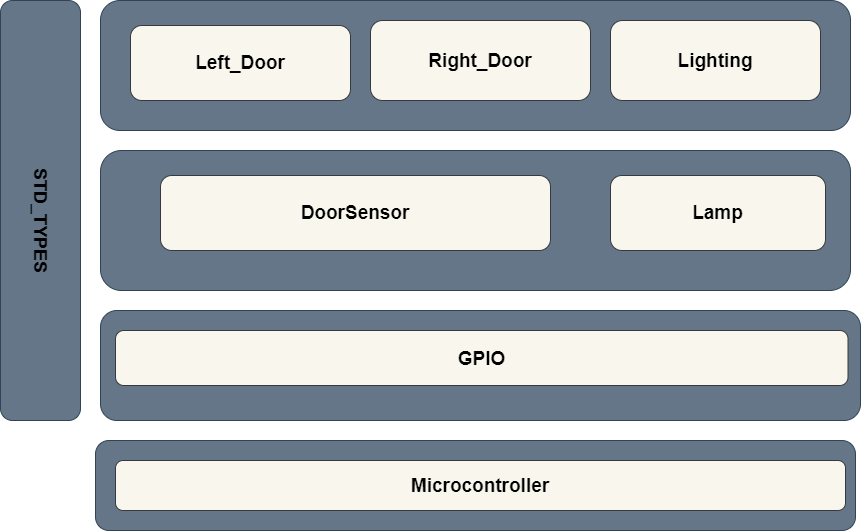
|  |
| --- |
|  |
| **Dimmer System** |
|  |

|  |
| --- |
|  |

* **Static Layered Architecture:**



* **APIs:**

|  |  |
| --- | --- |
| **Component Name** | GPIO\_Driver |
| **API Name** | Error\_Status D\_GPIO\_Init (u32 Copy\_u32Port, u8 Copy\_u8Pin, u32 Copy\_u8Mode); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status |
| **Parameters type** | u32 Copy\_u32Port   |  |  | | --- | --- | | PORT\_A | 0x01 | | PORT\_B | 0x02 | | PORT\_C | 0x04 | | PORT\_D | 0x08 | | PORT\_E | 0x10 | | PORT\_F | 0x20 | | PORT\_A | 0x01 |   u8 Copy\_u8Pin   |  |  | | --- | --- | | PIN0 | 0 | | PIN1 | 1 | | PIN2 | 2 | | PIN3 | 3 | | PIN4 | 4 | | PIN5 | 5 | | PIN6 | 6 | | PIN7 | 7 |   u8 Copy\_u8Mode   |  |  | | --- | --- | | INPUT\_PIN | 0 | | OUTPUT\_PIN | 1 | |
| **Description** | The functionality of this API to select the Mode if it’s INPUT or OUTPUT of a PIN by sending the pin number and port number as arguments |

|  |  |
| --- | --- |
| **Component Name** | GPIO\_Driver |
| **API Name** | Error\_Status D\_GPIO\_ReadPin (u32 Copy\_u32Port, u8 Copy\_u8Pin, u8\* Copy\_u8Data); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | u8 Copy\_u8Port   |  |  | | --- | --- | | PORT\_A | 0x01 | | PORT\_B | 0x02 | | PORT\_C | 0x04 | | PORT\_D | 0x08 | | PORT\_E | 0x10 | | PORT\_F | 0x20 | | PORT\_A | 0x01 |   u8 Copy\_u8Pin   |  |  | | --- | --- | | PIN0 | 0 | | PIN1 | 1 | | PIN2 | 2 | | PIN3 | 3 | | PIN4 | 4 | | PIN5 | 5 | | PIN6 | 6 | | PIN7 | 7 |   u8 \*Copy\_u32Data  It’s a pointer where the read value will be written in. |
| **Description** | The functionality of this API to read the of a PIN by sending the pin number and port number as arguments and receiving the read value in a pointer |

|  |  |
| --- | --- |
| **Component Name** | GPIO\_Driver |
| **API Name** | Error\_Status D\_GPIO\_WritePin (u32 Copy\_u32Port, u8 Copy\_u8Pin, u8 Copy\_u8Data); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | u8 Copy\_u8Port   |  |  | | --- | --- | | PORT\_A | 0x01 | | PORT\_B | 0x02 | | PORT\_C | 0x04 | | PORT\_D | 0x08 | | PORT\_E | 0x10 | | PORT\_F | 0x20 | | PORT\_A | 0x01 |   u8 Copy\_u8Pin   |  |  | | --- | --- | | PIN0 | 0 | | PIN1 | 1 | | PIN2 | 2 | | PIN3 | 3 | | PIN4 | 4 | | PIN5 | 5 | | PIN6 | 6 | | PIN7 | 7 |   u8 Copy\_u8Data   |  |  | | --- | --- | | SET\_PIN | 1 | | CLEAR\_PIN | 0 | |
| **Description** | The functionality of this API to write the Copy\_u8Data value in a PIN by sending the pin number and port number as arguments, and Copy\_u8Data |

|  |  |
| --- | --- |
| **Component Name** | DoorSensorDriver |
| **API Name** | Error\_Status DoorSensor\_Init(u8 DOOR\_SENSOR\_ID); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | u8 DOOR\_SENSOR\_ID   |  |  | | --- | --- | | DOOR\_SENSOR\_CH0 | 0 | | DOOR\_SENSOR\_CH1 | 1 |   DOOR\_SENSOR\_CH0   |  |  | | --- | --- | | SENSOR\_CH0\_PORT | PORT\_x (x: A to F) | | SENSOR\_CH0\_PIN | PINx (x: 0 to 7) | | SENSOR\_CH0\_MODE | INPUT\_PIN |   DOOR\_SENSOR\_CH1   |  |  | | --- | --- | | SENSOR\_CH1\_PORT | PORT\_x (x: A to F) | | SENSOR\_CH1\_PIN | PINx (x: 0 to 7) | | SENSOR\_CH1\_MODE | INPUT\_PIN |   SENSOR\_MODE   |  |  | | --- | --- | | INPUT\_PIN | 0 | | OUTPUT\_PIN | 1 | |
| **Description** | The functionality of this API to send the selected sensor pin number and port number as predefined arguments to GPIO. To enable its pin |

|  |  |
| --- | --- |
| **Component Name** | DoorSensorDriver |
| **API Name** | Error\_Status DoorSensor\_ReadStatus(u8 Sensor\_Ch, u8\* Sensor\_Status); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | u8 Sensor\_Ch   |  |  | | --- | --- | | DOOR\_SENSOR\_CH0 | 0 | | DOOR\_SENSOR\_CH1 | 1 |   DOOR\_SENSOR\_0   |  |  | | --- | --- | | SENSOR\_CH0\_PORT | PORT\_x (x: A to F) | | SENSOR\_CH0\_PIN | PINx (x: 0 to 7) |   DOOR\_SENSOR\_1   |  |  |  | | --- | --- | --- | | SENSOR\_CH1\_PORT | PORT\_x (x: A to F) | PORT\_x (x: A to F) | | SENSOR\_CH1\_PIN | PINx (x: 0 to 7) | PINx (x: 0 to 7) |   u8\* Sensor\_Status  it’s a pointer where the read value will be written in   |  |  | | --- | --- | | DOOROPEN | 0 | | DOORCLOSED | 1 | |
| **Description** | The functionality of this API to get the current status of the sensor and write its value in u8\* Sensor\_Status, by sending the sensor Channel number |

|  |  |
| --- | --- |
| **Component Name** | LampDriver |
| **API Name** | Error\_Status Lamp\_Init(void); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | It takes no parameters but there are Predefined values   |  |  | | --- | --- | | LIGHTING\_CH0 | LAMP\_CH0 |  |  |  |  | | --- | --- | --- | | LAMP\_PORT\_CH0 | PORT\_x (x: A to F) | 'F' | | LAMP\_PIN\_CH0 | PINx (x: 0 to 7) | 1 | |
| **Description** | The functionality of this API to send the Lamp pin number and port number as predefined arguments to GPIO. To enable its pin |

|  |  |
| --- | --- |
| **Component Name** | LampDriver |
| **API Name** | Error\_Status Lamp\_Update(u8 LAMP\_CH, u8 LAMP\_Status); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | U8 LAMP\_CH   |  |  | | --- | --- | | LIGHTING\_CH0 | LAMP\_CH0 |   Predefined values   |  |  |  | | --- | --- | --- | | LAMP\_PORT\_CH0 | PORT\_x (x: A to F) | 'F' | | LAMP\_PIN\_CH0 | PINx (x: 0 to 7) | 1 |   u8 LAMP\_Status   |  |  | | --- | --- | | L\_ON | 1 | | L\_OFF | 0 | |
| **Description** | The functionality of this API to set the selected lamp ON or OFF by sending port and pin number |

|  |  |
| --- | --- |
| **Component Name** | Lighting Driver |
| **API Name** | Error\_Status Lighting\_Init(void); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | It has no parameters |
| **Description** | The functionality of this API to call the lamp driver to initialize the predefined Lamp pin |

|  |  |
| --- | --- |
| **Component Name** | Lighting Driver |
| **API Name** | Error\_Status Lighting\_Control(u8 \*Copy\_u8LampStatus) |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | u8 Copy\_u8LampStatus  it’s a parameter carries the lamp desired status and sending it to take action upon it   |  |  | | --- | --- | | L\_ON | 1 | | L\_OFF | 0 | |
| **Description** | The functionality of this API to call the lamp driver to set the Lamp ON or OFF after checking the status of right door and left door |

|  |  |
| --- | --- |
| **Component Name** | LeftDoor Driver |
| **API Name** | Error\_Status Left\_Door(void); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | It has no parameters |
| **Description** | The functionality of this API to call the door sensor driver to initialize the predefined sensor pin |

|  |  |
| --- | --- |
| **Component Name** | LeftDoor Driver |
| **API Name** | Error\_Status LeftDoor\_ReadStatus(u8 \* Door\_Status); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | u8 \* Door\_Status  it’s a pointer carries the status of its door sensor to check if the door is opened or not to turn the lamp on or off.   |  |  | | --- | --- | | Door\_OPEN | 1 | | Door\_CLOSED | 0 | |
| **Description** | The functionality of this API to call the sensor driver to read its status if it’s OPEN or CLOSED |

|  |  |
| --- | --- |
| **Component Name** | RightDoor Driver |
| **API Name** | Error\_Status Right\_Door(void); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | It has no parameters |
| **Description** | The functionality of this API to call the door sensor driver to initialize the predefined sensor pin |

|  |  |
| --- | --- |
| **Component Name** | RightDoor Driver |
| **API Name** | Error\_Status RightDoor\_ReadStatus(u8 \* Door\_Status); |
| **Return type** | |  |  | | --- | --- | | Error\_Status values |  | | E\_OK | 0 | | E\_NOK | 1 |   Error\_Status, it returns E\_OK or E\_NO |
| **Parameters type** | u8 \* Door\_Status  it’s a pointer carries the status of its door sensor to check if the door is opened or not to turn the lamp on or off.   |  |  | | --- | --- | | Door\_OPEN | 1 | | Door\_CLOSED | 0 | |
| **Description** | The functionality of this API to call the sensor driver to read its status if it’s OPEN or CLOSED |