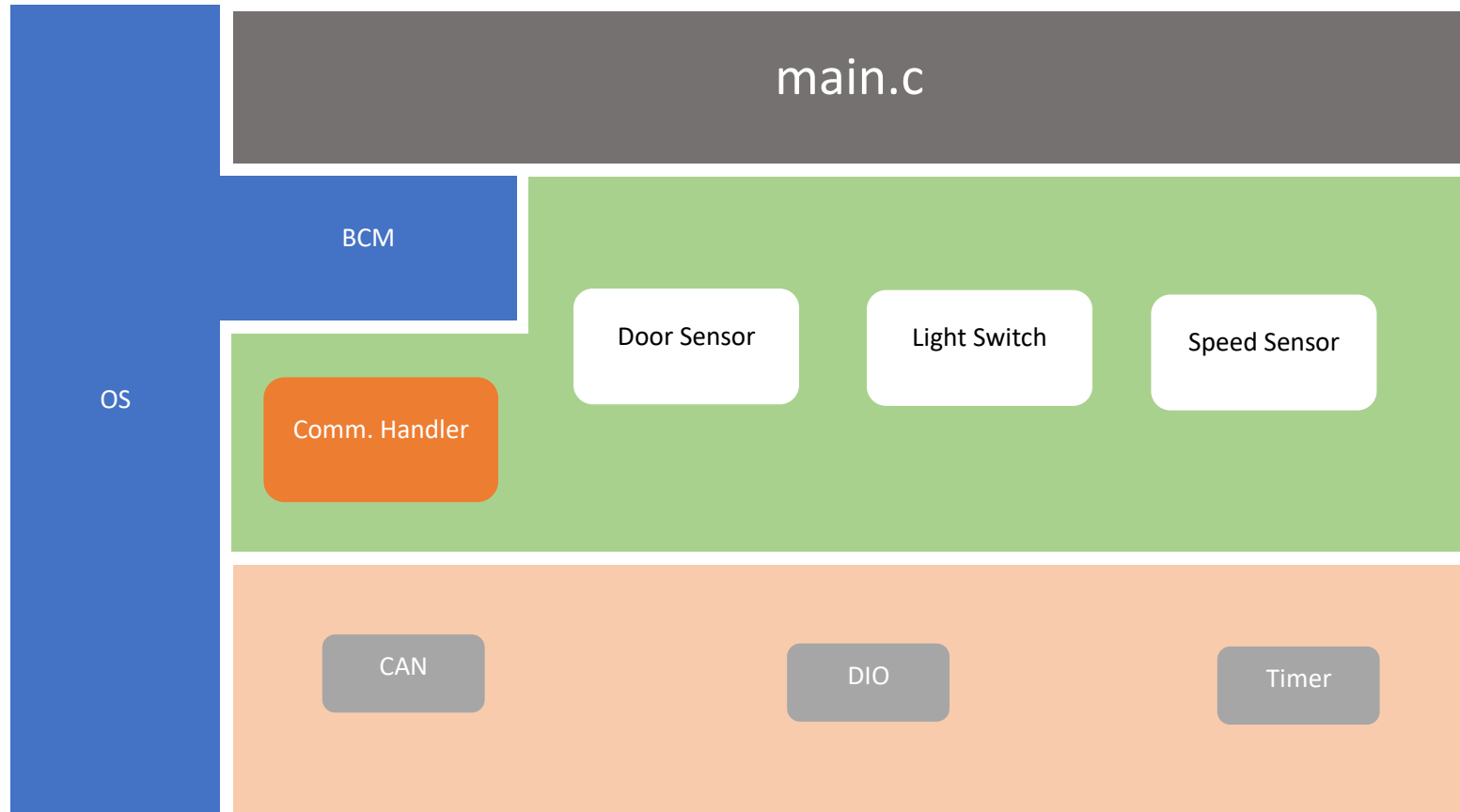


ECU1:



APIs:

1. Door sensor

- `state door(void):`

input(s): None

return: State of door (HIGH/LOW)

description: returns HIGH if door is open and LOW if door is closed

2. Light switch

- `state switch(void):`

input(s): None

return: State of the switch (HIGH/LOW)

description: returns HIGH if switch is on and LOW if switch is off

3. Speed sensor

- `state motion(void):`

description: returns HIGH if the car is moving and LOW if car is stationary

input(s): None

return: State of the motion (HIGH/LOW)

4. comm. Handler

- `void send_data(uint32 data):`

description: stores the readings of all sensors and send it to CAN peripheral

input(s): required data to be sent

return: None

5. CAN

- CAN_init(void):

description: initialize CAN communication protocol

input(s): None

return: None

- CAN_send(<parameters>,uint32 data):

description: specify information about the sender and the data being sent

input(s): <parameters> such as message id, its crc , etc... , data: the value wanted to be sent

return: None

6. DIO

- DIO_init(DIO_dir direction):

description: initialize DIO for the specified pin and its direction (input/output)

input(s): Pin direction (INPUT/OUTPUT)

return: None

- DIO_Read(DIO_port port ,DIO_pin pin):

description: read specified pin value

input(s): required port we want to read from, and the required pin we want to know its value

return: None

7. Timer

- `Timer_init(timer_name timer):`

description: initialize the required timer and its interrupt

input(s): the name of the timer we want to work with

return: None

- `Timer_SetTick(uint32 ticks):`

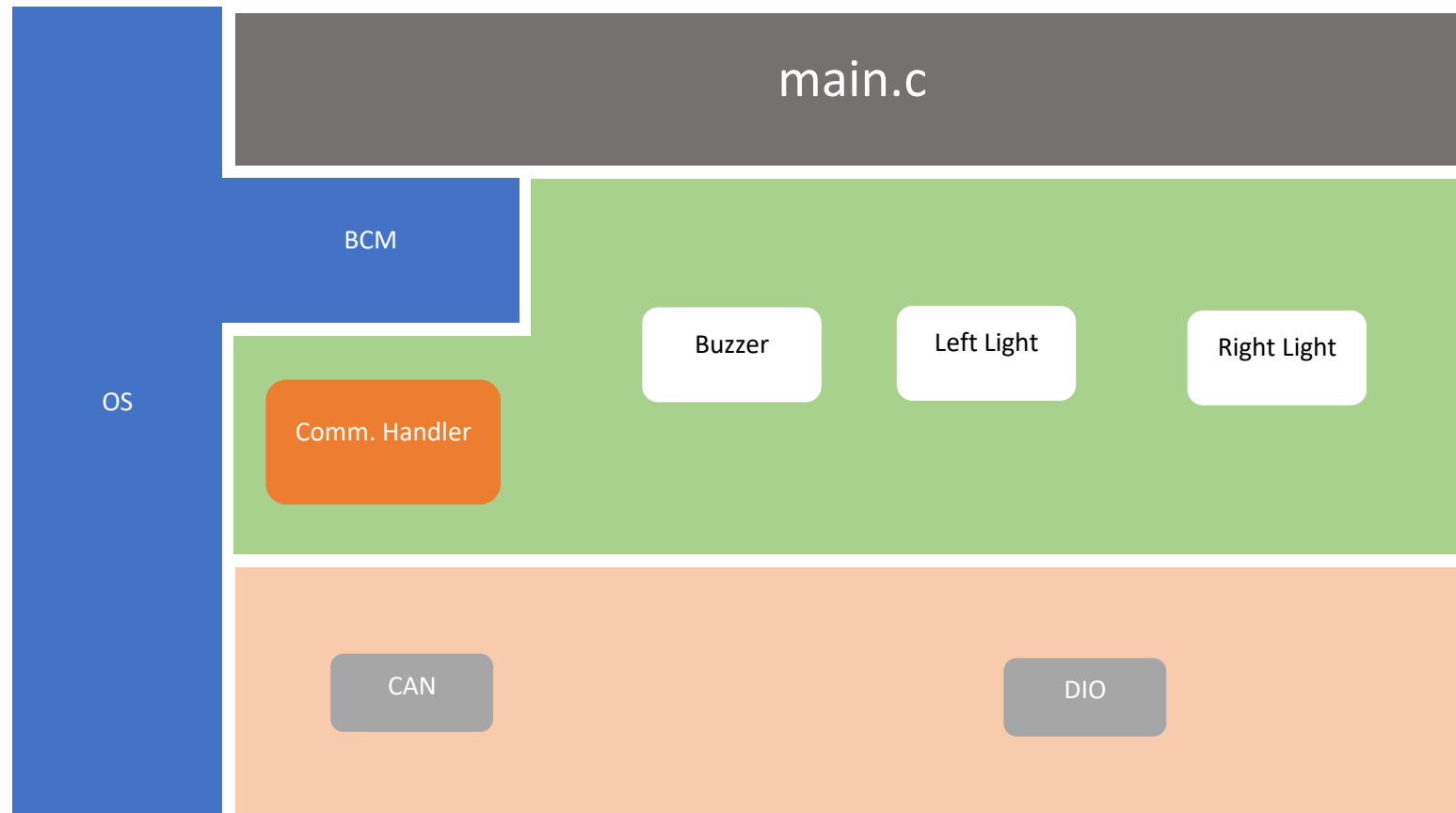
description: set the time required for 1 tick (in this project we will set it to 5 ms)

input(s): the equivalent value of the OCR to correspond to 5 ms

return: None

- `Void Timer_Handler(void):` ISR

ECU2:



APIS:

1. Lights (left and right)

- Void Lights (bool Value):

description: turns lights on or off

input(s): HIGH to turn lights on or LOW to turn lights off

return: None

2. Buzzer

- Void Buzzer (bool Value):

description: turns buzzer on or off

input(s): HIGH to turn buzzer on or LOW to turn buzzer off

return: None

3. comm. Handler

- void send_data(uint32 data):

description: send the readings of all sensors to CAN peripheral

input(s): the variable containing the readings of all sensors

return: None

4. CAN

- CAN_init(void):

description: initialize CAN communication protocol

input(s): None

return: None

- CAN_read(void):

description: Get the required data from emu1

input(s): None

return: variable containing all sensors readings

5. DIO

- DIO_init(DIO_dir direction):

description: initialize DIO for the specified pin and its direction (input/output)

input(s): Pin direction (INPUT/OUTPUT)

return: None

- DIO_Write(DIO_port port ,DIO_pin pin,bool value):

description: write the specified pin with required value (high/low)

input(s): required port we want to modify, and the required pin we want to set/clear its value,value we want to write

return: None

enums:

DIO_port: names of ports in the ecu (e.g.: PORT0,PORT1,PORTA,PORTV,...)













DIO_pin: names of pins in the port (e.g.: pin1,pin2,pin3,...)











Bool: consists of HIGH and LOW values

timer_name: names of timers (e.g.: T1A,T1B,...)

state: consists of ON and OFF values

Folder structure

- ▼  ECU1
 - >  Includes
 - ▼  App
 - >  main.c
 - ▼  HAL
 - >  Door
 - >  Light_Switch
 - >  Speed
 - ▼  MCAL
 - >  CAN
 - >  DIO
 - >  Timer
 - ...

- ▼  ECU2
 - >  Includes
 - ▼  App
 - >  main.c
 - ▼  HAL
 - >  Buzzer
 - >  Lights
 - ▼  MCAL
 - >  CAN
 - >  DIO