EgFWD

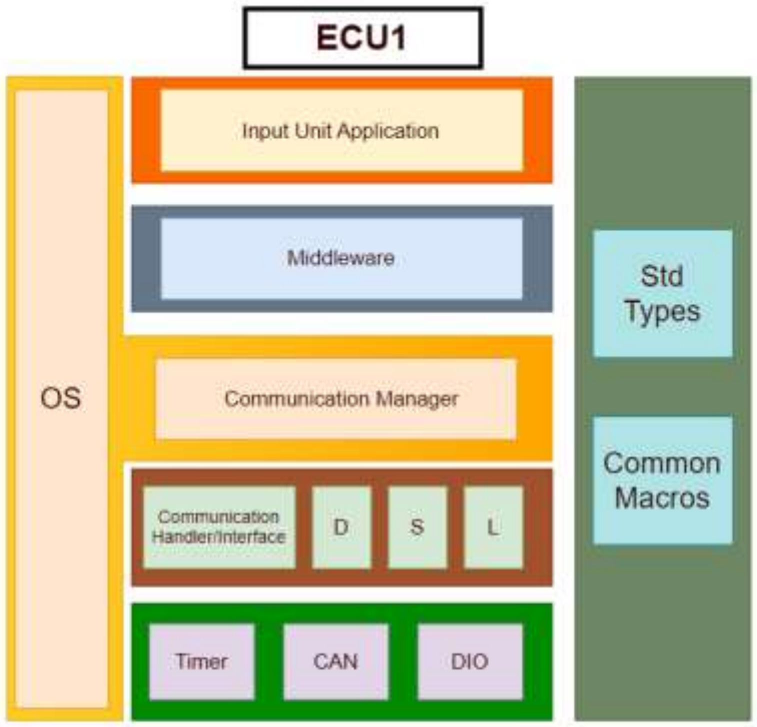
Advanced Embedded Systems

Project (3) | System Design

Automotive Door Control System Design Report

For ECU 1:

1) Make the layered architecture



2) Specify ECU components and modules

a. Application: Where system logic and tasks are implemented.
    i. main.c
b. Middleware: For routing the system to its desired destination. For example, either to send data from input devices and so being routed to communication manager or else saving this data in EEPROM and so being routed to memory manager.
    i. middleware.c | middleware.h
c. Service:
    i. RTOS: Handling tasks and operating system as stated in project rubric.
        1. All RTOS related files from previous course(RTOS Porting Section | memory management, heap, tasks.c, config.h …etc)
    ii. Communication(BCM): Used to send Status messages.
        1. BCM.c, BCM.h
d. Common: For standard data types and common macros.
    i. std_types.h
    ii. common_macros.h
    iii. communication_types.h
e. HAL:
    i. Onboard/ECUAL: containing Door sensor, Light switch, Speed sensor and all external input devices.
        1. doorSensor.c, doorSensor.h
        2. lightSwitch.c, lightSwitch.h

3. speedSensor.c, speedSensor.h

ii. MCAL:
   1. Timer
      a. timer.c, timer.h
      b. timer_cfg.c, timer_cfg.h
   2. CAN
      a. CAN.c, CAN.h
      b. CAN_cfg.c, CAN_cfg.h
   3. DIO
      a. DIO.c. DIO.h
      b. DIO_cfg.c, DIO_cfg.h

3) Provide full detailed APIs for each module as well as a detailed description for the used typedefs

- ❖ Application Module:
  - ➢ API-Functions:
    - App_InitError_t App_init ();
      - To create tasks and call all modules' init APIs as GPIO, Timer…etc.
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - void App_Start ();
      - All program logic goes here.
      - Reentrant, Asynchronous, Non-Recursive Function.
    - void timerSetCallback();
      - Non-Reentrant, Synchronous, Non-Recursive Function.
  - ➢ API-Types:
    - App_InitError
      - To return initialization state, either success or fail.
- ❖ Middleware Module:
  - ➢ API-Functions:
    - Middleware_SendState SendData (commProtocol_t);
      - To send data using specified communication protocol.
      - Reentrant, Asynchronous, Non-Recursive Function.
    - Middleware_SaveState SaveData (memoryType_t, dataSize_t);

            - To save data either on internal EEPROM or external one.
            - Reentrant, Asynchronous, Non-Recursive Function.
    - API-Types:
        - Middleware_SaveState:
            - To return save state, either success or fail.
        - Middleware_SendState:
            - To return send state, either success or fail.
- BCM Module:
    - API-Function:
        - Send_CAN (CAN_config_ptr);
            - Reentrant, Asynchronous, Non-Recursive Function.
        - Send_UART (UART_config_ptr);
            - Reentrant, Asynchronous, Non-Recursive Function.
        - Send_SPI (SPI_config_ptr);
            - Reentrant, Asynchronous, Non-Recursive Function.
        - Send_I2C (I2C_config_ptr);
            - Reentrant, Asynchronous, Non-Recursive Function.
    - API-Types:
        - BCM_SendState:
            - To return send state, either success or fail.
- Std_Types Module:
    - API-Types: (mentioned below are just a few examples, feel free to add what's needed)
        - uint8, uint16, uint32, int8, int16, int32…etc.
        - Boolean_t
        - NULL
- Communication_Types Module:
    - API-Types:
        - commProtocol_t
        - baudRate_t
        - parityType_t
        - memoryType_t

- dataSize_t

- Common_Macros Module:
  - API-Functions: (mentioned below are just a few examples, feel free to add what's needed)
    - SetBit(Port, Pin)
      - Non-Reentrant, Asynchronous, Non-Recursive Function.
    - ClearBit(Port, Pin)
      - Non-Reentrant, Asynchronous, Non-Recursive Function.
    - ToggleBit(Port, Pin)
      - Non-Reentrant, Asynchronous, Non-Recursive Function.
    - RotateRight(Port, n)
      - Non-Reentrant, Asynchronous, Non-Recursive Function.
    - RotateLeft(Port, n)
      - Non-Reentrant, Asynchronous, Non-Recursive Function.
- Timer Module:
  - API-Function:
    - Timer_initState Timer_init(Timer_config_ptr_t);
      - Reentrant, Asynchronous, Non-Recursive Function.
    - void Timer_Start();
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - void Timer_Stop();
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - void Timer_notification();
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - uint32_t Timer_getElapsedTime();
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - uint32_t Timer_ getRemainingTime ();
      - Non-Reentrant, Asynchronous, Non-Recursive Function.
    - void Timer_delay (uint32_t);
      - Non-Reentrant, Synchronous, Non-Recursive Function.

        - void Timer_delayMicroseconds(uint32_t);
            - Non-Reentrant, Synchronous, Non-Recursive Function.
    - API-Types:
        - Timer_Length_t
        - Timer_Mode_t
        - Timer_config_ptr_t
- CAN Module:
    - API-Functions
        - CAN_initState CAN_init(CAN_config_ptr_t);
            - Reentrant, Asynchronous, Non-Recursive Function.
        - void CAN_Send();
            - Non-Reentrant, Synchronous, Non-Recursive Function.
        - void CAN_Receive();
            - Non-Reentrant, Synchronous, Non-Recursive Function.
    - API-Types:
        - CAN_sendState
        - CAN_receiveState
        - CAN_initState
        - CAN_config_ptr_t
- DIO Module:
    - API-Function:
        - Dio_initState DIO_init(DIO_config_ptr_t);
            - Reentrant, Asynchronous, Non-Recursive Function.
        - uint8_t DIO_write(Port_t, Pin_t, LevelType_t);
            - Non-Reentrant, Asynchronous, Non-Recursive Function.
        - uint8_t DIO_toggle(Port_t, Pin_t);
            - Non-Reentrant, Asynchronous, Non-Recursive Function.
        - uint8_t DIO_read(Port_t, Pin_t);
            - Non-Reentrant, Asynchronous, Non-Recursive Function.
        - Uint32_t DIO_readPort(Port_t, Pin_t);
            - Non-Reentrant, Asynchronous, Non-Recursive Function.

        - Uint32_t DIO_writePort(Port_t);
            - Non-Reentrant, Asynchronous, Non-Recursive Function.
    - API-Types:
        - Port_t
        - Pin_t
        - LevelType_t
        - Direction_t
        - AttachType_t: PULL_UP, PULL_DOWN, OPEN_DRAIN
        - PinType_t: ADC, Normal…etc
        - Current_t: how much can a pin withstand current, as in Arm 2,4,8mA
        - Dio_config_ptr_t
        - Dio_initState
            - Either initialization success or fail.

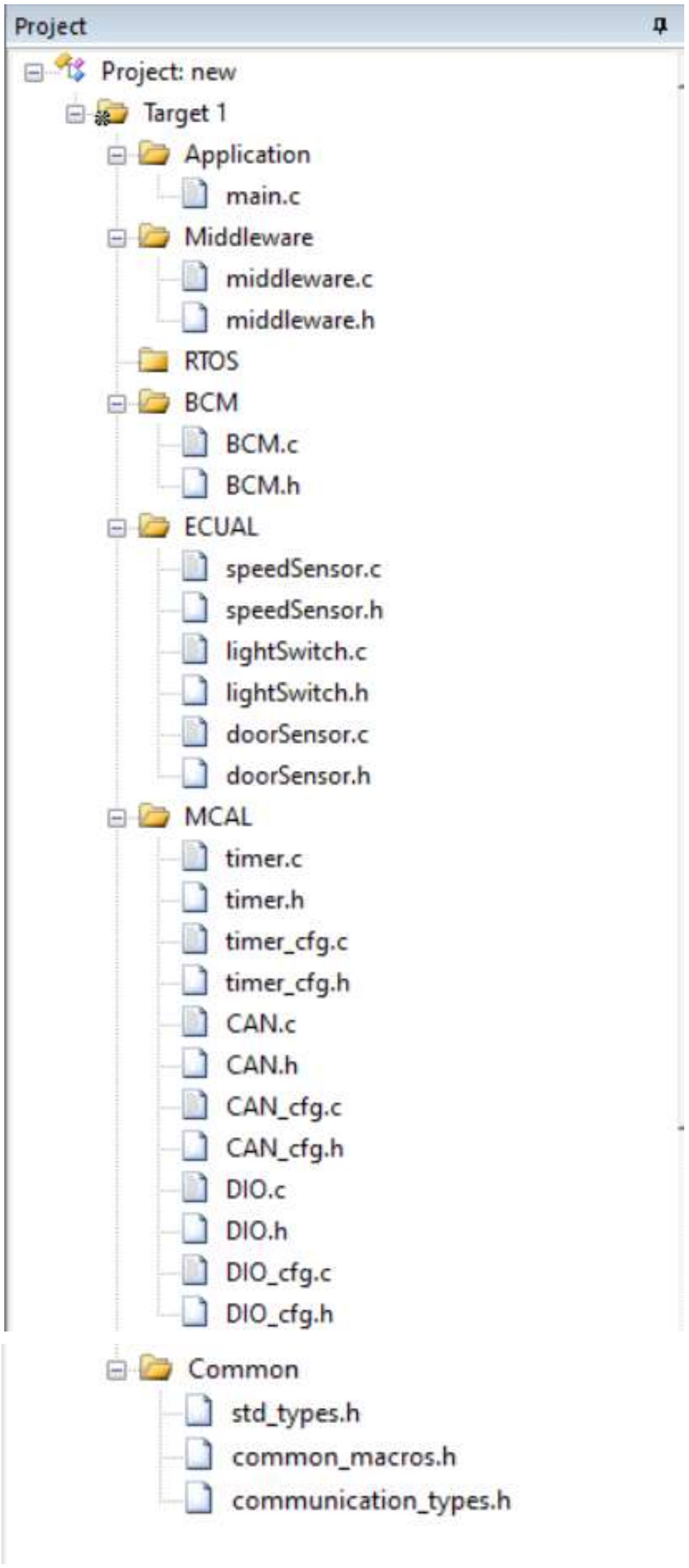
*PREVIOUS MODULES ARE SAME FOR BOTH ECUs, NEXT 3 INPUT MODULES ARE WHAT'S DIFFERENT

- Door Sensor Module:
    - API-Function:
        - Door_initState Door_init(Door_config_ptr_t);
            - Reentrant, Asynchronous, Non-Recursive Function.
        - Boolean_t DoorOpen(Port_t, Pin_t);
            - Non-Reentrant, Synchronous, Non-Recursive Function.
        - Boolean _t DoorClose(Port_t, Pin_t);
            - Non-Reentrant, Synchronous, Non-Recursive Function.
        - LevelType_t DoorState(Port_t, Pin_t);
            - Non-Reentrant, Synchronous, Non-Recursive Function.
    - API-Types:
        - Door_config_ptr_t
        - Door_initState
            - Either initialization success or fail.
- Light Switch Module:
    - API-Function:

        - LightSwitch_initState LightSwitch_init(LightSwitch _config_ptr_t);
            - Reentrant, Asynchronous, Non-Recursive Function.
            - Non-Reentrant, Synchronous, Non-Recursive Function.
        - void LightSwitchPressed (void);
            - ISR for light switch.
        - LevelType_t LightSwitchState(Port_t, Pin_t);
            - Non-Reentrant, Synchronous, Non-Recursive Function.
    - API-Types:
        - LightSwitch_config_ptr_t
        - LightSwitch_initState
            - Either initialization success or fail.
- Speed Sensor Module:
    - API-Function:
        - SpeedSensor _initState SpeedSensor _init(SpeedSensor _config_ptr_t);
            - Reentrant, Asynchronous, Non-Recursive Function.
        - uint32_t SpeedSensorValue (void);
            - returns speed read from speed sensor.
            - Non-Reentrant, Synchronous, Non-Recursive Function.
    - API-Types:
        - SpeedSensor _config_ptr_t
        - SpeedSensor_initState
            - Either initialization success or fail.

4) Prepare your folder structure according to the previous points

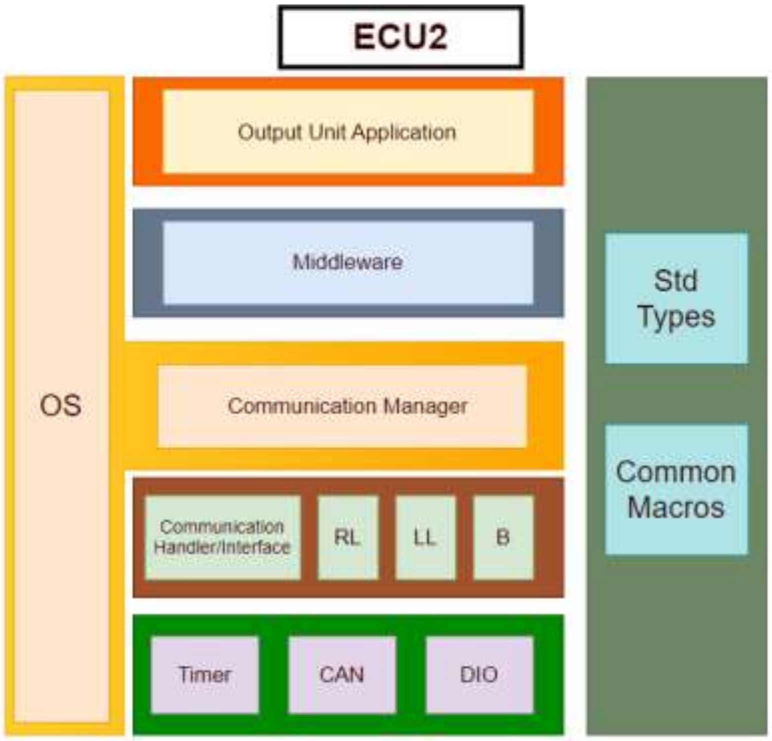
- Screenshot from Keil



```
Project
Project: new
  Target 1
    Application
      main.c
    Middleware
      middleware.c
      middleware.h
    RTOS
    BCM
      BCM.c
      BCM.h
    ECUAL
      speedSensor.c
      speedSensor.h
      lightSwitch.c
      lightSwitch.h
      doorSensor.c
      doorSensor.h
    MCAL
      timer.c
      timer.h
      timer_cfg.c
      timer_cfg.h
      CAN.c
      CAN.h
      CAN_cfg.c
      CAN_cfg.h
      DIO.c
      DIO.h
      DIO_cfg.c
      DIO_cfg.h
    Common
      std_types.h
      common_macros.h
      communication_types.h
```

For ECU 2:

1. Make the layered architecture



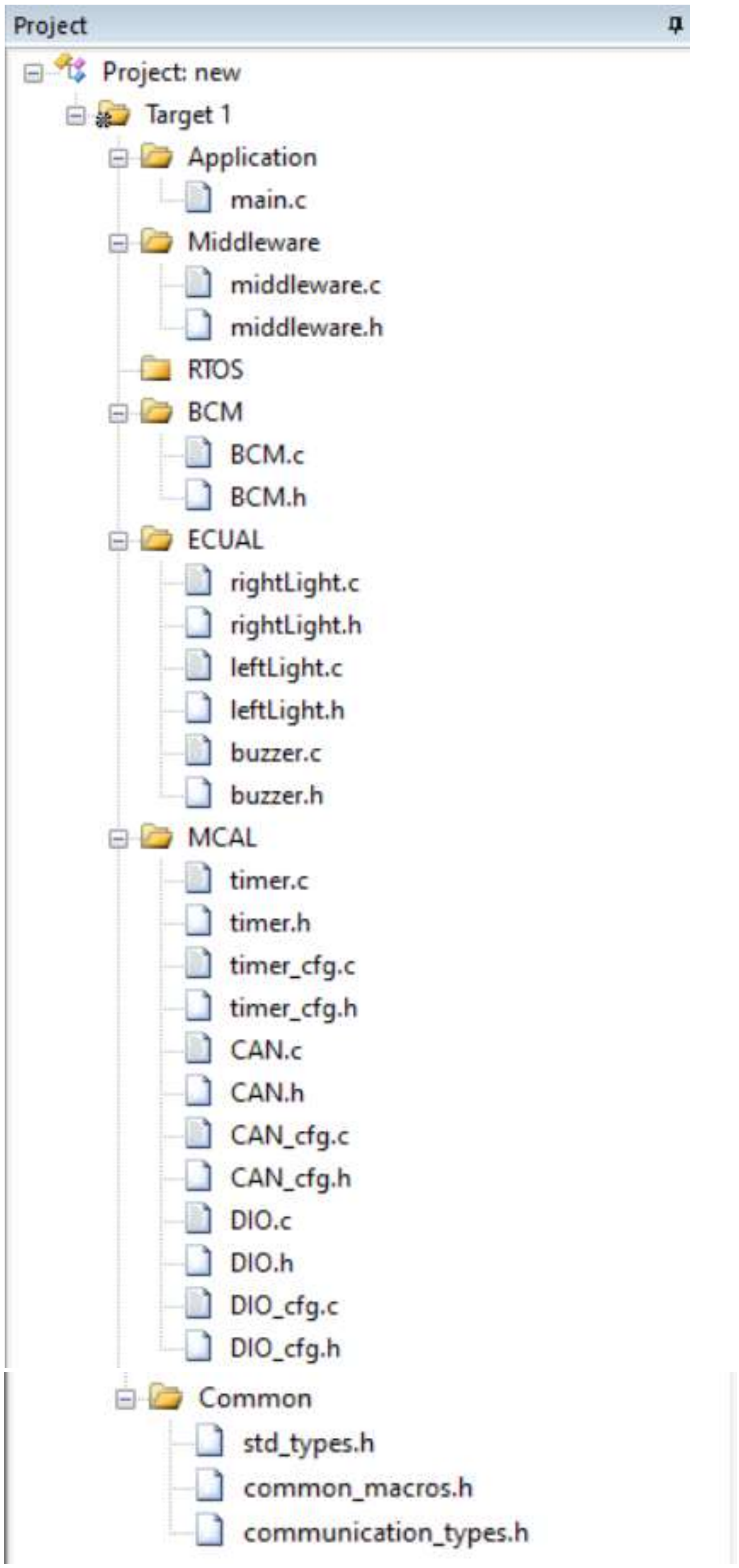
2. Specify ECU components and modules
    1. Application: Where system logic and tasks are implemented.
    2. Middleware: For routing the system to its desired destination.
    3. RTOS: Handling tasks and operating system as stated in project rubric.
    4. Common: For standard data types and common macros.
    5. HAL:
        1. Onboard/ECUAL: containing Right Light , Left Light, Buzzer and all external output devices.
        2. MCAL: containing Timer, CAN and DIO modules.
3. Provide full detailed APIs for each module as well as a detailed description for the used typedefs
    - ❖ Application Module
    - ❖ Middleware Module
    - ❖ BCM Module
    - ❖ Std_Types Module
    - ❖ Common_Macros Module
    - ❖ Timer Module
    - ❖ CAN Module
    - ❖ DIO Module

*PREVIOUS MODULES ARE SAME FOR BOTH ECUs, NEXT 3 OUTPUT MODULES ARE WHAT'S DIFFERENT

- ❖ Right Light Module:
  - ➢ API-Function:
    - RightLight _initState RightLight _init(RightLight _config_ptr_t);
      - Reentrant, Asynchronous, Non-Recursive Function.
    - Boolean_t RightLightOn(Port_t, Pin_t);
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - Boolean _t RightLightOff(Port_t, Pin_t);
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - LevelType_t RightLightState(Port_t, Pin_t);
      - Non-Reentrant, Synchronous, Non-Recursive Function.
  - ➢ API-Types:
    - RightLight_config_ptr_t
    - RightLight_initState
      - Either initialization success or fail.
- ❖ Left Light Module:
  - ➢ API-Function:
    - LeftLight _initState LeftLight _init(LeftLight _config_ptr_t);
      - Reentrant, Asynchronous, Non-Recursive Function.
    - Boolean_t LeftLightOn (Port_t, Pin_t);
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - Boolean _t LeftLightOff(Port_t, Pin_t);
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - LevelType_t LeftLightState(Port_t, Pin_t);
      - Non-Reentrant, Synchronous, Non-Recursive Function.
  - ➢ API-Types:
    - LeftLight_config_ptr_t
    - LeftLight_initState
      - Either initialization success or fail.

- ❖ Buzzer Module:
  - ➢ API-Function:
    - ▪ Buzzer_initState Buzzer_init(Buzzer_config_ptr_t);
      - Reentrant, Asynchronous, Non-Recursive Function.
    - ▪ Boolean_t Buzzer_On (void);
      - returns speed read from speed sensor.
      - Non-Reentrant, Synchronous, Non-Recursive Function.
    - ▪ Boolean_t Buzzer_Off (void);
      - returns speed read from speed sensor.
      - Non-Reentrant, Synchronous, Non-Recursive Function.
  - ➢ API-Types:
    - ▪ Buzzer_config_ptr_t
    - ▪ Buzzer_initState
      - Either initialization success or fail.

4. Prepare your folder structure according to the previous points



```
Project
Project: new
  Target 1
    Application
      main.c
    Middleware
      middleware.c
      middleware.h
    RTOS
    BCM
      BCM.c
      BCM.h
    ECUAL
      rightLight.c
      rightLight.h
      leftLight.c
      leftLight.h
      buzzer.c
      buzzer.h
    MCAL
      timer.c
      timer.h
      timer_cfg.c
      timer_cfg.h
      CAN.c
      CAN.h
      CAN_cfg.c
      CAN_cfg.h
      DIO.c
      DIO.h
      DIO_cfg.c
      DIO_cfg.h
    Common
      std_types.h
      common_macros.h
      communication_types.h
```