzer module ( make the pin output ) | |

|  |  |
| --- | --- |
| **Function Name** | Buzzer\_ON() |
| **API Type** | - |
| **Parameters (INPUTS)** | DIO\_Port , DIO\_Pin |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Turn on the buzzer | |

|  |  |
| --- | --- |
| **Function Name** | Buzzer\_OFF() |
| **API Type** | - |
| **Parameters (INPUTS)** | DIO\_Port , DIO\_Pin |
| **Parameters (OUTPUT)** | None |
| **Return** | E\_OK | 0 |
| E\_NOK | 1 |
| **Description** | Turn off the buzzer | |