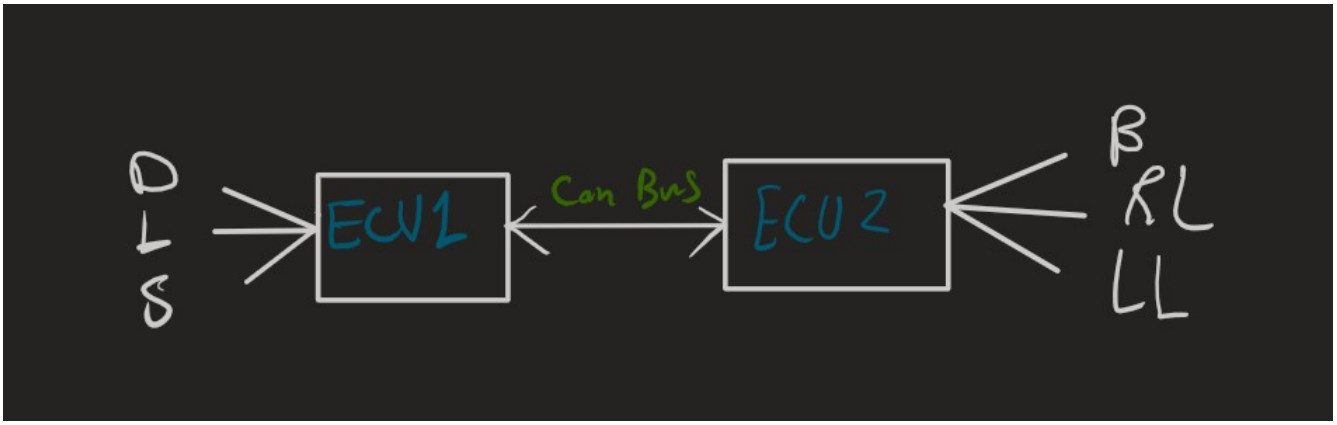
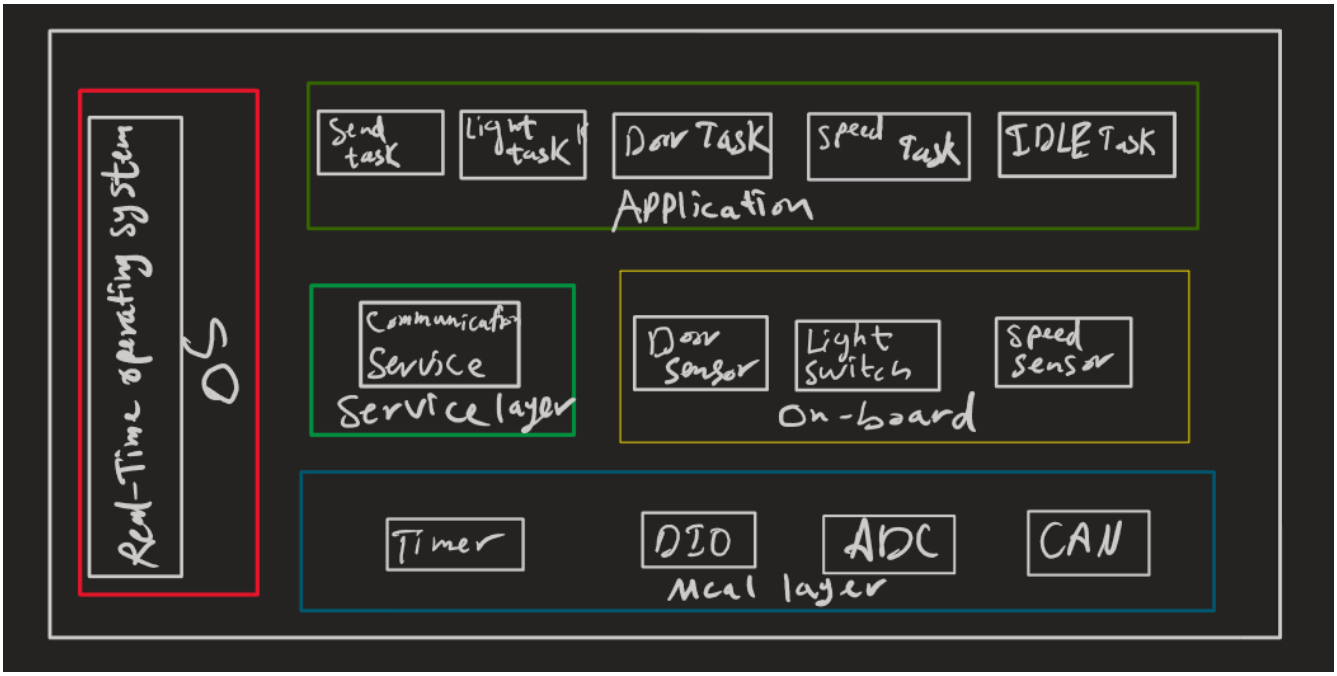
Hardware Connections:

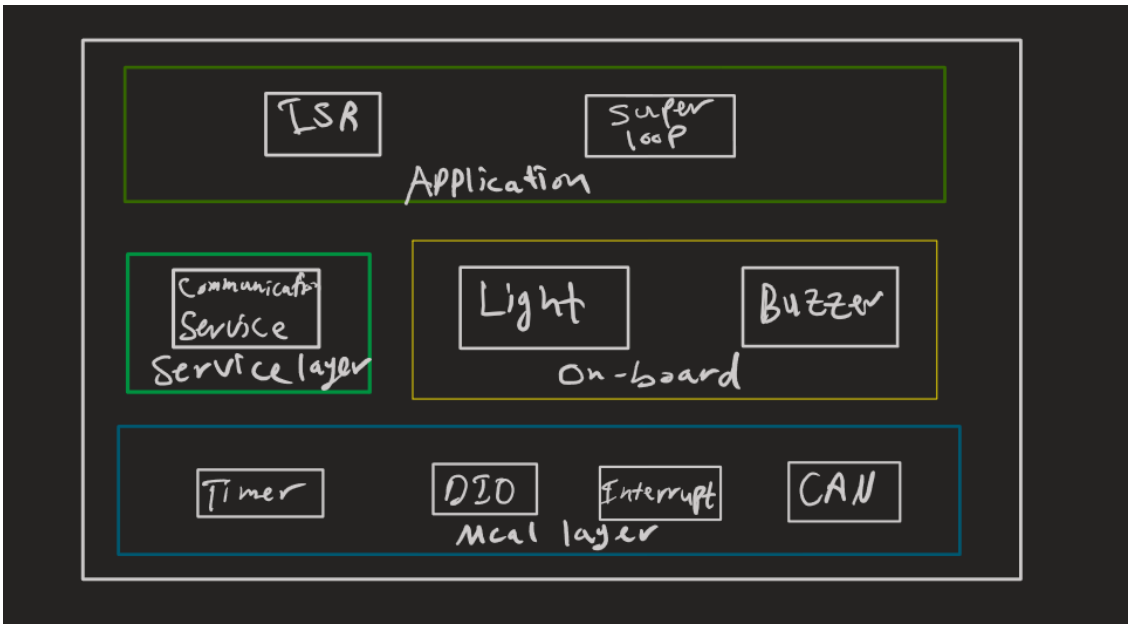


Static Desing:

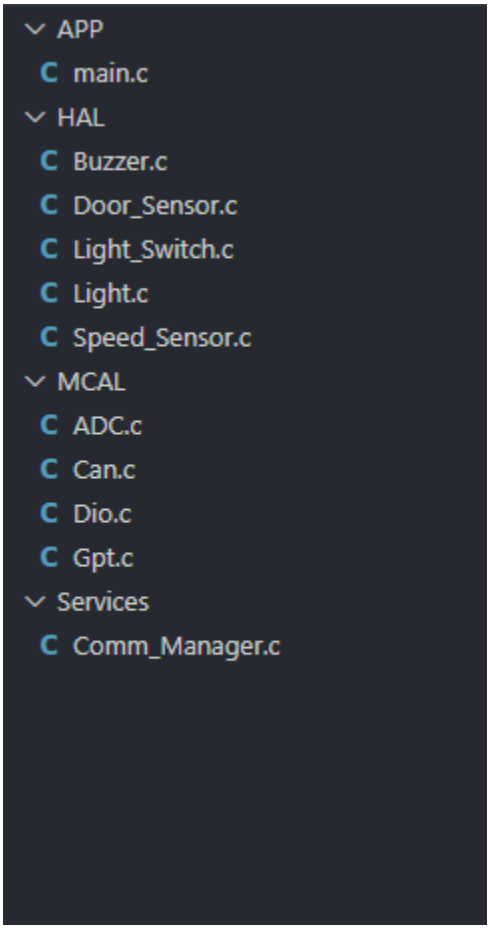
ECU1 runs Operating System:



ECU2:



Folder Structure:



* GPT (General Purpose timer) Module:

Required APIs:

```c
/************************************************************************\
 *          * Function Info *                                           *
 *                                                                      *
 * Syntax               : GPT_init                                     *
 * Description          : initialize all timers drivers                *
 * Sync\Async           : Synchronous                                  *
 * Parameters (in)      : config (array of the required configurations)*
 * Parameters (out)     : None                                         *
 * Return value:        : void                                         *
 ************************************************************************/
void GPT_init(GPT_Config_Ptr* config);

/************************************************************************\
 *          * Function Info *                                           *
 *                                                                      *
 * Syntax               : GPT_Start                                    *
 * Description          : start the timer for required ticks and fire ISR  *
 * Sync\Async           : Synchronous                                  *
 * Parameters (in)      : Number of ticks to count                     *
 * Parameters (out)     : None                                         *
 * Return value:        : void                                         *
 ************************************************************************/
void GPT_Start(uint32_t ticks);
/************************************************************************\
 *          * Function Info *                                           *
 *                                                                      *
 * Syntax               : GPT_Stop                                     *
 * Description          : stop the timer                               *
 * Sync\Async           : Synchronous                                  *
 * Parameters (in)      : None                                         *
 * Parameters (out)     : None                                         *
 * Return value:        : void                                         *
 ************************************************************************/
void GPT_Stop(void);

/************************************************************************\
 *          * Function Info *                                           *
 *                                                                      *
 * Syntax               : GPT_Delay                                    *
 * Description          : Blocking delay                               *
 * Sync\Async           : Synchronous                                  *
 * Parameters (in)      : delay (number of ms to wait)                 *
 * Parameters (out)     : None                                         *
 * Return value:        : void                                         *
 ************************************************************************/
void GPT_Delay(uint32_t delay);
```

* DIO (Digital Input Output) module:
Required APIs:

```c
/*************************************************************************\
 *          * Function Info *                                            *
 *                                                                       *
 * Syntax               : DIO_init                                       *
 * Description          : Initialize DIO Driver                          *
 * Sync\Async           : Synchronous                                    *
 * Parameters (in)      : config (pointer to array of configurations)    *
 * Parameters (out)     : None                                           *
 * Return value:        : void                                           *
 *************************************************************************/
void DIO_init(DIO_Config_Ptr* config);



/*************************************************************************\
 *          * Function Info *                                            *
 *                                                                       *
 * Syntax               : DIO_Write                                      *
 * Description          : write hight or low to output pin               *
 * Sync\Async           : Synchronous                                    *
 * Parameters (in)      :   pin (structure to port and pin number)       *
 *                          value (High or Low)                          *
 * Parameters (out)     : None                                           *
 * Return value:        : void                                           *
 *************************************************************************/
void DIO_Write(DIO_pin pin , uint8_t value);

/*************************************************************************\
 *          * Function Info *                                            *
 *                                                                       *
 * Syntax               : DIO_Read                                       *
 * Description          : Read the value of input pin                    *
 * Sync\Async           : Synchronous                                    *
 * Parameters (in)      : pin (structure to port and pin number)         *
 * Parameters (out)     : value (High or Low)                            *
 * Return value:        : void                                           *
 *************************************************************************/
void DIO_Read(DIO_pin pin , uint8_t* value);

/*************************************************************************\
 *          * Function Info *                                            *
 *                                                                       *
 * Syntax               : DIO_Toggle                                     *
 * Description          : Toggle value of output pin                     *
 * Sync\Async           : Synchronous                                    *
 * Parameters (in)      : pin (structure to port and pin number)         *
 * Parameters (out)     : Non                                            *
 * Return value:        : void                                           *
 *************************************************************************/
void DIO_Toggle(DIO_pin pin);
```

* CAN Module:

Required APIs:

```
/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : CAN_init                                             *
 * Description          : initialize all CAN modules                           *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : config (array of the required configurations)        *
 * Parameters (out)     : None                                                 *
 * Return value:        : void                                                 *
 *******************************************************************************/
void CAN_init(CAN_Config_Ptr* config);

/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : CAN_Send                                             *
 * Description          : Send data using CAN Bus                              *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : data (data to be send)                               *
 * Parameters (out)     : None                                                 *
 * Return value:        : void                                                 *
 *******************************************************************************/
void CAN_Send(uint32_t* data);

/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : CAN_Receive                                          *
 * Description          : Receive data using CAN Bus                           *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : Non                                                  *
 * Parameters (out)     : data (the received data)                             *
 * Return value:        : void                                                 *
 *******************************************************************************/
void CAN_Receive(uint32_t* data);
```

* ADC (Analoge to Digital Converter) Module:
Required APIs:

```c
/*************************************************************************\
 *          * Function Info *                                            *
 *                                                                       *
 * Syntax               : ADC_init                                       *
 * Description          : initialize all pins as Analoge inputs          *
 * Sync\Async           : Synchronous                                    *
 * Parameters (in)      : config (array of the required configurations)  *
 * Parameters (out)     : None                                           *
 * Return value:        : void                                           *
 *************************************************************************/
void ADC_init(ADC_Config_Ptr* config);

/*************************************************************************\
 *          * Function Info *                                            *
 *                                                                       *
 * Syntax               : ADC_Read                                       *
 * Description          : Read the current value of ADC pin              *
 * Sync\Async           : Synchronous                                    *
 * Parameters (in)      : pin (structure to port and pin)                *
 * Parameters (out)     : value (Actuel value of the ADC pin)            *
 * Return value:        : void                                           *
 *************************************************************************/
void ADC_Read(ADC_pin pin , uint8_t* value);
```

* Door Sensor Required APIs:

```c
/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : Door_Sensor_init                                     *
 * Description          : Initialize the door sensor                           *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : config (array of the required configurations)        *
 * Parameters (out)     : None                                                 *
 * Return value:        : void                                                 *
 *******************************************************************************/
void Door_Sensor_init(Door_Sensor_Config_Ptr* config);

/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : Door_Sensor_Read                                     *
 * Description          : read the state of door closed or open                *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : pin (structure of port and pin)                      *
 * Parameters (out)     : value (Current state of the door)                    *
 * Return value:        : void                                                 *
 *******************************************************************************/
void Door_Sensor_Read(Door_Sensor_pin pin , uint8_t* value);
```

* Light Switch Required APIs:

```c
/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : Light_Switch_init                                    *
 * Description          : Initialize the light switch                          *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : config (array of the required configurations)        *
 * Parameters (out)     : None                                                 *
 * Return value:        : void                                                 *
 *******************************************************************************/
void Light_Switch_init(Light_Switch_Config_Ptr* config);

/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : Light_Switch_Read                                    *
 * Description          : Read the state of light switch                       *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : pin (struture of port and pin)                       *
 * Parameters (out)     : value (current state of the light)                   *
 * Return value:        : void                                                 *
 *******************************************************************************/
void Light_Switch_Read(Light_Switch_pin pin , uint8_t* value);
```

* Speed Sensor Required APIs:

```c
/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : Speed_Sensor_init                                    *
 * Description          : Initialize the speed sensor                          *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : config (array of the required configurations)        *
 * Parameters (out)     : None                                                 *
 * Return value:        : void                                                 *
 *******************************************************************************/
void Speed_Sensor_init(Speed_Sensor_Config_Ptr* config);

/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : Speed_Sensor_Read                                    *
 * Description          : Read speed value                                     *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : pin (struture of port and pin)                       *
 * Parameters (out)     : value (Current speed value)                          *
 * Return value:        : void                                                 *
 *******************************************************************************/
void Speed_Sensor_Read(Speed_Sensor_pin pin , uint8_t* value);
```

* Light Required APIs:

```c
/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : Light_init                                           *
 * Description          : Initialize the light                                 *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : config (array of the required configurations)        *
 * Parameters (out)     : None                                                 *
 * Return value:        : void                                                 *
 *******************************************************************************/
void Light_init(Light_Config_Ptr* config);

/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : Light_On                                             *
 * Description          : turn on the light                                    *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : pin (struture of port and pin)                       *
 * Parameters (out)     : None                                                 *
 * Return value:        : void                                                 *
 *******************************************************************************/
void Light_On(Light_pin pin);

/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax               : Light_Off                                            *
 * Description          : turn off the light                                   *
 * Sync\Async           : Synchronous                                          *
 * Parameters (in)      : pin (struture of port and pin)                       *
 * Parameters (out)     : None                                                 *
 * Return value:        : void                                                 *
 *******************************************************************************/
void Light_Off(Light_pin pin);
```

* Buzzer Required APIs:

```c
/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax              : Buzzer_init                                           *
 * Description         : initialize the buzzer                                 *
 * Sync\Async          : Synchronous                                           *
 * Parameters (in)     : config (array of the required configurations)         *
 * Parameters (out)    : None                                                  *
 * Return value:       : void                                                  *
 *******************************************************************************/
void Buzzer_init(Buzzer_Config_Ptr* config);

/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax              : Buzzer_On                                             *
 * Description         : turn on the buzzer                                    *
 * Sync\Async          : Synchronous                                           *
 * Parameters (in)     : pin (struture of port and pin)                        *
 * Parameters (out)    : None                                                  *
 * Return value:       : void                                                  *
 *******************************************************************************/
void Buzzer_On(Buzzer_pin pin);

/*******************************************************************************\
 *          * Function Info *                                                  *
 *                                                                             *
 * Syntax              : Buzzer_Off                                            *
 * Description         : turn off the buzzer                                   *
 * Sync\Async          : Synchronous                                           *
 * Parameters (in)     : pin (struture of port and pin)                        *
 * Parameters (out)    : None                                                  *
 * Return value:       : void                                                  *
 *******************************************************************************/
void Buzzer_Off(Buzzer_pin pin);
```