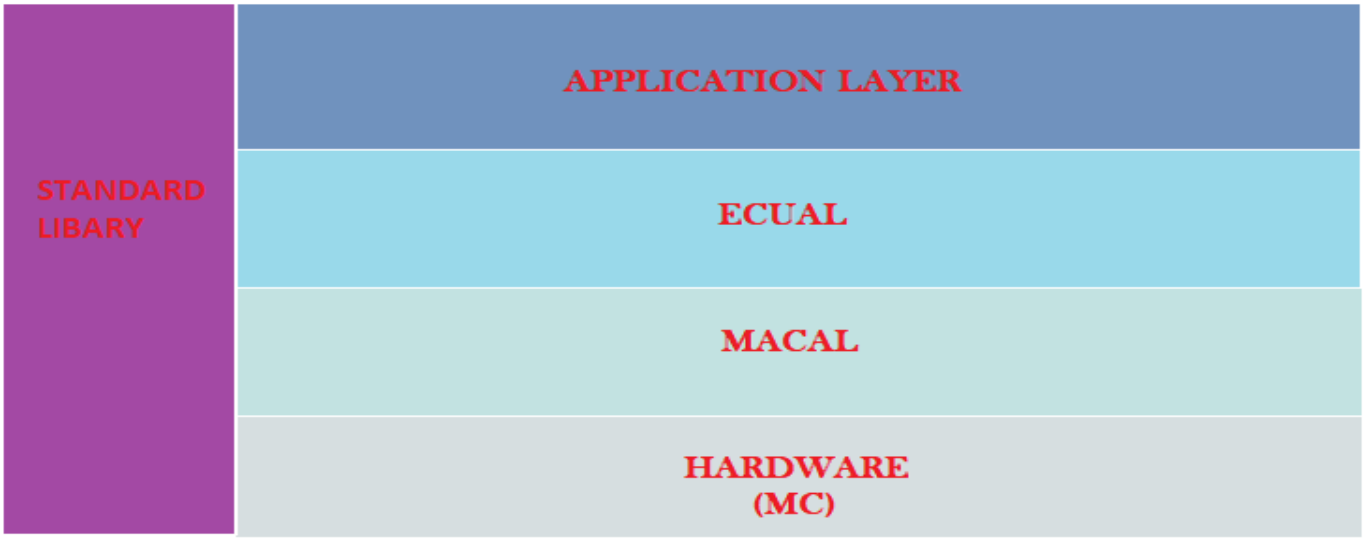
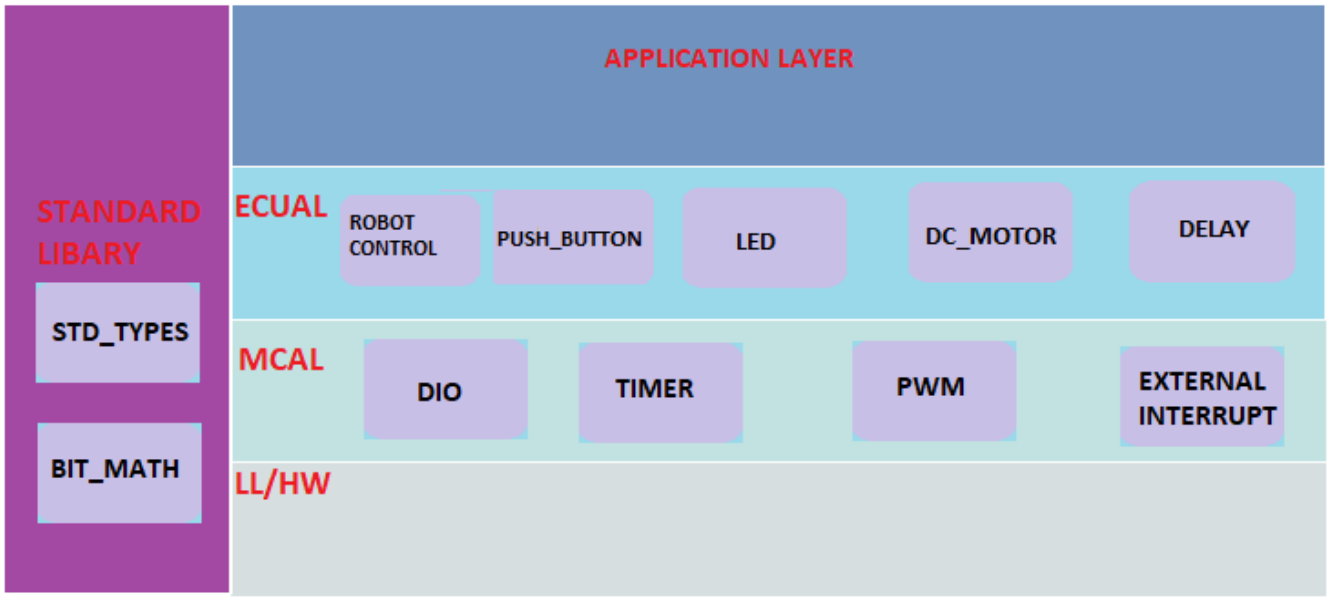
Moving Car Design

Layered architecture:



System modules:



Module Name APIs:

- ***DIO APIs:***

```
/**
 * @brief  Initialize the direction of specific pin @ref  direction_t
 * @param _pin_config   A Reference of the pin configuration  @pin_config_t
 * @return  status of the function
 *        E_OK :the function done successfully
 *        E_NOT_OK :the function has issues performing the function
 */
Std_ReturnType  DIO_pin_direction_intialize(const  pin_config_t *pin_config_ptr) ;
/**
* @brief  Write  the logic of specific pin @ref  logic_t
 * @param _pin_config   A Reference of the pin configuration  @pin_config_t
 * @param logic
 * @return status of the function
 *        E_OK :the function done successfully
 *        E_NOT_OK :the function has issues performing the function
 */
Std_ReturnType DIO_pin_write_logic(const pin_config_t *pin_config_ptr, logic_t a_logic);
/**
 * @brief  Read  the logic of specific pin @ref  logic_t
 * @param _pin_config   A Reference of the pin configuration  @pin_config_t
 * @param logic
 * @return status of the function
 *        E_OK :the function done successfully
 *        E_NOT_OK :the function has issues performing the function
 */
Std_ReturnType DIO_pin_read_logic(const  pin_config_t  * pin_config_ptr,  logic_t  *const
logic_ptr);
```

```
/**
 * @brief  Toggle  the logic of specific pin @ref  logic_t
 * @param _pin_config   A Reference of the pin configuration  @pin_config_t
 * @return status of the function
 *         E_OK :the function done successfully
 *         E_NOT_OK :the function has issues performing the function
 */
```

Std_ReturnType DIO_pin_toggle_logic(const pin_config_t *pin_config_ptr);

```
/**
 * @brief  Initialize the direction of specific pin and Initialize its logic
 * @param _pin_config A Reference of the pin configuration  @pin_config_t
 * @return status of the function
 *         E_OK :the function done successfully
 *         E_NOT_OK :the function has issues performing the function
 */
```

Std_ReturnType DIO_pin_intialize(const pin_config_t *pin_config_ptr);

```
/**
 *
 * @param port_index
 * @param direction
 * @return status of the function
 *         E_OK :the function done successfully
 *         E_NOT_OK :the function has issues performing the function
 */
```

Std_ReturnType DIO_port_direction_intialize(const port_index_t a_port_index, uint8 a_direction);

```
/**
 * @param port_index
 * @param logic
 * @return status of the function
 *         E_OK :the function done successfully
 *         E_NOT_OK :the function has issues performing the function
*/
```

Std_ReturnType DIO_port_write_logic(const port_index_t a_ port_index , uint8 a_logic);

/**

* @param port_index

* @param logic

* @return status of the function

* E_OK :the function done successfully

* E_NOT_OK :the function has issues performing the function

*/

Std_ReturnType DIO_port_read_logic(const port_index_t a_port_index , uint8 *const a_logic_ptr);

/**

* @param port_index

* @return status of the function

* E_OK :the function done successfully

* E_NOT_OK :the function has issues performing the function

*/

Std_ReturnType DIO_port_toggle_logic(const port_index_t a_port_index);

- ## *TIMER APIs:*

*/

* Description: Function to Initialize Timer Driver
* Working in Interrupt Mode
*Choose Timer initial value
* Choose Timer_ID (Timer0, Timer1, Timer2(
* Choose Timer_Mode (OverFlow, Compare(
* Choose Timer compare match value if using CTC mode
* Choose Timer_Clock

* @param _pin_config A Reference of the timer configuration

* @return status of the function

* E_OK :the function done successfully

* E_NOT_OK :the function has issues performing the function

/*

Std_ReturnType TIMER_init(const timer_config_t * timer_config_Ptr);

/*

* Description: Function to set the Call Back function address.

* @param -pointer to function & timer Id

* @return status of the function

* E_OK :the function done successfully

* E_NOT_OK :the function has issues performing the function

/*

Std_ReturnType TIMER_setCallBack(void(*a_ptr)(void), Timer_Type_t a_timer_type);

/*

* Description: Function to stop the clock of the timer to stop incrementing.

* @param timer Id

* @return status of the function

* E_OK :the function done successfully

* E_NOT_OK :the function has issues performing the function

*/

Std_ReturnType TIMER_stop(Timer_Type_t timer_type);

/*

* Description: Function to DeInit the timer to start again from beginning

* @param timer Id

* @return status of the function

* E_OK :the function done successfully

* E_NOT_OK :the function has issues performing the function

*/

Std_ReturnType TIMER_deInit(Timer_Type_t timer_type);

- *PWM APIs:*

/*

* Description: Function to Init the pwm

* @param A Reference of the pwm configuration

* @return status of the function

* E_OK :the function done successfully

* E_NOT_OK :the function has issues performing the function

*/

Std_ReturnType PWM_init(const PWM_config_t * config_Ptr);

/*

* Description: Function to DeInit the timer to start again from beginning

* @param A Reference of the pwm configuration & the desired duty cycle

* @return status of the function

* E_OK :the function done successfully

* E_NOT_OK :the function has issues performing the function

Std_ReturnType PWM_changeDuty(const PWM_config_t * config_Ptr,uint8_t a_duty);

- ## *EXTERNAL_INTERRUPT_APIs:*

```
/*
 * Description : Call the Call Back function in the application after the edge is detected
 * @param   A pointer to function  & the external interrupt id
 * @return status of the function
 *       E_OK :the function done successfully
 *       E_NOT_OK :the function has issues performing the function
```

Std_ReturnType EXT_INTx_setCallBack(void(*a_ptr)(void), const Interrupt_ID_t a_interrupt_number);

```
/*
 * Description : initialize the the dio pin to be an external interrupt
 * @param   A Reference of the external interrupt configuration
 * @return status of the function
 *       E_OK :the function done successfully
 *       E_NOT_OK :the function has issues performing the function
```

Std_ReturnType EXT_INTx_Init(const Interrupt_Config_t * Interrupt_Config_Ptr);

```
/*
 * Description : set the edge in which the external interrupt will be triggered
 * @param   edge type & the external interrupt id
 * @return status of the function
 *       E_OK :the function done successfully
 *       E_NOT_OK :the function has issues performing the function
```

Std_ReturnType EXT_INTx_setEdgeType(const Edge_type_t a_edgeType , Interrupt_ID_t a_interrupt_number);

```
/**
 * @brief DeInitialize the interrupt module
 * @param the external interrupt id
 * @return Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue to perform this action
 */
```

Std_ReturnType EXT_INTx _DeInit(const Interrupt_ID_t a_interrupt_number);

- ## *LED_APIs:*

```
/**
 * @brief Initialize the assigned pin to be OUTPUT and turn the led OFF or ON.
 * @param led : pointer to the led module configurations
 * @return Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue while performing this action
 */
```

Std_ReturnType LED_initialize(const led_t *led_ptr);

```
/**
 * @brief Turn the led on
 * @param led : pointer to the led module configurations
 * @return  Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue while performing this action
 */
```

Std_ReturnType LED_turn_on(const led_t *led_ ptr);

```
/**
 * @brief Turn the led off
 * @param led : pointer to the led module configurations
 * @return  Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue while performing this action
 */
```

Std_ReturnType LED_ turn_off (const led_t *led_ ptr);

```
/**
 * @brief Toggle the led
 * @param led : pointer to the led module configurations
 * @return  Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue while performing this action
 */
```

Std_ReturnType LED_ turn_toggle (const led_t *led_ ptr);

- ## *BUTTONS_APIs:*

```
/**
 * @brief Initialize the assigned pin to be Input.
 * @param btn pointer to the button configurations
 * @return  Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue while performing this action
 */
```

Std_ReturnType BTN_init(const button_t *btn_ptr);

```
/**
 * @brief Read the state of the button
 * @param btn pointer to the button configurations
 * @param btn_state button state @ref button_state_t
 * @return  Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue while performing this action
```

```
*/
```
Std_ReturnType BTN_read_state(const button_t *btn_ptr, button_state_t *btn_state_ptr);

- ## *MOTORS_APIs:*

```
/**
 * @brief Initialize the assigned pins to be OUTPUT and turn the motor OFF or ON.
 * @param _dc_motor pointer to the motor configurations
 * @return Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue to perform this action
 */
```
Std_ReturnType DC_MOTOR_init(const dc_motor_t *dc_motor_ptr);
```
/**
 * @brief Move the motor to the right direction
 * @param _dc_motor pointer to the motor configurations
 * @return Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue to perform this action
 */
```
Std_ReturnType DC_MOTOR_move_right(const dc_motor_t *dc_motor_ptr);
```
/**
 * @brief Move the motor to the left direction
 * @param _dc_motor pointer to the motor configurations
 * @return Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue to perform this action
 */
```
Std_ReturnType DC_MOTOR_move_left(const dc_motor_t *dc_motor_ptr);
```
/**
 * @brief stop the motor movement
 * @param _dc_motor pointer to the motor configurations
 * @return Status of the function
 *       (E_OK) : The function done successfully
 *       (E_NOT_OK) : The function has issue to perform this action
 */
```
Std_ReturnType DC_MOTOR_stop(const dc_motor_t *dc_motor_ptr);