

|  |
| --- |
| Static Design |
|  |
| February 9  Automotive Door Control System Design  Khaled Ibrahim Abdulaziz |



# 1.ECU1

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1.1 Layered Architecture Making the layered architecture and specifying ECU1 components and modules. 1.2 APIs and Typedefs1.2.1 PORT 1.2.1.1 APIs   |  |  | | --- | --- | | **Name** | Port\_Init | | **Syntax** | void Port\_Init(Port\_ConfigType \*PortConfigPtr) | | **Description** | Initializes the Pin with the configured functionality | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | PortConfigPtr | | **Parameters(out)** | None | | **Return value** | Void |   1.2.1.2 Typedefs   |  |  | | --- | --- | | **Name** | Port\_ConfigType | | **Type** | struct | | **Description** | A struct holds the configurations of the pin including its port, number, mode, level, etc.. |  1.2.2 DIO 1.2.2.1 APIs   |  |  | | --- | --- | | **Name** | Dio\_ReadChannel | | **Syntax** | Dio\_LevelType Dio\_ReadChannel(Dio\_PortType PortId, Dio\_ChannelType ChannelId) | | **Description** | Reads the DIO channel | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | PortId, ChannelId | | **Parameters(out)** | None | | **Return value** | Dio\_LevelType |  |  |  | | --- | --- | | **Name** | Dio\_WriteChannel | | **Syntax** | void Dio\_WriteChannel(Dio\_PortType PortId, Dio\_ChannelType ChannelId, Dio\_LevelType level) | | **Description** | Writes to the DIO channel | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | PortId, ChannelId, level | | **Parameters(out)** | None | | **Return value** | void |   1.2.2.2 Typedefs   |  |  | | --- | --- | | **Name** | Dio\_LevelType | | **Type** | enum | | **Description** | An enum holds the level of the pin which is Low (0) or High (1). |  |  |  | | --- | --- | | **Name** | Dio\_PortType | | **Type** | enum | | **Description** | An enum holds the ports of MCU such as Port\_A, Port\_B, etc.. |  |  |  | | --- | --- | | **Name** | Dio\_ChannelType | | **Type** | enum | | **Description** | An enum holds the pins of the port such as Pin\_0, Pin\_1, etc… |  1.2.3 ADC 1.2.3.1 APIs   |  |  | | --- | --- | | **Name** | ADC\_Init | | **Syntax** | void ADC\_Init(ADC\_ConfigType \*ADCConfigPtr) | | **Description** | Initializes ADC module with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | ADCConfigPtr | | **Parameters(out)** | None | | **Return value** | void |  |  |  | | --- | --- | | **Name** | ADC\_Read | | **Syntax** | void ADC\_ReadChannel(ADC\_ChannelType ChannelId) | | **Description** | Reads the current value of ADC channel | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | ChannelId | | **Parameters(out)** | None | | **Return value** | void |   1.2.3.2 Typedefs   |  |  | | --- | --- | | **Name** | ADCConfigPtr | | **Type** | struct | | **Description** | A struct that holds all ADC configurations. |  |  |  | | --- | --- | | **Name** | ADC\_ChannelType | | **Type** | enum | | **Description** | An enum represents the ADC channels like ADC\_0, ADC\_1, etc.. |  1.2.4 CAN 1.2.4.1 APIs   |  |  | | --- | --- | | **Name** | CAN\_Init | | **Syntax** | void CAN\_Init(CAN\_ConfigType \*CANConfigPtr) | | **Description** | Initializes CAN module with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | CANConfigPtr | | **Parameters(out)** | None | | **Return value** | void |  |  |  | | --- | --- | | **Name** | CAN\_Transmit | | **Syntax** | void CAN\_Transmit(uint32\_t data) | | **Description** | Sends data through CAN bus | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | data | | **Parameters(out)** | None | | **Return value** | void |   1.2.4.2 Typedefs   |  |  | | --- | --- | | **Name** | CAN\_ConfigType | | **Type** | struct | | **Description** | A struct that holds all CAN configurations. |  1.2.5 SysTick 1.2.5.1 APIs   |  |  | | --- | --- | | **Name** | SysTick\_Init | | **Syntax** | void SysTick\_Init(SysTick\_ConfigType SysTickConfigPtr) | | **Description** | Initializes SysTick timer with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | SysTickConfigPtr | | **Parameters(out)** | None | | **Return value** | void |   1.2.5.2 Typedefs   |  |  | | --- | --- | | **Name** | SysTick\_ConfigType | | **Type** | struct | | **Description** | A struct that holds all SysTick configurations. |  1.2.6 Door Sensor 1.2.6.1 APIs   |  |  | | --- | --- | | **Name** | DoorSensor\_Init | | **Syntax** | void DoorSensor\_Init(DS\_ConfigType DSConfig) | | **Description** | Initializes Door Sensor module with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | DSConfig | | **Parameters(out)** | None | | **Return value** | void |  |  |  | | --- | --- | | **Name** | DoorSensor\_GetState | | **Syntax** | DS\_StateType DoorSensor\_ GetState(void) | | **Description** | Gets the current state of Door Sensor | | **Sync/Async** | Synchronous | | **Reentrancy** | Reentrant | | **Parameters(in)** | void | | **Parameters(out)** | None | | **Return value** | DS\_StateType |   1.2.6.2 Typedefs   |  |  | | --- | --- | | **Name** | DSConfig | | **Type** | struct | | **Description** | A struct that holds all Door Sensor configurations. |  |  |  | | --- | --- | | **Name** | DS\_StateType | | **Type** | enum | | **Description** | An enum represents the Door Sensor state either Open (1) or Closed (0). |  1.2.7 Light Switch 1.2.7.1 APIs   |  |  | | --- | --- | | **Name** | LightSwitch\_Init | | **Syntax** | Void LightSwitch\_Init(LS\_ConfigType LSConfig) | | **Description** | Initializes Light Switch module with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Reentrant | | **Parameters(in)** | DSConfig | | **Parameters(out)** | None | | **Return value** | void |  |  |  | | --- | --- | | **Name** | LightSwitch\_GetState | | **Syntax** | LS\_StateType LightSwitch\_ GetState(void) | | **Description** | Gets the current state of Light Switch | | **Sync/Async** | Synchronous | | **Reentrancy** | Reentrant | | **Parameters(in)** | void | | **Parameters(out)** | None | | **Return value** | LS\_StateType |   1.2.7.2 Typedefs   |  |  | | --- | --- | | **Name** | LSConfig | | **Type** | struct | | **Description** | A struct that holds all Light Switch configurations. |  |  |  | | --- | --- | | **Name** | LS\_StateType | | **Type** | enum | | **Description** | An enum represents the Light Switch state either ON (1) or OFF (0). |  1.2.8 Speed Sensor 1.2.8.1 APIs   |  |  | | --- | --- | | **Name** | SpeedSensor\_Init | | **Syntax** | void SpeedSensor\_Init(SS\_ConfigType SSConfig) | | **Description** | Initializes Speed Sensor module with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Reentrant | | **Parameters(in)** | SSConfig | | **Parameters(out)** | None | | **Return value** | void |  |  |  | | --- | --- | | **Name** | SpeedSensor\_GetState | | **Syntax** | uint16\_t SpeedSensor\_ GetValue(void) | | **Description** | Gets the current value of Speed Sensor Module | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | void | | **Parameters(out)** | None | | **Return value** | uint16\_t |   1.2.8.2 Typedefs   |  |  | | --- | --- | | **Name** | SSConfig | | **Type** | struct | | **Description** | A struct that holds all Speed Sensor configurations. |  1.2.9 BCM 1.2.9.1 APIs   |  |  | | --- | --- | | **Name** | BCM\_Manager | | **Syntax** | void BCM\_Manager (Sensor\_IdType SensorId) | | **Description** | Manages the transmitted data through CAN Bus | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | SensorId | | **Parameters(out)** | None | | **Return value** | void |   1.2.9.2 Typedefs   |  |  | | --- | --- | | **Name** | Sensor\_IdType | | **Type** | enum | | **Description** | An enum holds the sensor Id. |  1.3 Folder Structure 📦ECU1  ┣ 📂Application  ┃ ┗ 📜main.c  ┣ 📂HAL  ┃ ┣ 📂Include  ┃ ┃ ┣ 📜Door\_Sensor.h  ┃ ┃ ┣ 📜Light\_Switch.h  ┃ ┃ ┗ 📜Speed\_Sensor.h  ┃ ┣ 📜Door\_Sensor.c  ┃ ┣ 📜Light\_Switch.c  ┃ ┗ 📜Speed\_Sensor.c  ┣ 📂Library  ┃ ┣ 📜Bit\_Math.h  ┃ ┣ 📜Platform\_Types.h  ┃ ┗ 📜Std\_Types.h  ┣ 📂MCAL  ┃ ┣ 📂ADC  ┃ ┃ ┣ 📂Config  ┃ ┃ ┃ ┣ 📜ADC\_Cfg.h  ┃ ┃ ┃ ┗ 📜ADC\_Lcfg.c  ┃ ┃ ┣ 📂Include  ┃ ┃ ┃ ┣ 📜ADC.h  ┃ ┃ ┃ ┗ 📜ADC\_types.h  ┃ ┃ ┗ 📜ADC.c  ┃ ┣ 📂CAN  ┃ ┃ ┣ 📂Config  ┃ ┃ ┃ ┣ 📜CAN\_Cfg.h  ┃ ┃ ┃ ┗ 📜CAN\_Lcfg.c  ┃ ┃ ┣ 📂Include  ┃ ┃ ┃ ┗ 📜CAN.h  ┃ ┃ ┗ 📜CAN.c  ┃ ┣ 📂DIO  ┃ ┃ ┣ 📂Include  ┃ ┃ ┃ ┣ 📜DIO.h  ┃ ┃ ┃ ┗ 📜DIO\_Types.h  ┃ ┃ ┗ 📜DIO.c  ┃ ┣ 📂PORT  ┃ ┃ ┣ 📂Config  ┃ ┃ ┃ ┣ 📜Port\_Cfg.h  ┃ ┃ ┃ ┗ 📜Port\_Lcfg.c  ┃ ┃ ┣ 📂Include  ┃ ┃ ┃ ┣ 📜Port.h  ┃ ┃ ┃ ┗ 📜Port\_Types.h  ┃ ┃ ┗ 📜Port.c  ┃ ┗ 📂SysTick  ┃ ┃ ┣ 📂Config  ┃ ┃ ┃ ┣ 📜SysTick\_Cfg.h  ┃ ┃ ┃ ┗ 📜SysTick\_Lcfg.c  ┃ ┃ ┣ 📂Include  ┃ ┃ ┃ ┣ 📜SysTick.h  ┃ ┃ ┃ ┗ 📜SysTick\_Types.h  ┃ ┃ ┗ 📜SysTick.c  ┗ 📂Service  ┃ ┣ 📂Include  ┃ ┃ ┣ 📜BCM.h  ┃ ┃ ┗ 📜OS.h  ┃ ┣ 📜BCM.c  ┃ ┗ 📜OS.c |

# 2.ECU2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2.1 Layered Architecture Making the layered architecture and specifying ECU2 components and modules. 2.2 APIs and Typedefs2.2.1 PORT 2.2.1.1 APIs   |  |  | | --- | --- | | **Name** | Port\_Init | | **Syntax** | void Port\_Init(Port\_ConfigType \*PortConfigPtr) | | **Description** | Initializes the Pin with the configured functionality | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | PortConfigPtr | | **Parameters(out)** | None | | **Return value** | Void |   2.2.1.2 Typedefs   |  |  | | --- | --- | | **Name** | Port\_ConfigType | | **Type** | struct | | **Description** | A struct holds the configurations of the pin including its port, number, mode, level, etc.. |  2.2.2 DIO 2.2.2.1 APIs   |  |  | | --- | --- | | **Name** | Dio\_ReadChannel | | **Syntax** | Dio\_LevelType Dio\_ReadChannel(Dio\_PortType PortId, Dio\_ChannelType ChannelId) | | **Description** | Reads the DIO channel | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | PortId, ChannelId | | **Parameters(out)** | None | | **Return value** | Dio\_LevelType |  |  |  | | --- | --- | | **Name** | Dio\_WriteChannel | | **Syntax** | void Dio\_WriteChannel(Dio\_PortType PortId, Dio\_ChannelType ChannelId, Dio\_LevelType level) | | **Description** | Writes to the DIO channel | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | PortId, ChannelId, level | | **Parameters(out)** | None | | **Return value** | void |   2.2.2.2 Typedefs   |  |  | | --- | --- | | **Name** | Dio\_LevelType | | **Type** | enum | | **Description** | An enum holds the level of the pin which is Low (0) or High (1). |  |  |  | | --- | --- | | **Name** | Dio\_PortType | | **Type** | enum | | **Description** | An enum holds the ports of MCU such as Port\_A, Port\_B, etc.. |  |  |  | | --- | --- | | **Name** | Dio\_ChannelType | | **Type** | enum | | **Description** | An enum holds the pins of the port such as Pin\_0, Pin\_1, etc… |  2.2.3 CAN 2.2.3.1 APIs   |  |  | | --- | --- | | **Name** | CAN\_Init | | **Syntax** | void CAN\_Init(CAN\_ConfigType \*CANConfigPtr) | | **Description** | Initializes CAN module with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | CANConfigPtr | | **Parameters(out)** | None | | **Return value** | void |  |  |  | | --- | --- | | **Name** | CAN\_Receive | | **Syntax** | uint32\_t CAN\_Receive (void) | | **Description** | Receives data through CAN bus | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | void | | **Parameters(out)** | None | | **Return value** | uint32\_t |   2.2.3.2 Typedefs   |  |  | | --- | --- | | **Name** | CAN\_ConfigType | | **Type** | struct | | **Description** | A struct that holds all CAN configurations. |  2.2.4 SysTick 2.2.4.1 APIs   |  |  | | --- | --- | | **Name** | SysTick\_Init | | **Syntax** | void SysTick\_Init(SysTick\_ConfigType SysTickConfigPtr) | | **Description** | Initializes SysTick timer with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | SysTickConfigPtr | | **Parameters(out)** | None | | **Return value** | void |   2.2.4.2 Typedefs   |  |  | | --- | --- | | **Name** | SysTick\_ConfigType | | **Type** | struct | | **Description** | A struct that holds all SysTick configurations. |  2.2.5 Buzzer 2.2.5.1 APIs   |  |  | | --- | --- | | **Name** | Buzzer\_Init | | **Syntax** | void Buzzer\_Init(Buzzer\_ConfigType BuzzerConfig) | | **Description** | Initializes Buzzer module with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | BuzzerConfig | | **Parameters(out)** | None | | **Return value** | void |  |  |  | | --- | --- | | **Name** | Buzzer\_SetState | | **Syntax** | void Buzzer\_ SetState(Buzzer\_StateType Buzzer\_State) | | **Description** | Sets the state of Buzzer | | **Sync/Async** | Synchronous | | **Reentrancy** | Reentrant | | **Parameters(in)** | Buzzer\_State | | **Parameters(out)** | None | | **Return value** | void |   2.2.5.2 Typedefs   |  |  | | --- | --- | | **Name** | BuzzerConfig | | **Type** | struct | | **Description** | A struct that holds all Buzzer configurations. |  |  |  | | --- | --- | | **Name** | Buzzer\_StateType | | **Type** | enum | | **Description** | An enum represents the Buzzer state either ON (1) or OFF (0). |  2.2.6 Light 2.2.6.1 APIs   |  |  | | --- | --- | | **Name** | Lights\_Init | | **Syntax** | Void Lights\_Init(Lights\_ConfigType LightsConfig) | | **Description** | Initializes Lights module with the desired configurations | | **Sync/Async** | Synchronous | | **Reentrancy** | Reentrant | | **Parameters(in)** | LightsConfig | | **Parameters(out)** | None | | **Return value** | void |  |  |  | | --- | --- | | **Name** | Lights\_SetState | | **Syntax** | void Lights\_ SetState(Lights\_SideType Lights\_Side, Lights\_StateType Lights\_State) | | **Description** | Sets the state of Lights by its side | | **Sync/Async** | Synchronous | | **Reentrancy** | Reentrant | | **Parameters(in)** | Lights\_Side, Lights\_State | | **Parameters(out)** | None | | **Return value** | void |   2.2.6.2 Typedefs   |  |  | | --- | --- | | **Name** | LightsConfig | | **Type** | struct | | **Description** | A struct that holds all Light configurations. |  |  |  | | --- | --- | | **Name** | Lights\_SideType | | **Type** | enum | | **Description** | An enum represents the Lights state either Left (0) or Right (1). |  |  |  | | --- | --- | | **Name** | Lights\_StateType | | **Type** | enum | | **Description** | An enum represents the Lights state either ON (1) or OFF (0). |  2.2.7 BCM 2.2.7.1 APIs   |  |  | | --- | --- | | **Name** | BCM\_Manager | | **Syntax** | void BCM\_Manager (Sensor\_IdType \*SensorId, uint32\_t \*data) | | **Description** | Manages the transmitted data through CAN Bus | | **Sync/Async** | Synchronous | | **Reentrancy** | Non-Reentrant | | **Parameters(in)** | SensorId, data | | **Parameters(out)** | None | | **Return value** | void |   2.2.7.2 Typedefs   |  |  | | --- | --- | | **Name** | Sensor\_IdType | | **Type** | enum | | **Description** | An enum holds the sensor Id. |  2.3 Folder Structure 📦ECU2  ┣ 📂Application  ┃ ┗ 📜main.c  ┣ 📂HAL  ┃ ┣ 📂Include  ┃ ┃ ┣ 📜Buzzer.h  ┃ ┃ ┗ 📜Light.h  ┃ ┣ 📜Buzzer.c  ┃ ┗ 📜Lights.c  ┣ 📂Library  ┃ ┣ 📜Bit\_Math.h  ┃ ┣ 📜Platform\_Types.h  ┃ ┗ 📜Std\_Types.h  ┣ 📂MCAL  ┃ ┣ 📂CAN  ┃ ┃ ┣ 📂Config  ┃ ┃ ┃ ┣ 📜CAN\_Cfg.h  ┃ ┃ ┃ ┗ 📜CAN\_Lcfg.c  ┃ ┃ ┣ 📂Include  ┃ ┃ ┃ ┗ 📜CAN.h  ┃ ┃ ┗ 📜CAN.c  ┃ ┣ 📂DIO  ┃ ┃ ┣ 📂Include  ┃ ┃ ┃ ┣ 📜DIO.h  ┃ ┃ ┃ ┗ 📜DIO\_Types.h  ┃ ┃ ┗ 📜DIO.c  ┃ ┣ 📂PORT  ┃ ┃ ┣ 📂Config  ┃ ┃ ┃ ┣ 📜Port\_Cfg.h  ┃ ┃ ┃ ┗ 📜Port\_Lcfg.c  ┃ ┃ ┣ 📂Include  ┃ ┃ ┃ ┣ 📜Port.h  ┃ ┃ ┃ ┗ 📜Port\_Types.h  ┃ ┃ ┗ 📜Port.c  ┃ ┗ 📂SysTick  ┃ ┃ ┣ 📂Config  ┃ ┃ ┃ ┣ 📜SysTick\_Cfg.h  ┃ ┃ ┃ ┗ 📜SysTick\_Lcfg.c  ┃ ┃ ┣ 📂Include  ┃ ┃ ┃ ┣ 📜SysTick.h  ┃ ┃ ┃ ┗ 📜SysTick\_Types.h  ┃ ┃ ┗ 📜SysTick.c  ┗ 📂Service  ┃ ┣ 📂Include  ┃ ┃ ┣ 📜BCM.h  ┃ ┃ ┗ 📜OS.h  ┃ ┣ 📜BCM.c  ┃ ┗ 📜OS.c |