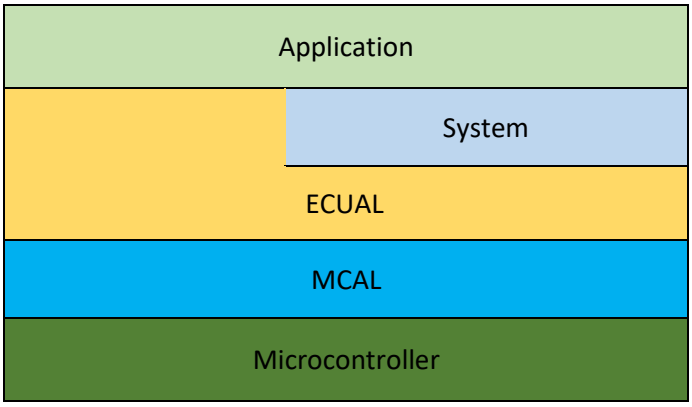
Moving Car System Design

Layered Architecture



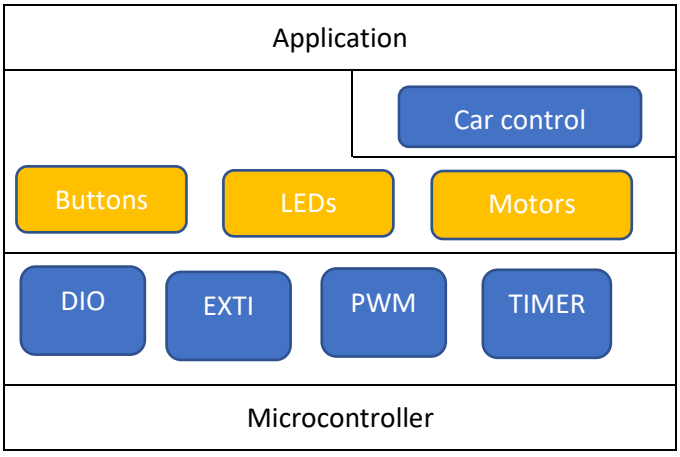
MCAL: All modules related to the internal microcontroller peripherals.

ECUAL: All drivers for all the external hardware connected to the microcontroller.

System: Modules to provide an application interface for some of the external hardware.

Application: The project's main logic.

System Modules



APIs

- **DIO APIs**

```c
/**
 * @def function to configure the direction of all DIO ports
 *           and set the initial values for pins
 */
void DIO_voidInit(void);

/**
 * @def function to configure a single DIO pin as input/output
 * @param Copy_Port the port of the required pin
 * @param Copy_Pin the pin number in the given port
 * @return error status
 */
uint8_t DIO_u8SetPinDir(uint8_t Copy_Port,uint8_t Copy_Pin, uint8_t Copy_Dir);


/**
 * @def function to configure a single DIO pin as input/output
 * @param Copy_Port the port of the required pin
 * @param Copy_Pin the pin number in the given port
 * @param Copy_Value desired value (high or low) to set the pin to
 * @return error status
 */
uint8_t DIO_u8SetPinVal(uint8_t Copy_Port,uint8_t Copy_Pin, uint8_t Copy_Value);

/**
 * @def function to configure a single DIO pin as input/output
 * @param Copy_Port the port of the required pin
 * @param Copy_Pin the pin number in the given port
 * @param Copy_Value desired value (high or low) to set the entire port to
 * @return error status
 */
uint8_t DIO_u8SetPortVal(uint8_t Copy_Port, uint8_t Copy_Value);

/**
 * @def function to get the value of a single DIO pin whether high or low
 * @param Copy_Port the port of the required pin
 * @param Copy_Pin the pin number in the given port
 * @param Copy_pu8Val pointer to a variable to store pin value (0-255)
 * @return error status
 */
uint8_t DIO_u8GetPinVal(uint8_t Copy_Port,uint8_t Copy_Pin, uint8_t* Copy_pu8Val);

/**
 * @def toggle the output value on the given DIO pin
 * @param Copy_Port the port of the required pin
 * @param Copy_Pin the pin number in the given port
 * @return error status
 */
uint8_t DIO_u8TogglePin(uint8_t Copy_Port,uint8_t Copy_Pin);
```

- **EXTI (External Interrupt) APIs**

```c
void EXTI_Init(void);

/**
 * Enable given interrupt
*/
EN_EXTIError_t EXTI_EnableInt(uint8_t Copy_u8IntNumber);

/**
 * Disable given interrupt
*/
EN_EXTIError_t EXTI_DisableInt(uint8_t Copy_u8IntNumber);

/**
 * Set function to call back if given interrupt occurs
 */
EN_EXTIError_t EXTI_SetCallback(uint8_t Copy_IntNumber, void (*Copy_Function)(void));
```

- **PWM APIs**

```c
/**
 * Initialize PWM
*/
void PWM_Init(void);

/**
 * generate signal on given pin with frequency according to given speed
*/
EN_PWMError_t PWM_SetSpeed(uint8_t Copy_port, uint8_t Copy_pin, uint8_t Copy_speed);
```

- **TIMER APIs**

```c
/**
 * Initialize the timer
*/
void Timer_Init(void);

/**
 * Adds delay of the given duration in the given units
*/
uint8_t Delay(uint8_t Copy_delay, EN_TimeUnit_t Copy_timeUnit);

/**
 * Set function to call back if timer interrupt is enabled
 */
uint8_t Timer_SetCallback(void (*Copy_pvFunction)(void));
```

ECUAL

- **LEDs APIs**

```
/**
 * enables displaying output on given led
*/
EN_LEDErrorState_t SW_u8EnableLED(ST_LED* Copy_LED);

/**
 * Disables displaying output on given led
 */
EN_LEDErrorState_t SW_u8DisableSwitch(ST_LED* Copy_LED);

/**
 * Set the state of the given led to On/Off
 */
EN_LEDErrorState_t LED_setState(ST_LED* Copy_LED, EN_LEDState Copy_LEDState);

/**
 * Toggles the given given led
 */

EN_LEDErrorState_t LED_Toggle(ST_LED* Copy_Led);
```

- **Buttons APIs**

```
/**
 * enables reading from given switch
*/
EN_SWError_t SW_u8EnableSwitch(ST_Switch* Copy_Switch);

/**
 * Disables reading from given switch
*/
EN_SWError_t SW_u8DisableSwitch(ST_Switch* Copy_Switch);

/**
 * Read the value on the switch pin and stores it in given reference
*/
EN_SWError_t SW_u8ReadSwitch(ST_Switch* Copy_Switch, EN_SWValue_t* Copy_SwitchValue);
```

- **Motors APIs**

```
/**
 * @def start motor with given speed
 * @param Copy_Motor the desired motor
 * @return error status
 */
EN_MotorError_t DCM_Start(ST_Motor_t Copy_Motor, uint8_t Copy_Speed);

/**
 * @def Stop the motor
 *
 */
EN_MotorError_t DCM_Stop(ST_Motor_t Copy_Motor);
```

System

- **Car Control APIs**

```
EN_CarError_t CAR_DriveForward(ST_Motor_t *Motors, uint8_t Copy_Speed);

EN_CarError_t CAR_TurnRight(u8 Copy_u8IntNumber);

EN_CarError_t CAR_TurnLeft(u8 Copy_u8IntNumber);
```